XI Міжнародна науково-практична конференція
XI Международная научно-практическая конференция
11th International Conference

ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ АСПЕКТИ ПОБУДОВИ ПРОГРАМНИХ СИСТЕМ

ТЕОРЕТИЧЕСКИЕ И ПРАКТИЧЕСКИЕ АСПЕКТЫ ПОСТРОЕНИЯ ПРОГРАММНЫХ СИСТЕМ

THEORETICAL AND APPLIED ASPECT OF PROGRAM SYSTEMS DEVELOPMENT

TAAPSD'2014

Праці конференції
Труды конференции
Proceeding

15-17 грудня 2014 року
Київ

УДК: 004

Теоретичні та прикладні аспекти побудови програмних систем: матеріали міжнародної наукової конференції, м. Київ, 15-17 грудня 2014 р. / редкол.: Д.Б. Буй на ін. – Кіровоград: ПП «Центр оперативної поліграфії «Авангард»», – 2014. – 280 с.

В збірнику зібрані праці XI Міжнародної науково-практичної конференції «Теоретичні та прикладні аспекти побудови програмних систем» **TAAPSD'2014,** яка проходила в м. Київ 15-17 грудня 2014 р.

УДК: 004

XI Міжнародна науково-практична конференція
«ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ АСПЕКТИ ПОБУДОВИ ПРОГРАМНИХ СИСТЕМ»
(Україна, Київ, 15-17 грудня 2014 року)

Міністерство освіти і науки України, Київський Національний Університет імені Тараса Шевченка, Національний університет «Києво-Могилянська Академія», Кіровоградський державний педагогічний університет імені Володимира Винниченка, технічний університет м. Кошице (Словацька республіка), університет Менделя м. Брно (Чеська республіка) проводять **XI Міжнародну науково-практичну конференцію «Теоретичні та прикладні аспекти побудови програмних систем» (TAAPSD'2014).**
Конференція відбулася в м. Києві, 15-17 грудня 2014 року на базі Інституту післядипломної освіти Київського національного університету імені Тараса Шевченка.

НАУКОВІ НАПРЯМИ КОНФЕРЕНЦІЇ

- Теорія програмування
- Формальні методи розробки програмних систем
- Верифікація і тестування програмних систем
- Представлення даних
- Логічні методи обробки комп'ютерних знань
- Інтелектуальні інформаційні системи
- Захист та кодування інформації
- Системні методології
- Моделі предметних областей та математичні методи їх дослідження
- Технології та сучасний інструментарій програмування
- Індустріальні основи виробництва програм
- Web-програмування та електронна комерція
- Розподілені інформаційно-обчислювальні системи і середовища
- Інноваційні методи і технології навчання
- Бази даних/знань
- Моделі предметних областей та математичні методи їх дослідження
- Інші (за погодженням з програмним комітетом)

ОФІЦІЙНІ РОБОЧІ МОВИ КОНФЕРЕНЦІЇ:

українська, російська, англійська

ПРОГРАМНИЙ КОМІТЕТ

– завідувач кафедри математичної інформатики, декан факультету кібернетики Київського Національного університету імені Тараса Шевченка, доктор фізико-математичних наук, член-кореспондент НАН України, заслужений діяч науки і техніки України, професор **А. В. Анісімов, голова**;
– академік НАН України, професор кафедри теоретичної кібернетики Київського Національного університету імені Тараса Шевченка, доктор фізико-математичних наук, професор **В. Н. Редько, голова**;
– професор кафедри теорії і технології програмування Київського Національного університету імені Тараса Шевченка, доктор фізико-математичних наук, професор **Д.Б.Буй, заст. голови**;
– завідувач кафедри теорії і технології програмування Київського Національного університету імені Тараса Шевченка, доктор фізико-математичних наук, професор **М.С. Нікітченко, заст. голови**;

О.В. Авраменко (Україна)	О.Б. Стеля (Україна)
Ю.В. Бойко (Україна)	В.В. Пасічник (Україна)
В.В. Бублик (Україна)	С.Д. Погорілий (Україна)
М.М. Глибовець (Україна)	М.О. Сидоров (Україна)
С.П. Горлач (Германія)	О.Л. Синявський (Україна)
С.С. Гороховський (Україна)	В.Г. Скобелєв (Україна)
Ш. Гудак (Словакія)	С.Ф. Теленик (Україна)
А .Ю. Дорошенко (Україна)	Х. Турулл-Торез (Нова Зеландія)
В. І. Кудін (Україна)	К. Хаенсген (Германія)
К.М. Лавріщева (РФ)	В.С. Харченко (Україна)
Г.В. Сандраков (Україна	С.С. Шкільняк (Україна)

ОРГАНІЗАЦІЙНИЙ КОМІТЕТ

- Д.Б. Буй, голова
- О.С. Сенченко, заст. голови
- Т.В. Россада, секретар
- М.М. Рубан, секретар

ЗМІСТ

РЕАЛІЗАЦІЯ ПОСЛУГ ЗАХИСТУ ІНФОРМАЦІЇ В SAAS МОДЕЛІ ОБСЛУГОВУВАННЯ ХМАРИ

І.Ф. Аулов
Харківський національний університет радіоелектроніки, Україна
auluv@iit.com.ua

Вступ

У зв'язку з поширенням мережі Інтернет все більше компаній та держустанов переходить до моделі надання послуг з використанням веб-застосунків.

Необхідність реалізації механізмів автентифікації та ідентифікації користувачів, забезпечення конфіденційності та цілісності даних, надання послуг ЕЦП в веб-застосунках SaaS, де основними вимогами є робота на різних платформах (переносимість), інтероперабельність та швидкодія, потребує вирішення задачі пошуку найбільш оптимального рішення за критерієм переносимість/швидкодія.

Метою дослідження є порівняння різних підходів до реалізації криптографічних бібліотек для застосування в веб-застосунках SaaS моделі надання послуг в хмарах.

Реалізація криптографічної бібліотеки як додатку до браузера на мові C/C++

В загальному випадку додатки до веб-браузера надають можливість взаємодіяти з нативними бібліотеками, що реалізовані з використанням мови програмування C/C++, з кодом на JavaScript із використанням можливостей стандартного інтерфейсу додатку та маршалінгу основних типів даних.

Сьогодні найбільше розповсюдження отримав інтерфейс додатків NPAPI[1], що підтримується практично усіма браузерами для ПК, окрім Internet Explorer, який використовує технологію ActiveX. Альтернативою використання додатків NPAPI, є інтерфейс додатків Google Native Client (NaCl), що підтримується в браузерах Google Chrome та ОС Chrome OS. Його поява пов'язана з наміром Google відмовитися від підтримки NPAPI інтерфейсу.

Головними перевагами використання додатків в браузері є їх швидкодія та можливість доступу до апаратного забезпечення комп'ютера. Порівняно з звичайними програмами для ПК, додатки потребують додаткові обчислювальні ресурси на здійснення маршалінгу даних, що передаються між додатком та браузером. В той же час суттєвим недоліком, що стримує розповсюдження додатків, є необхідність створення додатку окремо для кожного з браузерів, а у випадку інтерфейсу NPAPI, додаток повинен бути скомпільований під кожну з платформ.

Реалізація криптографічної бібліотеки на мові Java

За допомогою мови програмування Java можна створювати криптографічні бібліотеки для веб-застосунків у вигляді Java-апплетів. Використання Java-апплетів дозволяє реалізувати кросплатформені веб-застосунки, переклавши вирішення проблеми підтримки різних платформ та браузерів на розробників середовища виконання Java (JRE) [2].

Взаємодія між JavaScript браузером та Java-апплетом відбувається за допомогою технології LiveConnect, що дозволяє надавати безпосередній доступ до функцій апплета та виконувати операції над основними типами даних.

До переваг використання Java-апплетів можна віднести роботу в браузерах та на платформах, що підтримуються JRE, швидкодію, можливість доступу до апаратних засобів з використанням JNI та файлової системи. Недоліком технології Java-апплетів є необхідність використання JRE та часті зміни в політиці безпеки Java-апплетів, що потребує частих оновлень.

Реалізація криптографічної бібліотеки на мові JavaScript

В першу чергу, використання мови JavaScript для реалізації криптографічної бібліотеки для веб-застосунків вирішує задачі інтероперабельності та роботи на різних платформах за рахунок її підтримки більшістю сучасних браузерів.

З появою типізованої підмножини JavaScript - asm.js [3], яка дозволяє визначати типи змінних на етапі компіляції, та оптимізації інтерпретаторів браузерів для використання asm.js, дозволило підвищити швидкодію веб-застосунків розроблених з використанням мови JavaScript та зробило доцільним їх використання порівняно з Java-апплетами або додатками до браузера, які написані на C/C++.

На сьогодні основним недоліком мови JavaScript є відсутність стандартних способів безпосередньо з JavaScript, не використовуючи додатково додатки до веб-браузерів або java-апплети отримати доступ до підключених апаратних засобів захисту інформації, наприклад, електронних ключів.

Порівняння криптографічних бібліотек

В якості стенду для тестування використовувався комп'ютер з ОС Windows 7 x64, процесор Intel Core i3 3,3 GHz, ОЗП 4 GB, веб-браузер Google Chrome 32 v38, JRE 8.25. Бібліотеки було скомпільовані з використанням максимальної оптимізації за швидкодією. Для кожної з функцій, що тестувалася, було виконано

100 незалежних вимірювань часу та обрано найменший з них. Результати порівняння бібліотек представлені в таблиці 1.

Таблиця 1. Результати порівняння бібліотек за швидкодією

Показник	Мова програмування		
	C/C++	Java	JavaScript
1. Розмір бібліотеки, МБ	4,1	0,7	4,3
2. Пам'ять, МБ	16	4,5(~320)	17(~500)
3. Швидкість гешування (ГОСТ 34.311-95), Мб/с	285	53	153
4. Швидкість шифрування (ГОСТ 28147), Мб/с	465	220	272
5. Швидкість ЕЦП (ДСТУ 4145-2002, поле 257), мс	0,7/4,2	17,2/21,5	2,8/18,2
6. Швидкість НШ (ДГ - в гр.т. е.кр., поле 571), мс	6,8/6,8	50,9/51,8	21,6/21,5

Висновки

Використання мови JavaScript для криптографічної бібліотеки в веб-застосунках дозволяє реалізувати підтримку основних платформ, в тому числі і мобільних. Це в сукупності з швидкодією, яка в середньому в п'ять разів гірша за реалізацію на мові C/C++, дозволяє використовувати їх в веб-застосунках для кінцевих користувачів.

Використанням компілятора Emscripten [4] дозволило підвищити швидкість розробки JavaScript бібліотеки за рахунок використання коду бібліотеки C/C++. Перевагою використання підмножини JavaScript - asm.js є збільшення швидкодії в браузерах, що підтримують asm.js, та сумісність з браузерами, що її не підтримують.

Недоліком використання JavaScript бібліотеки є можливість модифікації бібліотеки сторонніми модулями JavaScript.

Список використаних джерел

1. MDN. Plug-in Basics: [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/en-US/Add-ons/Plugins/Gecko_Plugin_API_Reference/Plug-in_Basics
2. The Java Tutorials. Java Applets: [Електронний ресурс]. – Режим доступу: https://docs.oracle.com/javase/tutorial/deployment/applet/
3. asm.js: [Електронний ресурс]. – Режим доступу: http://asmjs.org/

4. Emscripten: An LLVM-to-JavaScript Compiler: [Електронний ресурс]. – Режим доступу: http://kripken.github.io/emscripten-site/

SOME APPROXIMATE PROPERTIES OF CESARO MEANS

M.S. Bakuridze, M.I. Didmanidze
Batumi Shota Rustaveli State University, Batumi, Georgia
mari.didmanidze@gmail.com

In this paper we establish some new properties $\sigma_n^a(x,f)$ and $t_n^a(x,f)$. Further assume that

$$g(n,f)=\frac{1}{n}\int_{\frac{\pi}{n}}^{\pi}\frac{\omega(t,f)_c}{t^2}dt \qquad \text{Here we note that}$$

$$\omega\left(\frac{1}{n},\ f\right)\le g(n,\ f)\ .$$

Theorem 1. a) Let $a\in(0,\ 1),\ p\in(1,+\infty)$ and $ap>1$. Than for the function $f\in C(T)$ the following inequality is true

$$\left\|\sigma_n^{-a}(f)-f\right\|_c\le A(p,a)n^a\omega\left(\frac{1}{n},f\right)_p+A(a)g(n,f)$$

b) Let $p\in(1,+\infty)$. Than for the function $f\in C(T)$ the following inequality is true

$$\left\|\sigma_n^{-\frac{1}{p}}(f)-f\right\|_c\le A(p)\left[n^{\frac{1}{p}}(\ln n)^{1-\frac{1}{p}}\omega\left(\frac{1}{n},f\right)_p+g(n,f)\right]$$

c) Let $p\in(1,+\infty),\ a\in(0,\ 1)$ and $ap\in]0,1[$ than for the function $f\in C(T)$ the following inequality is true

$$\left\|\sigma_n^{-a}(f)-f\right\|_c\le A(p,a)n^{\frac{1}{p}}\omega\left(\frac{1}{n},f\right)_p+A(a)g(n,f);.$$

Proof.

Conclude that o $\frac{2}{\pi}\int_0^{\pi}K_n^{-a}(t)dt=1$. Therefore, we find

$$\sigma_n^{-a}(x,f)-f(x)=\frac{1}{\pi}\int_0^{\pi}\varphi(x,t)K_n^{-a}(t)dt$$

where $\varphi(x,f)=f(x+t)+f(x-t)-2f(x)$.

we will have

$$\sigma_n^{-a}(x,f)-f(x)=\frac{1}{\pi}\int_0^{\pi}\varphi(x,t)K_n^{-a}(t)dt+\frac{1}{\pi}\int_{\frac{\pi}{n}}^{\pi}\varphi(x,t)\varphi_n^{-a}(t)dt+$$

$$+\frac{1}{\pi}\int_{\frac{\pi}{n}}^{\pi}\varphi(x,t)r_n^{-a}(t)dt\equiv U_n^{(1)}(x,f,a)+U_n^{(2)}(x,f,a)$$

From which, we conclude

$$\left\|U_n^{(1)}(.,f)\right\|_c\le A(a)\omega\left(\frac{1}{n},f\right)_c .$$

Further, we will have

$$\left\|U_n^{(3)}(.,f)\right\|_c\le A(a)g(n,f) \text{ as}$$

$$\sin\left[\left(n+\frac{1}{2}-\frac{a}{2}\right)t-\frac{a\pi}{2}\right]=$$

$$=\sin\left(n+\frac{1}{2}-\frac{a}{2}\right)t\cos\frac{a\pi}{2}-\cos\left(n+\frac{1}{2}-\frac{a}{2}\right)t\sin\frac{a\pi}{2}$$

$$\sin\left(n+\frac{1}{2}-\frac{a}{2}\right)t=\sin nt\cos\left(\frac{1}{2}-\frac{a}{2}\right)t+\cos nt\sin\left(\frac{1}{2}-\frac{a}{2}\right)t,$$

$$\cos\left(n+\frac{1}{2}-\frac{a}{2}\right)t=\cos nt\cos\left(\frac{1}{2}-\frac{a}{2}\right)t-\sin nt\sin\left(\frac{1}{2}-\frac{a}{2}\right)t$$

Then that the evaluation of the expression $U_n^{(2)}(x,f,a)$ it is sufficient to estimate

$$U_n^{(4)}(x,f,a)=n^a\int\limits_{\frac{\pi}{n}}^{\pi}\varphi(x,t)\frac{\cos\left(\frac{1}{2}-\frac{a}{2}\right)t}{\left(\sin\frac{t}{2}\right)^{1-a}}\sin ntdt$$

$$U_n^{(5)}(x,f,a)=n^a\int\limits_{\frac{\pi}{n}}^{\pi}\varphi(x,t)\frac{\cos\left(\frac{1}{2}-\frac{a}{2}\right)t}{\left(\sin\frac{t}{2}\right)^{1-a}}\cos ntdt$$

Let's consider expression $U_n^{(4)}(x,f,a)$ and consider that

$$V_a(t)\equiv\frac{\cos\left(\frac{1}{2}-\frac{a}{2}\right)t}{\left(\sin\frac{t}{2}\right)^{1-a}}$$

If in the expression $U_n^{(4)}(x,f,a)$, we replace t with $t+\frac{\pi}{n}$ and summarize the received two values $U_n^{(4)}(x,f,a)$ we will have

$$2U_n^{(4)}(x,f,a)=$$

$$n^a\left\{\int\limits_{\frac{\pi}{n}}^{\pi}\varphi(x,t)V_a(t)\sin ntdt-\int\limits_{0}^{\pi-\frac{\pi}{n}}\varphi\left(x,t+\frac{\pi}{2}\right)V_a\left(t+\frac{\pi}{n}\right)\sin ntdt\right\}=$$

$$=n^a\left\{\int\limits_{\frac{\pi}{n}}^{\pi-\frac{\pi}{n}}\left[\varphi(x,t)-\varphi\left(x,t+\frac{\pi}{n}\right)\right]V_a(t)\sin ntdt+\right.$$

$$\left.+\int\limits_{\frac{\pi}{n}}^{\pi-\frac{\pi}{n}}\varphi\left(x,t+\frac{\pi}{n}\right)\left[V_a(t)-V_a\left(t+\frac{\pi}{n}\right)\right]\sin nt-\right.$$

$$-\int_0^{\pi-\frac{\pi}{n}}\varphi\left(x,t+\frac{\pi}{n}\right)V_a\left(t+\frac{\pi}{n}\right)\sin ntdt+\int_{\pi-\frac{\pi}{n}}^{\pi}\varphi(x,t)V_a(t)\sin ntdt\Bigg\}\equiv$$

$$\sum_{i=6}^{9}U_n^{(i)}(x,f,a).$$

Using Holder's inequality, according to the equality, we find

$$\left\|U_n^{(6)}(x,f,a)\right\|_c\le n^a\left\{\int_{\frac{\pi}{n}}^{\pi-\frac{\pi}{n}}\left|\varphi(x,t)-\varphi\left(x,t+\frac{\pi}{n}\right)\right|^p dt\right\}^{\frac{1}{p}}\times$$

$$\times\left\{\int_{\frac{\pi}{n}}^{\pi-\frac{\pi}{n}}|V_a(t)|^{\frac{p}{p-1}}\right\}^{\frac{p-1}{p}}$$

Accordingly, give the right to conclude that,

$$\left\|U_n^{(7)}(x,f,a)\right\|_c\le A(a)\omega\left(\frac{1}{n},f\right)_c.$$

If we take the scheme of estimation of the expression $U_n^{(7)}(x,f,a)$.

We find that

$$\left\|U_n^{(i)}(x,f,a)\right\|_c\le A(a)\omega\left(\frac{1}{n},f\right)_c\qquad(i=13,14)$$

so, we find

$$\left\|U_n^{(4)}(x,f,a)\right\|_c\le A(p,a)n^a\omega\left(\frac{1}{n},f\right)_p,\quad ap>1$$

$$\left\| U_n^{(4)}(x,f,a) \right\|_c \le A(p) n^{\frac{1}{p}} (\ln n)^{1-1/p} \omega\left(\frac{1}{n}, f\right)_p, \quad ap = 1$$

$$\left\| U_n^{(4)}(x,f,a) \right\|_c \le A(p,a) n \frac{1}{p} \omega\left(\frac{1}{n}, f\right)_p, \quad ap < 1.$$

Accordingly, relations prove the truth of the theorem 1.
The following is also true

References

1. Bakuridze M. Some properties Cesaro Means of Trigonometric Fourier Series and its Conjugates. INTERNATIONAL SCIENTIFIC JOURNAL 1(36), TBILISI 2010.
2. Bakuridze M. Some properties of Fourier Trigonometric Series of Even and Odd Functions. INTERNATIONAL SCIENTIFIC JOURNAL 3(41), TBILISI 2011.
3. Bakuridze M. On Fourier Trigonometric Series. Abstract. II International Conference. September 15-19, 2011, Batumi, Georgia.
4. Bakuridze M. Didmanidze M. Fomina T.. Classical theorem on almost everywhere convergence of Fourier transform on the set. Materials II-practical international scientific Internet Conference. Doneck, 2013.

БИНОМИАЛЬНЫЕ СИСТЕМЫ СЧИСЛЕНИЯ С ДВОИЧНЫМ АЛФАВИТОМ

А.А. Борисенко, И.А. Кулик, А.В. Иванчук, Е.М. Скордина, С.М. Мальченков
Сумский государственный университет, Украина
electron@sumdu.edu.ua

Введение

В данной работе рассматриваются основные теоретические положения построения двоичных биномиальных систем счисления, основанные на [1]. Практика биномиального счета, как это часто бывает, значительно опередила адекватную ему теорию. На сегодня получено ряд оригинальных биномиальных устройств и программ, например, [2,3], но отсутствует законченная их теория. Этот пробел

в теории биномиальных систем счисления частично восполнен в [4 - 7]. В данной работе эта теория получает дальнейшее развитие.

Двоичная биномиальная система счисления

Определение 1. Двоичной k - биномиальной системой счисления называется числовая функция

$$F = x_{r-1}C_{n-1}^{k-q_{r-1}} + \ldots + x_i C_{n-r+i}^{k-q_i} + \ldots + x_1 C_{n-r+1}^{k-q_1} + x_0 C_{n-r}^{k-q_0} \quad (1)$$

с системами ограничений

$$\begin{cases} k \le n \le n-1, \\ q = k \end{cases} \quad (2,3)$$

и

$$\begin{cases} n-k = r-q, \\ 0 \le q \le k-1, \end{cases} \quad (4,5)$$

где r - количество разрядов биномиального числа (длина), $r \in 1, 2, \ldots$;

k - максимальное число единиц, входящее в биномиальное число;

i - порядковый номер разряда, $i = 0, 1, \ldots, r - 1$;

x_i - биномиальная двоичная цифра (0 или 1);

n - целочисленный параметр системы счисления;

q - число единиц в биномиальном числе;

q_i - сумма единичных значений цифр x_i от (r -1)-го разряда до (i+1) - го включительно,

$$q_i = \sum_{j=i+1}^{r} x_j ;$$

$i = 0, 1, \ldots, r-1$; $x_r = 0$.

В качестве весового коэффициента i-го разряда в числовой функции (1) выступает биномиальный коэффициент $C_{n-r+i}^{k-q_i}$. Он зависит как от позиции $i = 0, 1, \ldots, r - 1$ рассматриваемого разряда, так и от суммы q_i предшествующих этому разряду цифр x_i.

Определение 2. *Любая конечная последовательность двоичных цифр x_{r-1} ..., x_1, x_0, удовлетворяющая ограничениям* (2,3) *или* (4,5), *называется биномиальным числом.*

Основные свойства двоичных биномиальных чисел

Приведенные ниже леммы и теоремы приводятся без доказательств. С некоторыми из них можно будет познакомиться в работах [4-7].

Лемма 1. *Количество нулей в биномиальных числах, задаваемых ограничениями* (2):

$$l = r - k.$$

Теорема 1. *Количество возможных нулей, содержащихся в биномиальных числах, образуемых ограничениями* (3):

$$l = 0, 1, ..., n - k - 1,$$

а количество возможных единиц, образуемых ограничениями (4, 5):

$$q = 0, 1, ..., k - 1.$$

Теорема 2. *Количество нулей в биномиальных числах, задаваемых ограничениями* (4, 5):

$$l = n - k.$$

Теорема 3. *Максимальная длина биномиальных чисел, задаваемых ограничениями (*4, 5):

$$r_{max} = n - 1,$$

а минимальная −

$$r_{min} = n - k.$$

Теорема 4. *Количество различных длин биномиальных чисел, задаваемых ограничениями* (2, 3):

$$d' = n - k,$$

а ограничениями (4, 5):

$$d'' = k.$$

Следствие 1. Количество различных длин всех биномиальных чисел, определяемых ограничениями (2, 3) и (4, 5):

$$d = d' + d'' = (n - k) + k = n.$$

Теорема 5. *Минимальная величина максимального числа единиц k в биномиальном числе*:

$$k_{min} = 1,$$

а максимальная

$$k_{max} = n - 1.$$

Теорема 6. *Ограничения* (2, 3) *и* (4, 5) *разбивают биномиальные числа на два непересекающихся класса, первый из которых содержит k единиц и* $0 \le l \le n - k - 1$ *нулей, а второй -* $l = n - k$ *нулей и* $0 \le q \le k - 1$ *единиц.*

Следствие 1. Биномиальные числа первого класса заканчиваются 1, а второго - 0, так как при появлении *k*-й единицы или (*n*-*k*)-го нуля необходимый признак формирования биномиального числа получен.

Лемма 2 *Количество биномиальных чисел первого класса с фиксированным числом нулей l,* $0 \le l \le n - k - 1$:

$$N_l = C^{l}_{k+l-1}.$$

Следствие 1. При $l = l_{max} = n - k - 1$

$$N_l = N_{l_{\max}} = C^{l_{\max}}_{k+l_{\max}-1} = C^{n-k-1}_{n-2} = C^{k-1}_{n-2}.$$

Следствие 2. При $l = l_{min} = 0$

$$N_l = N_{l\min} = C_{k-1}^0 = 1.$$

Теорема 7. *Количество биномиальных чисел первого класса*

$$N_k = C_{n-1}^k.$$

Лемма 3 *Количество биномиальных чисел второго класса с фиксированным числом единиц q, $0 \le q \le k$ - 1:*

$$N_q = C_{n-k+q-1}^q.$$

Следствие 1. При $q = q_{max} = k$ - 1

$$N_q = N_{q\max} = C_{n-k+q\max-1}^{q\max} = C_{n-k+k-1-1}^{k-1} = C_{n-2}^{k-1}.$$

Следствие 2. При $q = q_{min} = 0$

$$N_q = N_{q\min} = C_{n-k-1}^0 = 1.$$

Теорема 8. *Количество биномиальных чисел второго класса*

$$N_{n-k} = C_{n-1}^{k-1}.$$

Теорема 9. *Количество биномиальных чисел, задаваемых ограничениями* (2, 3) *и* (4, 5):

$$N = C_n^k.$$

Ниже в табл. 1 для $k = 4$ и $n = 6$ приведены все биномиальные числа и их количественные эквиваленты, удовлетворяющие приведенным выше теоремам и определению биномиального числа.

Таблица 1. Биномиальные числа и их количественные эквиваленты

Биномиальное число	Количественный эквивалент
00	$0C_5^4 + 0C_4^4$
010	$0C_5^4 + 1C_4^4 + 0C_3^3$
0110	$0C_5^4 + 1C_4^4 + 1C_3^3 + 0C_2^2$
01110	$0C_5^4 + 1C_4^4 + 1C_3^3 + 1C_2^2 + 0C_1^1$
01111	$0C_5^4 + 1C_4^4 + 1C_3^3 + 1C_2^2 + 1C_1^1$
100	$1C_5^4 + 0C_4^3 + 0C_3^3$
1010	$1C_5^4 + 0C_4^3 + 1C_3^3 + 0C_2^2$
10110	$1C_5^4 + 0C_4^3 + 1C_3^3 + 1C_2^2 + 0C_1^1$

10111	$1C_5^4+0C_4^3+1C_3^3+1C_2^2+1C_1^1$
1100	$1C_5^4+1C_4^3+0C_3^2+0C_2^2$
11010	$1C_5^4+0C_4^3+0C_3^2+1C_2^2+0C_1^1$
11011	$1C_5^4+1C_4^3+0C_3^2+1C_2^2+1C_1^1$
11100	$1C_5^4+1C_4^3+1C_3^2+0C_2^1+0C_1^1$
11101	$1C_5^4+1C_4^3+1C_3^2+0C_2^1+1C_1^1$
1111	$1C_5^4+1C_4^3+1C_3^2+1C_2^1$

Список используемых источников

1. Борисенко А.А. Системы счисления с биномиальным основанием и двоичным алфавитом / А.А. Борисенко – ВИНИТИ, №909-82, 1982. – 6 с.
2. Борисенко А.А. Синтез автоматов с регулярной структурой для генерирования кодов с постоянным весом / А.А. Борисенко, С.И. Губарев, Г.В. Куно, В.А. Алексеев // Автоматизированные системы управления и приборы автоматики. – 1987. – №81. – С. 101 – 104.
3. Borysenko Olexiy. Binary Image Compression Based on Binomial Numbers / Olexiy Borysenko, Igor Kulik, Sergiy Kostel, Olena Skordina // Bulletin, Mathematics-Informatics-Physics Series, Petroleum. – 2010. – Gas University of Ploiesti, № 2. – P. 1-12.
4. Борисенко А.А. Основы теории двоичного биномиального счета // Вестник СумГУ. – 1999. – №1(12). – С. 71-75.
5. Борисенко А. А. Биномиальный счет. Теория и практика: Монография / А.А. Борисенко. – Сумы: ИТД "Университетская книга", 2004. – 170 с.
6. Борисенко А.А. Введение в теорию биномиального счета: Монография / А.А. Борисенко. – Сумы, "Университетская книга", 2004. – 88 с.
7. Борисенко А.А. Биномиальное кодирование: Монография / А.А. Борисенко, И.А. Кулик. − Сумы: Изд-во СумГУ, 2010. – 206 с.

ЗАГАЛЬНА МОДЕЛЬ УСПАДКУВАННЯ

Д.Б. Буй, С.В. Компан
Київський національний університет імені Тараса Шевченка, Україна
buy@unicyb.kiev.ua, robin_2005@mail.ru

Постановка задачі

В [1] було введено поняття класу. Під класом будемо розуміти двохкомпонентну структуру $\langle s, \mu \rangle$, де s – функціональне бінарне відношення, яке атрибутам ставить у відповідність їх типи (множину значень з універсального домену D), а μ – функціональне бінарне відношення, яке сигнатурам методів ставить у відповідність їх реалізацію (семантику). Виконаємо формальне уточнення успадкування класів, а також побудову батьківського класу по двом заданим класам. При уточненні успадкування та відновленні батьківського класу модифікатори доступу ігноруємо.

Побудова батьківського класу по двом заданим класам

Нехай маємо функціональне бінарне відношення γ, яке відповідає відношенням s або μ батьківського класу. Відповідно функціональні бінарні відношення α, β, які відповідають першим або другим компонентам похідних класів.

Нехай $X = dom\alpha \bigcap dom\beta$ множина спільних імен класів. Очевидно, що $X = X' \bigcup X''$, де $X' = \{x | x \in X, \alpha(x) = \beta(x)\}$; $X'' = \{x | x \in X, \alpha(x) \neq \beta(x)\}$.

Очевидно, що $X' \bigcap X'' = \varnothing$; не важко перевірити, що виконується еквівалентність $\alpha \approx \beta \Leftrightarrow X'' = \varnothing$, де $\approx$ – відношення сумісності ($\alpha \approx \beta \Leftrightarrow \alpha|X = \beta|X$, де, в свою чергу, наприклад, $\alpha|X$ – обмеження відношення α за множиною X, тобто, $\alpha|X = \alpha \bigcap X \times \pi_2^2 \alpha$ [2]).

Розглянемо три способи побудови відношення γ, яке відповідає, як і раніше, батьківському класу.

1) $\gamma = \alpha \bigcap \beta$. Можна перевірити, що тоді $\gamma = \alpha|X' = \beta|X'$, $dom\gamma = X'$. Відношення α та β відновлюються по відношенню

γ в такий спосіб: $\alpha = \gamma \bigcup (\alpha \big| X'') \bigcup \alpha \big| (dom\alpha \setminus dom\beta)$,

$\beta = \gamma \bigcup (\beta \big| X'') \bigcup \beta \big| (dom\beta \setminus dom\alpha)$.

Дамо змістовну інтерпретацію. Очевидно, що відношення α та β повністю симетричні і їх відновлення по відношенню γ, що відіграє роль батьківського класу, відбувається за схемою розширення (рис. 1).

Розглянемо вищесказане на прикладі. Нехай маємо два класи (мова програмування Java v.8 [3]):

```
class A {                          class B {
int a; int b; float p;             int a; float b; String s;
float g(){return a+b;}             float g(){return a+b;}
}                                  }
```

Тоді батьківський клас C можна побудувати так (рис.1):

```
                  class C {
                  int a; float b;
                  float g(){return a+b;}}

class A extends C {                class B extends C {
float p;                           String s;
}                                  }
```

Рис. 1. Побудова батьківського класу при умові $\gamma = \alpha \bigcap \beta$

Зазначимо, що коли $\gamma = \alpha \bigcap \beta = \varnothing$ батьківський клас будується, але він порожній. Покажемо на прикладі:

Нехай маємо два класи (мова програмування Java v.8 [3]):

```
class A {                          class B {
int a; int b; float p;             float a; float b; String s;
int g(){return a*b;}               float g(){return a+b;}
}                                  }
```

Очевидно, над класами A і B виконується умова $\gamma = \alpha \bigcap \beta = \varnothing$, тому можна побудувати порожній батьківський клас (рис. 2):

```
                  class C {  }
class A extends C {                class B extends C {
int a; int b; float p;             float a; float b; String s;
int g(){return a*b;}               float g(){return a+b;}
}                                  }
```

Рис. 2. Побудова батьківського класу при умові $\gamma = \alpha \bigcap \beta = \varnothing$

2) Цей спосіб побудови батьківського класу відповідає ситуації, коли відношення α (для визначенності "ліве") має пріоритет. Уточнення таке: $\gamma = \alpha_l \nabla \beta = \alpha \big| X$, тобто, змістовно кажучи, на спільних іменах пріоритет має відношення α.

Відношення α та β відновлюються по відношенню γ так:

$\alpha = \gamma \cup \alpha \big| (dom\alpha \setminus dom\beta)$,

$\beta = \gamma \big| X' \cup \beta \big| X'' \cup \beta \big| (dom\beta \setminus dom\alpha)$.

Змістовна інтерпретація: α відновлюється за схемою поповнення множини імен відношення $\gamma(X)$ іменами множини $dom\alpha \setminus dom\beta$; таким чином, мова йде про схему розширення. Що стосується відношення β, то тут ситуація більш складна:

а) на множині X' відношення γ і β співпадають;

б) на множині X'' відношення β перевизначається;

в) β розширюється на множину ("нових") імен $dom\beta \setminus dom\alpha$.

3) Відношення α та β просто міняються ролями, тому формальні вирази запишемо стисло: $\gamma = \alpha \nabla_r \beta = \beta \big| X$ (на спільних іменах пріоритет має відношення β, "праве"); відношення α та β відновлюються за рівностями: $\beta = \gamma \cup \beta \big| (dom\beta \setminus dom\alpha)$, $\alpha = \gamma \big| X' \cup \alpha \big| X'' \cup \alpha \big| (dom\alpha \setminus dom\beta)$ (тут вже α перевизначається на множині імен X'').

Розглянемо вищесказане на прикладі. Нехай маємо два класи (мова програмування Java v.8 [3]):

```
class A {
int a; int b;
int g(){return a+b;}
}
```

```
class B {
int a; float b; String s;
int g(){return (int) (a*b);}
}
```

Тоді, батьківський клас C та класи A, B, які будуть похідними від нього, будуються так (клас B – пріоритет) (рис. 3):

```
                    class C {
                    int a; float b;
                    int g(){return (int) (a*b);}
                    }
    class A extends C {              class B  extends C {
    int b;                           String s;
    int g(){return a+b;}             }
}
```

Рис. 3. Побудова батьківського класу з пріоритетом

З рис. 3 видно, що ми відновили батьківський клас по двом заданим з врахуванням правостороннього пріоритету.

Висновки

В роботі розглянута та уточнена загальна схема успадкування за схемами розширення та пере визначення. Також наведені три способи побудови батьківського класу по двом заданим класам: батьківський клас будується перетином вихідних класів, батьківський клас будується на основі одного («лівого» або «правого») вихідного класу.

Список використаних джерел

1. Буй Д.Б. Диаграммы классов ООП: формализация и анализ / Д.Б. Буй, С.В. Компан // Труды ИПММ НАН Украины. – 2013. – Том 27. – С. 51-65.
2. Буй Д.Б. Властивості теоретико-множинних конструкцій повного образу та обмеження / Д.Б. Буй, Н.Д. Кахута // Вісник Київського університету імені Тараса Шевченка. Сер.: фіз.-мат. науки. – 2005. – Вип. 2. – С. 232-240.
3. Мова програмування Java. [електронний ресурс]: точка входу http://www.java.com

ОГЛЯД ТИПІВ РЕКОМЕНДАЦІЙНИХ СИСТЕМ ТА ВИКОРИСТАННЯ БАЗ ДАНИХ ДЛЯ ПІДВИЩЕННЯ ЇХ ПРОДУКТИВНОСТІ

Д.Б. Буй, С.В. Компан, К.Д. Мохамед, С.А. Поляков
Київський національний университет імені Тараса Шевченка
buy@unicyb.kiev.ua

Вступ

В роботі розглядаються деякі питання, пов’язані з побудовою рекомендаційних систем [1, 2]. В останнє десятиліття розвиток електронної комерції, систем Інтернет-новин, відео та музичних онлайн-сервісів, електронних журналів, соціальних мереж та блогів зробив вкрай актуальним питання пошуку та фільтрації інформації. Користувачі мають доступ до великих обсягів інформації, але не мають ні часу, ні бажання читати всю потенційно корисну інформацію. Наприклад, в звичайному книжковому магазині покупець може вибирати одну книгу з кількох сотень або тисяч. В Інтернет-магазині таких книг може бути десятки тисяч. Більш конкурентноспроможними стануть ті Інтернет-магазини, що спрогнозують, які саме товари або сервіси зацікавлять потенційного покупця. В невеличких магазинах з постійною та стабільною клієнтурою продавець добро знає вкуси своїх покупців та надає їм саме ті товари, які їм потрібні. В Інтернет-магазинах з мільйонами відвідувачів роль такого продавця-консультанта виконує рекомендаційна система.

Рекомендаційні системи, як правило, мають справу з великими обсягами даних, які потребують для своєї обробки багато часу. Це неприйнятно для програм, працюючих в режимі реального часу. Тому важливе значення набувають питання, пов’язані з оптимізацією, зокрема вибору сховища даних. В розпорядженні програмістів є багато типів СУБД, включаючи реляційні та постреляційні, такі як сховища «ключ-значення», документно-орієнтовані сховища, сховища на базі колонок та графові сховища. Крім того можна використовувати спеціалізовану файлову систему, як це зроблено, наприклад, в Hadoop. Отже, питання вибору «правильної» СУБД має критичне значення для побудови конкурентноспроможної рекомендаційної системи.

Типи рекомендаційних систем

Рекомендаційні системи є підкласом систем фільтрації інформації, що прагнуть передбачити перевагу, яку користувач буде віддавати на користь одного об'єкта над іншими. Іншими словами, вони займаються прогнозуванням вибору, який робить

користувач з декількох варіантів. Наведемо приклади поширених типів рекомендаційних систем.

1. Пропозиція новин для читачів онлайн-газети, на основі прогнозу читацьких інтересів.

2. Пропозиція клієнтам товарів в Інтернет-магазині, які вони хотіли б купити, залежно від історії їх покупок і/або продукту.

3. Пропозиція нових фільмів для перегляду в онлайн відеосервісі, в залежності від історії перегляду фільмів.

В рекомендаційних системах існує два класи сутностей – користувачі та об'єкти. Користувачі надають перевагу одним об'єктам перед іншими, що можна зрозуміти з аналізу даних про їх вподобання. Дані представляються у вигляді спеціальної матриці переваг (utility matrix). Вона надає для кожної пари (користувач, об'єкт) значення, яке показує наскільки користувач віддає перевагу цьому об'єкту перед іншими. Таке значення часто називають рейтингом або оцінкою. Матриця переваг розряджена, тобто більшість клітинок є порожніми. Порожні клітинки з'являються у тих випадках, коли немає достатньої інформації, наскільки користувачу подобається об'єкт, тобто це значення невідоме. В якості приклада розглянемо матрицю переваг для фільмів. Стовпці відповідають фільмам а рядки – користувачам.

Рейтинги фільмів позначаються числами від 1 до 5. Число 1 відповідає найнижчому рейтингу, а число 5 – найвищому. Порожня клітинка означає, що користувач не поставив рейтинг для цього фільму. Як правило, більшість клітинок буде порожньою. Користувачів позначимо буквами A, B, C, D.

Таблиця 1. Матриця рейтингу фільмів

	«Зоряні війни»	«Матриця»	«Планета мавп»	«Хороший поганий, злий»	«Джанго звільнений»	«Чудова сімка»
A	4	5		1		2
B	5		5			
C					5	
D		1			2	

Метою рекомендаційної системи є прогнозування значень для порожніх клітинок. Для цього застосовуються два базових метода:

1) *контентні системи*, які шукають подібні об'єкти, використовуючи подібність властивостей об'єктів;

2) *колаборативні системи* (collaborative-filtering), що зосереджуються на відносинах між користувачами і об'єктами. Подібність користувачив визначається схожістю оцінок однакових предметів.

В контентних системах для кожного об’єкту визначається сукупність атрибутів, які містять його властивості. Наприклад, для фільму це можуть бути актори, які грають у фільмі, режисер фільму, назва фільму, рік випуску, жанр фільму та рейтинг, поставлений користувачем. Значення атрибутів записуються у вигляді вектору. Нехай, наприклад, властивості фільму складаються тільки з акторів та рейтингу, причому загальна кількість акторів, які можуть бути у всіх фільмах дорівнює n. Тоді фільми задаються вектором довжини n+1, в якому кожному актору відповідає окремий компонент, і ще один компонент, що містить рейтинг. Якщо актор приймає участь у фільмі, то відповідний компонент містить 1, в противному випадку – 0. В компоненті, яка відповідає рейтингу, міститься середній рейтинг фільму або значення 0, якщо користувачі не оцінювали фільм. Два таких вектора можна порівняти за допомогою різних мір, зокрема, косинусної відстані векторів (cosine distance of vectors).

Крім вектору властивостей фільмів створюється вектор переваг користувачів. Цей вектор розраховується на базі векторів фільмів. Наприклад, якщо матриця переваг містить 10 фільмів, яким користувач поставив рейтинги, і актор Марк Гемілл («Зоряні війни») зустрічається у 4 фільмах, то в векторі переваг користувача відповідна компонента отримає значення 0.25. Далі вектор переваг порівнюється з вектором деякого фільму, який користувач ще не дивився. Якщо вектори будуть подібні один до одного, то система надасть користувачу рекомендацію подивитися цей фільм.

Системи на базі коллаборативної фільтрації шукають подібність не предметів, а користувачів. Кожний користувач в таких системах визначається не профілем переваг, а рядком з матриці переваг, який містить введені їм рейтинги, або одиниці та нулі, в залежності від того, дивився він фільм чи ні. Розрахунок подібності користувачів по рядкам в матриці корисності роблять таким самим чином, як і для контентних систем, а саме, використовують косинусну векторну відстань або Джакардову відстань (Jaccard Distance). Цей підхід може бути використаний двояким чином: або шляхом порівняння рядків для знаходження подібних користувачів, або, шляхом порівняння стовпців для знаходження подібних об’єктів.

На сьогодні стає популярним ще один метод побудови рекомендаційних систем на базі SVD (Сингулярне Розкладання Векторів). Він широко використовується в коллаборативній фільтрації. Метод заснований на розкладенні матриці оцінок, яка є розрядженою, на декілька матриць меншого розміру, що відповідають за окремі властивості об’єктів.

Техніка обробки даних MapReduce

На практиці рекомендаційні системи працюють з великими обсягами даних. Для оброблення таких обсягів даних за розумний час використовують спеціальну техніку, яка називається MapReduce [3, 4, 5]. Вперше вона була використана Google, саме ця компанія дала назву цій техніці. На сьогодні існує кілька програмних систем, які втілили цю техніку, найбільш відомою з яких є Hadoop від компанії-розробника Apache. Крім того такі СУБД як MongoDB [6] та CouchDB теж втілили цей метод. Основна ідея полягає в тому, щоб розподілити роботу між великою кількістю комп'ютерів, які утворюють кластер. Процес обчислення складається з двох кроків – Map та Reduce.

На першому кроці задача розділяється на підзадачи, кожна з яких задається у вигляді пари «ключ-значення». Цей поділ потрібно зробити таким чином, щоб підзадачи могли виконуватися незалежно одна від одної. Це розподілення залежить від задачі і виконується програмою, написаною спеціально для конкретної задачі. Далі ці підзадачі розподіляються між окремими комп'ютерами кластеру і йде процес їхнього розв'язання.

На другому кроці, який називається Reduce, результати виконаних задач групуються по значенню їхніх ключів. Над кожною групою виконується операція агрегації, яка задається окремою програмою. Результати згортки груп об'єднуються і утворюють розв'язок задачі.

Прикладом задачі, яка розв'язується за допомогою MapReduce, є програма, що підраховує для кожного слова кількість його входження в сукупність документів. На першому кроці документи розбиваються на окремі слова і повертаються пари, де ключем виступає слово, а значенням є число 1. На другому етапі однакові слова групуються, а в якості операції Reduce використовується функція, яка підсумовує кількість однакових слів.

Якщо функція Reduce є асоціативною та комутативною, то порядок виконання операцій згортки стає неважливим. В такому випадку частина Reduce-операцій може бути виконаною в середині Map-функцій. Наприклад, всі однакові слова, які знаходяться в одному документі, підраховуються на етапі розбиття документу на окремі слова. А на Reduce-кроці додаються вже однакові слова, що знаходяться в різних документах.

Застосування СУБД у рекомендаційних системах

Як було вже сказано, рекомендаційні системи працюють з великими обсягами даних. Тому для них важливе питання підвищення продуктивності. Один з шляхів полягає в переході на сховища даних типу NoSQl (NotOnlySQL). Розглянемо це на прикладі системи Рамблер-новини [7].

Рекомендаційний сервіс проекту Рамблер-новини ґрунтується на об'єднанні користувачів в групи за подібністю інтересів і обчисленні найбільш популярних новин серед груп в заданому проміжку часу.

Для обчислення новинних рекомендацій спочатку було прийнято рішення проводити обробку новин за останні 5 днів. Середній обсяг даних за вказаний період становить приблизно 7 ГБ. Для реалізації алгоритму був обраний фреймворк Hadoop, що є де-факто стандартом для потокової обробки великих обсягів даних. Але час обробки та надання рекомендацій виявився досить великим. Тому було прийняте рішення провести експеримент по використанню СУБД. Враховуючи обсяги даних, реляційні СУБД не дозволяли досягнути потрібної швидкості, тому вибір йшов серед СУБД типу NoSQL.

Більшість сучасних NoSQL-рішень реалізують парадигму обчислень MapReduce. Це дає можливість поряд з фундаментальною властивістю горизонтального масштабування переносити алгоритми, призначені для фреймворків типу Hadoop, на сховища NoSQL та отримувати всі додаткові переваги, в першу чергу такі як висока швидкість обробки даних.

Серед документно-орієнтованих баз даних вибір припав на проект MongoDB. Ця СУБД працює з документами, представленими у форматі JSON (JavaScript Object Notation). Для роботи з документами використовується мова JavaScript та Application Programming Interface для доступу з інших мов програмування. Важливою перевагою на користь даної системи була підтримка посилань на інші документи. Крім того дані зберігаються в бінарному форматі, що оптимізує трафік між клієнтом і сховищем.

В результаті проведеного експерименту з’ясувалося, наприклад, що одна з задач, а саме завдання підрахунку мінімального хешу після запису однієї тисячі нових записів, завершувалось через 400-500 мілісекунд, замість 3 попередніх секунд, що надало можливість робити це в режимі реального часу. Крім того інформація, що оброблена в попередній час, накопичується в базі даних, тому і її не потрібно було заново перераховувати.

Висновки

Таким чином, застосування NoSQL-підходу до розв'язання задач подібного класу підвищує ефективність систем, що, в свою чергу, дозволяє обробити більшу кількість рекомендаційних даних за той же самий час та підвищити якість прогнозування.

Список використаних джерел

1. Rajaraman A.. Mining of Massive Datasets / Anand Rajaraman, Jeffrey David Ullman // Cambridge University Press, New York, 2011. – 503 p.
2. Adomavicius G.. "Towards the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," / G. Adomavicius, A. Tuzhilin // IEEE Trans. on Data and Knowledge Engineering 17:6, 2005. – p. 734–749.
3. Dean J. MapReduce: Simplified data processing on large clusters / J. Dean , S. Ghemawat // OSDI'04: 6th Symposium on Operating System Design and Implementation Proceedings. – Berkeley, CA, USA: USENIX Association, 2004. – p. 137–149.
4. Ghemawat S. "The Google file system," / S. Ghemawat, H. Gobioff, S.-T. Leung // 19th ACM Symposium on Operating Systems Principles, 2003.
5. hadoop.apache.org, Apache Foundation
6. Chodorow K. MongoDB: The Definitive Guide/ K. Chodorow // O'Relly Media Inc., 2013. – 300 p.
7. Клеменков П. А. Построение новостного рекомендательного сервиса реального времени с использованием NoSQL СУБД / П. А. Клеменков // Информатика и ее применение. – 2013. – том 7, выпуск 3. – С. 14–21.

АКСІОМАТИКА БАГАТОЗНАЧНИХ ЗАЛЕЖНОСТЕЙ ТАБЛИЧНИХ БАЗ ДАНИХ: ПОВНОТА ТА ЇЇ КРИТЕРІЙ

Д.Б. Буй, А.В. Пузікова
Київський національний університет імені Тараса Шевченка, Україна
buy@unicyb.kiev.ua, anna_inf@mail.ru

Аксіоматика багатозначних залежностей

Всі неозначувані тут поняття та позначення використовуються в розумінні монографії [1], зокрема, $s \mid X$ – обмеження рядка s за множиною атрибутів X.

Нехай t – таблиця, R – схема таблиці t (скінченна множина атрибутів), X, Y, W, Z – підмножини схеми R, s, s_1, s_2 – рядки таблиці t. В подальшому розгляді множина R зафіксована, крім того, фіксується універсальний домен D – множина, з якої атрибути приймають значення в інтерпретаціях.

Скажемо, що на таблиці t схеми R виконується багатозначна залежність (БЗЗ) (див., наприклад, [1]) $X \to\to Y$, якщо для двох довільних рядків s_1, s_2 таблиці t, які збігаються на множині атрибутів X, існує рядок $s_3 \in t$, який дорівнює об'єднанню обмежень рядків s_1, s_2 на множини атрибутів $X \bigcup Y$ і $R \setminus (X \bigcup Y)$ відповідно:

$$(X \to\to Y)(t) = true \overset{def}{\Leftrightarrow} \forall s_1, s_2 \in t(s_1 \mid X = s_2 \mid X \Rightarrow$$
$$\Rightarrow \exists s_3 \in t(s_3 = s_1 \mid (X \bigcup Y) \bigcup s_2 \mid R \setminus (X \bigcup Y))).$$

Отже, з семантичної точки зору БЗЗ – це предикат, заданий трьома (скінченними) множинами атрибутів X, Y і R.

Скажемо, що рядки s_1, s_2 таблиці t знаходяться у відношенні $=_X$, якщо вони збігаються на множині атрибутів X:

$$s_1 =_X s_2 \overset{def}{\Leftrightarrow} s_1 \mid X = s_2 \mid X.$$

Відношення $=_X$ розбиває множину рядків s таблиці t на класи еквівалентності, які мають наступне зображення:

$$[s]_{=_X} = \{s \mid X\} \otimes \pi_Y [s]_{=_X} \otimes \pi_{R \setminus (X \cup Y)} [s]_{=_X}.$$

Скажемо, що таблиця $t(R)$ є *моделлю* множини БЗЗ G, якщо кожна БЗЗ $X \to\to Y \in G$ виконується на таблиці $t(R)$:

$$t(R) \text{ модель } G \overset{def}{\Leftrightarrow}$$

$$\overset{def}{\Leftrightarrow} \forall (X \to\to Y)(X \to\to Y \in G \Rightarrow (X \to\to Y)(t) = true).$$

Вище запис $t(R)$ означає, що таблиця t має схему R.

Скажемо, що БЗЗ $X \to\to Y$ *семантично слідує* з множини БЗЗ G, якщо на кожній таблиці $t(R)$, яка є моделлю множини БЗЗ G, виконується також БЗЗ $X \to\to Y$:

$$G \models X \to\to Y \overset{def}{\Leftrightarrow} \forall t(R)(t \text{ модель } G \Rightarrow$$
$$\Rightarrow (X \to\to Y)(t) = true).$$

Лема 1 (*аксіома рефлексивності*). $\forall t(X \to\to Y)(t) = true$, де $Y \subseteq X$. □

Наслідок 1. $\varnothing \models X \to\to Y$, для $Y \subseteq X$. □

Лема 2. $\forall t(X \to\to Y)(t) = true$, де $X \bigcup Y = R$. □

Наслідок 2. $\varnothing \models X \to\to Y$, для $X \bigcup Y = R$. □

БЗЗ вигляду $X \to\to Y$, де $Y \subseteq X$ або $X \bigcup Y = R$, називається *тривіальною*.

Лема 3 (*правило повноти*).

$(X \to\to Y)(t) = true \Rightarrow (X \to\to R \setminus (X \bigcup Y))(t) = true$. □

Наслідок 3. $G \models X \to\to Y \Rightarrow G \models X \to\to R \setminus (X \bigcup Y)$. □

Лема 4 (правило *поповнення*). $(X \to\to Y)(t) = true \& Z \subseteq W \Rightarrow$

$\Rightarrow (X \bigcup W \to\to Y \bigcup Z)(t) = true$. □

Наслідок 4.

$G \models X \to\to Y \ \& Z \subseteq W \Rightarrow G \models X \bigcup W \to\to Y \bigcup Z$. □

Лема 5 (правило *транзитивності*). $(X \to\to Y)(t) = true \ \&$

$\& (Y \to\to Z)(t) = true \Rightarrow (X \to\to Z \setminus Y)(t) = true$. □

Наслідок 5.

$G \models X \to\to Y \ \& \ G \models Y \to\to Z \Rightarrow G \models X \to\to Z \setminus Y$. □

Наслідок 6. $G \vDash X \to\to Y \ \& \ G \vDash Y \to\to Z \ \& Z \bigcap Y = \varnothing \Rightarrow$

$\Rightarrow G \vDash X \to\to Z$. □

Аксіома рефлексивності і правила повноти, поповнення та транзитивності наведені у [2]. Ці правила назвемо *правилами виведення для БЗЗ*, бо вони мають синтаксичну природу.

Скажемо, що БЗЗ $X \to\to Y$ *синтаксично слідує* з множини БЗЗ G відносно схеми R ($G \vdash_R X \to\to Y$), якщо існує скінченна послідовність БЗЗ $\varphi_1, \varphi_2, ..., \varphi_{m-1}, \varphi_m$, така, що $\varphi_m = X \to\to Y$ і для $\forall i = \overline{1, m-1}$ кожна φ_i є або аксіома рефлексивності, або належить G, або отримана за яким-небудь правилом виведення для БЗЗ з попередніх у цій послідовності БЗЗ φ_j, φ_k, $j, k < i$. Послідовність $\varphi_1, \varphi_2, ..., \varphi_{m-1}, \varphi_m$ назвемо доведенням, наслідуючи традиції математичної логіки [3].

Нехай задана деяка множина БЗЗ G. *Замикання* $[G]_R$ – це множина БЗЗ, які синтаксично слідують з G відносно схеми R:

$$[G]_R \overset{def}{=} \{X \to\to Y \mid G \vdash_R X \to\to Y\}.$$

Лема 6. Виконуються властивості:

1) $G \subseteq [G]$ (*зростання*);

2) $[[G]] = [G]$ (*ідемпотентність*);

3) $G \subseteq H \Rightarrow [G] \subseteq [H]$ (*монотонність*).□

Таким чином, за термінологією [4] оператор $G \mapsto [G]$ є оператором замикання.

З описаних вище аксіоми і правил виведення можна отримати інші (похідні) правила виведення для БЗЗ.

Лема 7 (правило *псевдотранзитивності*).

$\{X \to\to Y, Y \bigcup W \to\to Z\} \vdash X \bigcup W \to\to Z \setminus (Y \bigcup W)$. □

Лема 8 (правила різниці).

1. $\{X \to\to Y\} \vdash X \to\to Y \setminus X$;
2. $\{X \to\to Y \setminus X\} \vdash X \to\to Y$;
3. $\{X \to\to Y\} \vdash X \to\to R \setminus Y$. □

Лема 9 (правило *адитивності*).

$\{X \to\to Y_1, X \to\to Y_2\} \vdash X \to\to Y_1 \bigcup Y_2$.□

Наслідок 7. Для $n = 2,3,...$ має місце твердження: $\{X \to\to Y_1,...,X \to\to Y_n\} \vdash X \to\to Y_1 \bigcup ... \bigcup Y_n$. □

Лема 10 (правила *декомпозиції*).

1. $\{X \to\to Y_1, X \to\to Y_2\} \vdash X \to\to Y_1 \bigcap Y_2$;
2. $\{X \to\to Y_1, X \to\to Y_2\} \vdash X \to\to Y_1 \setminus Y_2$.□

Наслідок 8. Для $n = 2,3,...$ має місце твердження: $\{X \to\to Y_1,...,X \to\to Y_n\} \vdash X \to\to Y_1 \bigcap ... \bigcap Y_n$. □

Правила псевдотранзитивності, адитивності та декомпозиції наведені у роботах [2, 5].

Аксіоматика багатозначних і функціональних залежностей

Нагадаймо, що на таблиці t виконується функціональна залежність (ФЗ) (див., наприклад, [1]) $X \to Y$, якщо для двох довільних рядків s_1, s_2 таблиці t, які збігаються на множині атрибутів X, має місце їх рівність і на множині атрибутів Y:

$$(X \to Y)(t) = true \overset{def}{\Leftrightarrow} \forall s_1, s_2 \in t\left(s_1\middle|X = s_2\middle|X \Rightarrow s_1\middle|Y = s_2\middle|Y\right)$$

.

Нехай задані множини ФЗ F і БЗЗ G. Скажемо, що таблиця $t(R)$ є *моделлю* множини $F \bigcup G$, якщо кожна залежність $\varphi \in F \bigcup G$ виконується на таблиці t:

$$t(R) \text{ } модель \text{ } F \bigcup G \overset{def}{\Leftrightarrow} \forall \varphi(\varphi \in F \bigcup G \Rightarrow \varphi(t) = true).$$

ФЗ або БЗЗ φ *семантично слідує* з об'єднання множин ФЗ і БЗЗ $F \bigcup G$ відносно схеми R, якщо на кожній таблиці $t(R)$, яка є моделлю $F \bigcup G$, виконується також φ:

$$F \bigcup G \vDash \varphi \overset{def}{\Leftrightarrow} \forall t(R)(t(R) - модель \text{ } F \bigcup G \Rightarrow \varphi(t) = true).$$

Лема 11. Нехай H_1 і H_2 – множини залежностей (ФЗ або БЗЗ), і нехай T_1 і T_2 множини їх відповідних моделей. Тоді виконується імплікація $H_1 \subseteq H_2 \Rightarrow T_1 \supseteq T_2$. □

Наслідок 9. Виконуються імплікації:

1. $F \vDash \varphi \Rightarrow F \bigcup G \vDash \varphi$;
2. $G \vDash \varphi \Rightarrow F \bigcup G \vDash \varphi$. □

Лема 12 (правило розширення ФЗ до БЗЗ).
$(X \to Y)(t) = true \Rightarrow (X \to\to Y)(t) = true$. □

Наслідок 10. $F \vDash X \to Y \Rightarrow F \vDash X \to\to Y$. □

Лема 13 (спільне правило для ФЗ та БЗЗ). $(X \to\to Z)(t) = true$ & $(Y \to Z')(t) = true$ & $Z' \subseteq Z$ & $Y \bigcap Z = \varnothing \Rightarrow$ $\Rightarrow (X \to Z')(t) = true$. □

Наслідок 11. $G \vDash X \to\to Z$ & $F \vDash Y \to Z'$ & $Z' \subseteq Z$ & & $Y \bigcap Z = \varnothing \Rightarrow F \bigcup G \vDash X \to Z'$. □

Розглянуті у лемах 12 і 13 правила наведені у [2].

Скажемо, що ФЗ або БЗЗ φ *синтаксично слідує* з об'єднання множин ФЗ і БЗЗ $F \bigcup G$ відносно схеми R ($F \bigcup G \vdash_R \varphi$), якщо існує скінченна послідовність ФЗ або БЗЗ $\varphi_1, \varphi_2, ..., \varphi_{m-1}, \varphi_m$, така, що $\varphi_m = \varphi$ і для $\forall i = \overline{1, m-1}$ кожна φ_i є або аксіома рефлексивності (для ФЗ або БЗЗ) , або належить $F \bigcup G$, або отримана за яким-небудь правилом виведення (повноти (для БЗЗ), поповнення (для ФЗ або БЗЗ), транзитивності (для ФЗ або БЗЗ), правила розширення ФЗ до БЗЗ або правила з леми 13) з попередніх у цій послідовності ФЗ або БЗЗ φ_j, φ_k, $j, k < i$. Як і раніше, послідовність $\varphi_1, \varphi_2, ..., \varphi_{m-1}, \varphi_m$ назвемо доведенням φ з об'єднання множин $F \bigcup G$.

Нехай задані деякі множини F і G.

Замикання $[F \bigcup G]_R$ – це множина усіх ФЗ і БЗЗ, які синтаксично слідують з $F \bigcup G$ відносно схеми R:

$$[F \bigcup G]_R \overset{def}{=} \{\varphi \mid F \bigcup G \vdash \varphi\}.$$

Лема 14. Виконуються властивості:

1) $F \bigcup G \subseteq [F \bigcup G]$ (*зростання*);
2) $[[F \bigcup G]] = [F \bigcup G]$ (*ідемпотентність*);
3) $F' \bigcup G' \subseteq F \bigcup G \Rightarrow [F' \bigcup G'] \subseteq [F \bigcup G]$ (*монотонність*);
4) $[F] \subseteq [F \bigcup G]$, $[G] \subseteq [F \bigcup G]$;
5) $[F] \bigcup [G] \subseteq [F \bigcup G]$. □

З властивостей 1-3 леми 14 за термінологією [4] оператор $F \bigcup G \mapsto [F \bigcup G]_R$ є оператором замикання.

В наступній лемі вкажемо ще одне спільне правило для БЗЗ і ФЗ.

Лема 15. $\{X \to\to Y, X \bigcup Y \to Z\} \vdash X \to Z \setminus Y$. □

Замиканням $[X]_{F \bigcup G, R}$ множини X (відносно множини $F \bigcup G$ і схеми R) називається сім'я усіх множин правих частин БЗЗ, які синтаксично слідують з множини $F \bigcup G$:

$$[X]_{F \bigcup G, R} \overset{def}{=} \{Y \mid X \to\to Y \in [F \bigcup G]_R\}.$$

Лема 16. Виконуються властивості:

1. $[X]_F \in [X]_{F \bigcup G, R}$;
2. $[X]_{F \bigcup G, R} = [[X]_F]_{F \bigcup G, R}$. □

Зауважимо, що оператор $X \mapsto [X]_{F \bigcup G, R}$ не є оператором замикання; для обґрунтування достатньо вказати, що даний оператор не володіє властивістю ідемпотентності (поняття множини атрибутів і замикання множини атрибутів відносно об'єднання множин ФЗ і БЗЗ $F \bigcup G$ і схеми R мають різну природу, тому вираз $[[X]_{F \bigcup G}]_{F \bigcup G, R}$ не має сенсу в означених термінах).

Базисом $[X]^{bas}_{F \bigcup G, R}$ множини X відносно множини $F \bigcup G$ і схеми R називається підсім'я замикання $[X]_{F \bigcup G, R}$, яка володіє властивостями:

1. $\forall W (W \in [X]^{bas}_{F \bigcup G, R} \Rightarrow W \neq \varnothing)$;
2. $\forall W_i, W_j (W_i, W_j \in [X]^{bas}_{F \bigcup G, R} \Rightarrow W_i \bigcap W_j = \varnothing)$;
3. $\forall Y (Y \in [X]_{F \bigcup G, R} \Rightarrow \exists i, 1 \le i \le n (Y = \bigcup_{W_i \in [X]^{bas}_{F \bigcup G, R}} W_i))$

Лема 17. Виконуються властивості.

1. $\bigcup_{W \in [X]^{bas}_{F \bigcup G, R}} W = R$, для $X \subseteq R$;
2. $A \in [X]_F \Rightarrow A \in [X]^{bas}_{F \bigcup G, R}$. □

Нехай φ – ФЗ або БЗЗ.

Твердження 1 (коректність аксіоматики БЗЗ і ФЗ). Якщо φ синтаксично виводиться з множини $F \bigcup G$, то φ виводиться з $F \bigcup G$ семантично:

$$F \bigcup G \vdash \varphi \Rightarrow F \bigcup G \vDash \varphi \,.\, \square$$

Твердження 2 (повнота аксіоматики БЗЗ і ФЗ). Якщо φ семантично виводиться з множини $F \bigcup G$, то φ виводиться з $F \bigcup G$ синтаксично:

$$F \bigcup G \vDash \varphi \Rightarrow F \bigcup G \vdash \varphi \,.\square$$

Твердження про повноту аксіоматики наводиться в [2].

Теорема 1. Відношення семантичного та синтаксичного слідування для аксіоматики БЗЗ та ФЗ збігаються:

$$F \bigcup G \vDash \varphi \Leftrightarrow F \bigcup G \vdash \varphi \,.\square$$

Критерій повноти аксіоматики багатозначних і функціональних залежностей

Аналіз доведення основного результату про збіжність відношень синтаксичного та семантичного слідування для аксіоматики БЗЗ та ФЗ показує, що воно проведено в припущенні: $|D| \geq 2$ та $|R| \geq 2$. Для повноти викладення залишається розглянути збіжність відношень $\vDash$ та $\vdash$ у випадках $|D| < 2$ або $|R| < 2$.

Лема 18. Усі БЗЗ виконуються на порожній таблиці. БЗЗ вигляду $X \to\to \varnothing$, зокрема, БЗЗ вигляду $\varnothing \to\to \varnothing$ виконується на довільній таблиці. □

Лема 19. Довільна таблиця є моделлю порожньої множини БЗЗ. □

Лема 20. Множина тривіальних БЗЗ замкнена відносно правил повноти, поповнення та транзитивності. □

Лема 21. Всі БЗЗ виконуються на довільній однорядковій таблиці. □

Нехай G – множина БЗЗ, через $(G)_{ntr}$ (None TRivial) позначимо підмножину множини G, що отримується вилученням тривіальних БЗЗ.

Лема 22. Виконуються наступні твердження:

1) $G \vdash \varphi$, якщо $\varphi \in G$;
2) $G \vDash \varphi$, якщо $\varphi \in G$;

3) $G \vdash \varphi \Leftrightarrow (G)_{ntr} \vdash \varphi$;

4) $G \models \varphi \Leftrightarrow (G)_{ntr} \models \varphi$;

5) $G \vdash \varphi$, якщо φ – тривіальна БЗЗ;

6) $G \models \varphi$, якщо φ – тривіальна БЗЗ. □

Залежність збіжності відношень синтаксичного та семантичного слідування для аксіоматики БЗЗ при різних значеннях потужностей множин R та D вказана у табл. 1. Символ "+" (відповідно "–") в комірці означає, що при вказаних припущеннях відношення $\models$ та $\vdash$ збігаються (не збігаються).

Таблиця 1. Всі варіанти потужностей множин R та D для аксіоматики багатозначних залежностей

D \ R	$\|R\|=0$	$\|R\|=1$	$\|R\|\geq 2$
$\|D\|=0$	+	+	–
$\|D\|=1$	+	+	–
$\|D\|\geq 2$	+	+	+

Твердження 3. Заповнення табл. 1 для аксіоматики БЗЗ коректне. □

Лема 23. Виконується імплікація: $X \to Y$ – тривіальна $\Rightarrow$ $X \to\to Y$ – тривіальна. □

Наслідок 12. Множина тривіальних ФЗ і БЗЗ замкнена відносно правил виведення аксіоматики ФЗ і БЗЗ. □

Зауважимо, що спільне для ФЗ і БЗЗ правило $\dfrac{X \to\to Z, Y \to Z',\ Z' \subseteq Z,\ Y \bigcap Z = \varnothing}{X \to Z'}$ не може бути застосовано до двох тривіальних залежностей; це забезпечується умовами $Y \bigcap Z = \varnothing$ і $Z' \subseteq Z$, з яких випливає, що $Y \bigcap Z' = \varnothing$, отже, ФЗ $Y \to Z'$ не є тривіальною.

Залежність збіжності відношень $\models$ та $\vdash$ для аксіоматики БЗЗ та ФЗ при різних значеннях потужностей множин R та D вказана у табл. 2 (усі позначення ті ж самі, що і для табл. 1).

Таблиця 2. Всі варіанти потужностей множин R та D для аксіоматики БЗЗ та ФЗ

D \ R	$\|R\|=0$	$\|R\|=1$	$\|R\|\geq 2$
$\|D\|=0$	+	–	–
$\|D\|=1$	+	–	–
$\|D\|\geq 2$	+	+	+

Твердження 4. Заповнення табл. 2 для аксіоматики БЗЗ та ФЗ коректне. □

Теорема 2. Відношення $\models$ та $\vdash$ для аксіоматики БЗЗ і ФЗ співпадають тоді і тільки тоді, коли $|D| \geq 2$ або $|R| = 0$. □

Висновки

В роботі розглянута аксіоматика багатозначних залежностей в табличних базах даних і аксіоматика функціональних та багатозначних залежностей; встановлені критерії повноти цих аксіоматик в термінах потужностей універсального домену та множини атрибутів. Наступний крок полягає в дослідженні незалежності аксіом та правил виведення для розглянутих аксіоматизацій.

Список використаних джерел

1. Реляційні бази даних: табличні алгебри та SQL-подібні мови / В.Н. Редько, Ю.Й. Брона, Д.Б. Буй, С.А. Поляков. – Київ: Видавничий дім «Академперіодика», 2001. – 198 с.
2. Beeri C. A complete axiomatization for functional and multivalued dependencies / C. Beeri, R. Fagin, J. Howard // Proceedings of the ACM-SIGMOD Conference, August 3-5, 1977, Toronto, Canada. – P. 47-61.
3. Линдон Р. Заметки по логике / Р. Линдон. – Москва: Мир, 1968. – 128 с.
4. Скорняков Л.А. Элементы теории структур / Л.А. Скорняков. – Москва: Наука, 1982. – 160 с.
5. Мейер Д. Теория реляционных баз данных: [пер. с англ.] / Д. Мейер. – Москва: Мир, 1987. – 608 с.

МОЛЕЛЬ ДАННЫХ ДОКУМЕНТО-ОРИЕНТИРОВАННОЙ СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ MONGODB

Д.Б. Буй, Н.Н. Рубан
Киевский национальный университет имени Тараса Шевченко,
Украина
buy@unicyb.kiev.ua, rubannn@gmail.com

Введение

Реляционные базы данных долгое время являлись преобладающим способом хранения данных; немалую роль в этом сыграло использование языка SQL, который опирается на реляционные алгебры Кодда [1]. Реляционные базы данных предоставляют хорошую поддержку для данных с четкой структурой. Однако современные разработки в IT сфере требуют инструментарий для больших данных (Big Data), которые обладают экстремально большим объемом и разнообразием структур данных. Использование традиционных реляционных баз данных, обременительно при обработке таких данных из-за строгих ограничений на структуру данных. Другими словами, канонические структуры данных реляционного подхода сейчас служат тормозом их эффективного применения. С другой стороны базы данных класса NoSQL (Not only SQL), такие как HBase, MongoDB, Cassandra, и т.д. получают все большую популярность, поскольку позволяют обрабатывать большие объемы данных со сложной структурой [2].

Один из типов систем управления базами данных (СУБД) класса NoSQL являются документо-ориентированные СУБД, такие как MongoDB и CouchDB, которые основываются на открытом стандарте представления и обмены данными JSON. Ниже мы рассмотрим формальную модель, которая описывает структуры данных, документо-ориентированной СУБД MongoDB (см. также [3]).

Модель данных

База данных MongoDB состоит из множества коллекций (по аналогии с реляционными базами – таблиц), которые в свою очередь состоят из документов (записей). Каждый документ по своей сути является JSON-структурой – множеством пар ключ-значение. Соответствие двух терминологий представлено в табл. 1.

Таблица 1. Сравнение терминологий MongoDB и SQL

MongoDB	SQL
База данных	База данных
Коллекция	Таблица
Документ	Запись

Перейдем к формальным определениям. Множество всех конечных подмножеств X будем обозначать как $2^X_{fin} = \{X' | X' \subseteq X \;\&\; X' - \textbf{\textit{конечно}}\}$.

Пусть дано множество имен V и множество атомарных данных D. Именные множества, множество которых обозначим как D^V, определяются как конечные отображения из множества имен V во множество D, то есть $D^V \overset{def}{=} \{\alpha \mid \exists V' \in 2^V_{fin}, \alpha : V' \to D\}$.

Множество документов Doc и коллекций Col, строится индуктивно по так называемому рангу (0,1,2,…).

Множество документов ранга 0 совпадает с D^V, которое обозначим через Doc^0 (т.е. $Doc^0 \overset{def}{=} D^V$). Поскольку в качестве значения ключа документа может быть массив поддокументов, то обозначим множество всех таких массивов через $M^0 \overset{def}{=} D^*$, где $D^* = \bigcup_{i=0,1,2,\ldots} D^i$ ($D^0 = \{\Lambda\}$, $D^n = \underbrace{D \times \ldots \times D}_{n}$, $n = 1,2,\ldots$). Говоря содержательно, массив – конечная последовательность (по другой терминологии кортеж). Множество коллекции ранга 0 обозначим как Col^0, оно есть множество всех конечных подмножеств Doc^0, т.е. $Col^0 = 2^{Doc^0}_{fin}$.

Пусть заданы документы и коллекции ранга k. Тогда документы ранга $k+1$ задаются как $Doc^{k+1} = (\bigcup_{i=0}^{k} Doc^i)^V$. Т.е. значением имени может быть атомарное данное либо документ одного из предыдущих рангов.

Аналогично $M^{k+1} = (\bigcup_{i=0}^{k} (M^i \cup Doc^i))^*$. Соответственно множество коллекций ранга $k+1$ задается как множество всех конечных подмножеств документов ранга $k+1$, т.е. $Col^{k+1} = 2^{Doc^{k+1}}_{fin}$.

Пример

Приведем пример документа, который содержит информацию о рецепте блюда:

```
{
   "_id" : ObjectId("5358ce9175c9b773521ca792"),
   "cafe" : "East-West",
   "ingr" : [
         "orange",
         "apple",
          "nut",
         "olive",
         "potato",
   ]
}
```

Модифицируем документ таким образом, чтобы в рецепте отображалась информация о количестве каждого ингредиента:

```
{
   "_id" : ObjectId("5358ce9175c9b773521ca792"),
   "cafe" : "East-West",
   "ingr" : [
      {
         "name" : "orange",
         "weight" : 238
      },
      {
         "name" : "apple",
         "weight" : 573
      },
      {
         "name" : "nut",
         "weight" : 644
      },
      {
         "name" : "olive",
         "weight" : 462
      },
      {
         "name" : "potato",
         "weight" : 584
      }
   ]}
```

Таким образом, в ключе *ingr* получили массив поддокументов, каждый из которых состоит из двух пар ключ-значение (ключи *name* и *weight*).

Выводы

В работе построена модель данных для документо-ориентированной СУБД MongoDB. Построение такой модели делает возможным следующий шаг – рассмотрение манипуляций над данными, т.е. построение алгебры данных.

Список использованных источников

1. Реляційні бази даних: табличні алгебри та SQL-подібні мови / В.Н. Редько, Ю.Й. Брона, Д.Б. Буй, С.А. Поляков. – Київ: Видавничий дім «Академперіодика», 2001. – 198 с.
2. Бэнкер К. MongoDB в действии: [пер. с англ.] / К. Бэнкер. – Москва: ДМК Пресс, 2012. – 394с.
3. Буй Д.Б. Формальне визначення моделі даних, яка використовується в NoSQL базах даних / Д.Б. Буй, С.А. Поляков, Ю.А. Гришко // Вісник Київського національного університету імені Тараса Шевченка. Сер.: фіз.-мат. науки. – 2013. – Спецвипуск. – С. 56-60.
4. Ульман Дж. Основы систем баз данных: [пер. с англ.] / Дж. Ульман. – Москва: Финансы и статистика, 1983. – 334 с.

ITC INTEGRATION IN PRIMARY SCHOOL SUBJECTS

D.L. Geladze
Batumi Shota Rustaveli State University, Batumi, Georgia
darejan66@mail.ru

It is widely believed that Information and Communication Technology (ICT) investment can have an important role to play in raising educational standards. Skinner (1954, 1958) claimed that new technologies in schools could make learning more efficient. As discussed by Angriest and Lavvy (2002), the educational use of computers generally falls under two broad headings: computer skills training, which teaches students how to use computers, and Computer-Aided Instruction (CAI), which uses computers to teach things that may or may not have any relation to technology. Becta (2002) and Ofsted (2001), and similarly points to a positive link between high standards across the curriculum and ICT use in schools. However, as for most of the studies reviewed by Kirkpatrick and Cuban (1998), results are generally inferred from a simple positive correlation between ICT and pupil performance. Fuchs and Woessman (2004), which uses international data from the Programme for International Student Assessment (PISA). The authors show that while the simple bivariate correlation between the availability of computer sat school and school performance is strongly and significantly positive, this becomes small and insignificant when other school characteristics are taken into account. This suggests that establishing whether computers have a causal impact requires experimental or quasi-experimental evidence, where a 'treatment' and 'control' group can be properly defined.

It is difficult to firmly establish a causal relationship between computers and educational outcomes and there are only a small number of studies in the economic literature which try to do so. We examine the relationship between changes in ICT investment and changes in educational performance in Local Education Authorities (LEAs). In contrast with most previous studies in the economic literature, we find evidence for a positive impact of ICT investment on educational performance in primary schools. A positive effect is observed for English and Science, though not for Mathematics. Hence it seems that, in a context where there was a significant expansion of ICT investment, one can uncover evidence of an improvement in pupil achievement linked to ICT. This provides an interesting parallel to the existing work that does not find beneficial effects for pupils and to the related work on firms where there is evidence that ICT investment enhances firm productivity.

We reconcile our positive results with others in the literature by arguing that it is the joint effect of large increases in ICT funding coupled with a fertile background for making an efficient use of it that led to positive effects of ICT expenditure on educational performance in Georgian primary schools.

References

1. Angrist, J., Lavy, V. "New Evidence on Classroom Computers and Pupil Learning", Economic Journal, 112, 2002, 735-765.
2. Becta, British Educational Communications and Technology Agency, "ImpaCT2: The Impact of Information and Communication Technologies on Pupil Learning and Attainment", ICT in Schools Research and Evaluation Series, 7, 2002.
3. Fuchs, T., Woessmann, L. "Computers and Student Learning: Bivariate and Multivariate Evidence on the Availability and Use of Computers at Home and at School", CESifo Working Paper, 1321. 2004.

ДОСЛІДЖЕННЯ ТЕХНОЛОГІЇ ЕЛЕКТРОННОГО НАВЧАННЯ З ВРАХУВАННЯМ ВИМОГ КОРИСТУВАЧІВ

Є.Г. Гнатчук
Хмельницький національний університет, Україна
veta_lina@rambler.ru

Вступ

Вперше термін e-Learning (електронне навчання) був використаний в жовтні 1999 року в Лос-Анджелісі на семінарі CBT Systems. На території країн СНД, зокрема в Україні, частіше використовується термін дистанційне навчання. В результаті більшість спеціалістів розуміють дистанційне навчання, як аналог поняття e-Learning, а інші вважають, що дистанційне навчання більш вузьке поняття, ніж e-Learning, деякі, що більш широке. Під терміном e-Learning (електронне навчання) розуміють навчання, що побудоване з використанням Інтернет-технологій. Існує велика кількість організацій, що працюють в галузі розробки стандартів для e-Learning. Самим розповсюдженим на сьогоднішній день стандартом для e-Learning є SCORM. Використання стандарту SCORM дозволяє забезпечити сумісність компонентів та можливість їх багаторазового використання: навчальний матеріал представлено окремими невеликими блоками, котрі можуть включатись у різні навчальні курси та використовуватись системою дистанційного навчання незалежно від того, ким, де та за допомогою яких засобів вони були створені, передбачається використання дистанційних курсів, які створені різними розробниками [1].

Постановка задачі

Електронне навчання впевнено позиціонує себе як альтернатива традиційному навчанню, загалом це спостерігається в двох сферах: корпоративній та освітянській. В корпоративній сфері e-Learning використовується як початкова підготовка співробітників компанії, їх атестації та підвищення кваліфікації, а в освітянській – для абітурієнтів, як варіант отримання освіти. Лідером по використанню e-Learning систем є США та Канада, а серед європейських країн – Великобританія, Німеччина, Італія та Франція. В США електронне навчання пропонують більше 200 університетів та тисяча коледжів, а кількість онлайнових курсів збільшується приблизно на 30-40% щорічно. Але є дані, які приводить компанія Forrester Research: близько 70% слухачів, що починають вивчати курс не закінчують його, а 11% не повернули свої інвестиції в систему електронного навчання, 21% респондентів використовують більше однієї системи електронного навчання. Це

може бути викликано наступними причинами: 1) користувач вже вивчив те, що вважав за потрібне і вже застосовує отримані знання в роботі; 2) користувач має певні знання з запропонованого курсу та не вважає доцільним проходити його повторно перед підвищенням рівня; 3) матеріал подається в незручній для користувача формі та темпі; 4) відсутність мотивації та самодисципліни.

Отже, необхідно проаналізувати використання систем електронного навчання з позиції користувача та запропонувати рішення, що зроблять навчання комфортним для нього.

Основна частина

Основною проблемою, яка виникає при використанні систем e-Learning, є те, що ці системи не враховують вимог та потреб окремо взятого користувача. За результатами проведених досліджень компанією Knowledge Anywhere на думку співробітників тих організацій, де використовують e-Learning, найбільш важливим для них є можливість навчання на робочому місці в зручний час (49% респондентів). 26% опитаних назвали важливим для себе можливість проходити навчання в зручному для них темпі. 7% респондентів при навчанні використовували відео або аудіо конференції, а 37% компаній використовували змішаний підхід до навчання – суміщаючи відео конференції, e-Learning та інші форми навчання. З цього можна зробити висновок, що вибір форми навчання залишається за організацією та не враховує такий важливий аспект, як можливість та бажання користувача при сприйнятті навчального матеріалу. Отже, більша ефективність використання систем може бути досягнута тільки при детальному врахуванні таких факторів, як структура курсу та методика подачі матеріалу, що вивчається.

Існує чотири загальні види систем електронного навчання [4]. Поділ їх на групи та основне призначення приведено в таблиці 1. Аналізуючи можливості існуючих систем, можна зробити висновок, що крім загальних можливостей для користувача, таких як реєстрація, можливість автоматичного нагадування паролів, можливості обрати курс, отримати оцінку після тестування та інші, що відносяться до сфери організації та керування навчанням, немає можливостей, що стосуються методики подачі матеріалу, врахування індивідуальних особливостей користувача, формування та збереження мотиваційної складової, також крім вивчення теоретичних основ, ще й закріплення практичних навичок.

Тому, доцільно запропонувати такі можливості для користувача, як вибір мови вивчення курсу, проведення початкового тестування стосовно рівня знань на момент початку

навчання у цій дисципліні, та суміжних з нею, врахування особливостей сприйняття матеріалу та його закріплення, взаємодії не тільки з викладачами, але й з іншими користувачами та формувати навчальний контент на основі дослідження поведінки користувача.

Таблиця 1. Види систем електронного навчання

Види систем електронного навчання			
Авторські програмні продукти	Авторські програмні продукти	Системи керування контентом	Системи керування навчанням та навчальним контентом
Дозволяють викладачу створювати власний навчальний контент, не знаючи мов програмування	Надають навчальний контент, контролюють використання навчальних ресурсів, адміністрування окремих слухачів та окремих груп, організовують взаємодію з викладачами, звітність та інше.	Об'єднують та керують великою кількістю людей, що працюють над створенням курсів, яким потрібно використовувати ті самі фрагменти навчальних матеріалів у різних курсах.	Спрямовані на керування змістом навчальних програм, а не самим процесом навчання, і орієнтовані не на слухачів, а на розробників, фахівців з методологічного компонування курсів і керівників проектів навчання

Технічна реалізація системи електронного навчання представляє собою достатньо складний програмно-апаратний комплекс. З програмним забезпеченням працює декілька категорій користувачів: викладачі, слухачі, автори навчальних курсів, адміністратори, менеджери, що контролюють процес навчання. Для того, щоб врахувати потреби користувача, необхідно розробити систему супроводу процесу навчання, що буде аналізувати поведінку користувача і видавати відповідно навчальний контент. Структура системи електронного навчання приведена на рис.1. Компоненти системи можуть взаємодіяти з інформаційними системами організації та з зовнішніми постачальниками навчальних матеріалів. Важливими складовими такої системи є

система супроводу процесу навчання, що буде збирати дані про поведінку користувача, виявляти загальні залежності і робити логічні висновки для ефективної видачі контенту, база даних, де міститься інформація про користувачів, процес навчання та результати навчання, та база знань, що розподіляє навчальний контент.

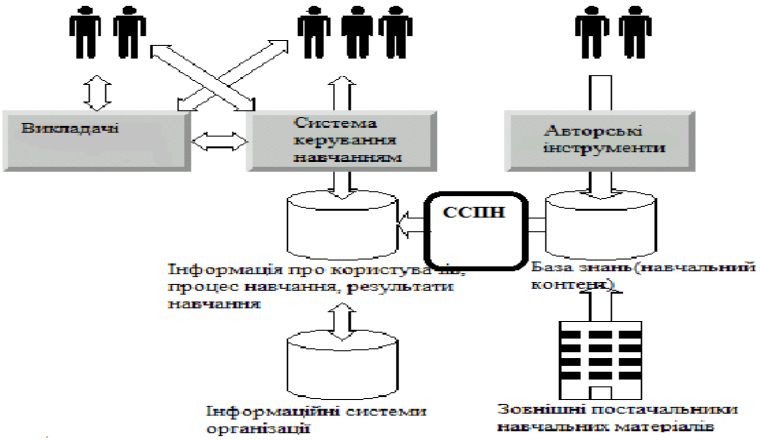


Рис. 1. Структура системи електронного навчання

Висновки

Таким чином, дослідження показали, що в більшості випадків системи електронного навчання враховують вимоги викладачів, авторів навчальних матеріалів, адміністраторів таких систем, а майже не враховують вимоги користувача таких систем. В зв'язку з цим, виникає необхідність розроблення системи супроводу процесу навчання, що буде збирати дані про поведінку користувача, виявляти загальні залежності і робити логічні висновки для ефективної видачі контенту та створення комфортних умов для навчання.

Список використаних джерел

1. Advanced Distributed Learning. Sharable Content Object Reference Model (SCORM) 2004. / Перевод с англ. Е.В. Кузьминой. ─ М.: ФГУ ГНИИ ИТТ "Информика", 2005. ─ 29 с.
2. 50% компаний США используют e-learning. – Режим доступу:

http://www.distancelearning.ru/db/el/6BA99B5047DB50F3C3256C2400258647/doc.html.
3. Электронное обучение в Украине. ─ Режим доступу: http://elearning-ua.blogspot.com/2008_10_01_archive.html. ─ Назва з екрана.
4. Дядичев В.В. Аналіз засобів організації електронного навчання/ В. В. Дядичев, В. Ю. Ващенко // Вісник ЛНУ імені Тараса Шевченка. ─ 2011. ─ № 12 (223), Ч. І. ─ с.97-107.

ПРАВОВИЙ СТАТУС ІДЕНТИФІКАЦІЙНИХ ДАНИХ ОСОБИ ПРИ РЕАЛІЗАЦІЇ МЕХАНІЗМУ ЕЛЕКТРОННОЇ ТРАНСКОРДОННОЇ ІДЕНТИФІКАЦІЇ

Ю.В. Гончарова, І.Д. Горбенко
Харківський національний університет імені В.Н. Каразіна, Україна
bist@karazin.ua, goncharova.yuliia@gmail.com

Вступ

В умовах прискорення світової глобалізації питання надійного та безперешкодного обміну даними між віддаленими суб'єктами стає дедалі більш актуальним. Для забезпечення довіри між суб'єктами таких інформаційних систем вони повинні «представити себе», тобто ідентифікувати.

Обравши євроінтеграційний вектор розвитку, Україна поклала на себе зобов'язання дотримання європейських вимог та стандартів майже в усіх сферах життя, зокрема у питанні безпеки взаємодії кінцевих користувачів транскордонних інформаційних систем. Зараз в Україні розпочато розробку механізму такої ідентифікації, проте як само особи мають представлятися, на жаль, досі не з'ясовано. Понятійну базу щодо електронної ідентифікації в рамках українського правового простору автором було розроблено та викладено у [1].

Питання правомочності використання персональних даних фізичних осіб з метою їхньої ідентифікації у транскордонних інформаційних системах ї є метою цієї роботи.

Співвідношення ідентифікаційних та персональних даних

Власне для забезпечення роботи механізму транскордонної ідентифікації, з боку особи, що бажає бути ідентифікованою, повинні бути надані ідентифікаційні дані. У проекті Закону України «Про внесення змін до Закону України «Про електронний цифровий підпис» було визначено, що ідентифікаційні дані – це унікальний набір даних, який дозволяє встановити тотожність фізичної або юридичної особи, або фізичної особи, яка представляє юридичну особу. На сьогодні відкритим залишається питання належності таких даних до персональних. У більш широкому розумінні досі не зрозуміло, що Законодавцем буде запропоновано у якості ідентифікаційних даних.

Авторами знайдено наступні визначення поняття «ідентифікаційні дані», що вже були надані у нормативно-правовій базі України, які продемонстровано у таблиці 1.

Таблиця 1. Знайдені згадки поняття «ідентифікаційні дані»

Визначення	Джерело
Ідентифікаційні дані - такі дані, що необхідні для того, щоб однаково охарактеризувати окремий предмет, партію або страту. Прикладом є: зона балансу матеріалу, тип ядерного матеріалу, ідентифікація партії, опис матеріалу, а також вид і дата зміни інвентарної кількості.	Облік та контроль ядерного матеріалу, фізичний захист ядерного матеріалу і ядерних установок. Державний комітет ядерного регулювання України. Тлумачний словник українських термінів. НП 306.7.086-2004 08.06.2004 N 101
Ідентифікаційні дані - відомості про фізичних осіб: а) резидентів - прізвище, ім'я та по батькові; дата народження; серія і номер паспорта (або іншого документа, що посвідчує особу), дата видачі та орган, який його видав; відомості про місце проживання або місце перебування; ідентифікаційний номер (реєстраційний номер облікової картки платника податків) згідно з Державним реєстром фізичних осіб - платників податків або серія та номер паспорта, у якому проставлено відмітку органів державної податкової служби про відмову в одержанні ідентифікаційного (реєстраційного) номера; б) нерезидентів - прізвище, ім'я та по батькові (за наявності); дата народження; серія і номер паспорта (або іншого документа, що посвідчує особу), дата видачі та орган, який його видав; громадянство; відомості про місце проживання або місце тимчасового перебування фізичної особи в Україні.	Постанова правління Національного банку України «Про затвердження Положення про здійснення банками фінансового моніторингу» (Положення, п.1.2) 14.05.2003 N 189. Поточна редакція — Прийняття від 31.01.2011

На семінарі, проведеному 6 серпня 2012 року Державною службою України з питань захисту персональних даних було надано перелік особових даних, що зазначено у таблиці 2, де,

зокрема визначено множину ідентифікаційних даних: прізвище, ім'я, по батькові, адреса, телефон тощо.

Таблиця 2. Перелік даних, наданий державною службою України з питань захисту персональних даних, що можна віднести до особових

Загальні персональні дані	«Чутливі» персональні дані
ідентифікаційні дані (прізвище, ім'я, по батькові, адреса, телефон тощо);	расова належність;
паспортні дані;	політичні погляди;
особисті відомості (вік, стать, сімейний стан тощо);	релігійні переконання;
склад сім'ї;	світоглядні переконання;
освіта;	стан здоров'я
професія;	стан статевого життя;
біометричні дані (зріст, вага, особливі прикмети тощо);	членство в політичних партіях та професійних спілках;
психологічні дані (особистість, характер тощо);	
житлові умови;	
спосіб життя;	
життєві інтереси та захоплення;	
споживчі звички;	
фінансова інформація;	
електронні ідентифікаційні дані (трафік, IP-адреса тощо);	
електронні дані про локалізацію (GSM, GPS тощо);	
запис зображень (фото, відео);	
звукозапис;	
інші персональні дані	

На думку авторів такий перелік не можна вважати еталонним, оскільки неочевидна необхідність деяких елементів для її ідентифікації – наприклад, номеру телефону, використання якого доцільно лише у випадку, якщо телефон приймає участь у процесі ідентифікації, адже повсюдне використання мобільних пристроїв а також верифікація особи за їхньої допомогою стає більш розповсюдженою.

При цьому за Законом України «Про захист персональних даних» під персональними даними розуміють відомості чи сукупність відомостей про фізичну особу, які ідентифікована або

може бути *ідентифікована*. А отже, можна поставити знак рівності між «ідентифікаційними» даними та «персональними даними».

Проте важливо виокремити множину інформації, що становить ідентифікаційну. На думку авторів, доцільно використовувати визначений Правлінням національного банку перелік: саме він розмежовує резидентів та не резидентів країни, що є значним аспектом саме транскордонної ідентифікації. Перелік особових персональних даних є надлишковим для реалізації міжнародної ідентифікації, оскільки не задовольняє основній меті: встановлення однозначної тотожності фізичної або юридичної особи, власника (володільця, розпорядника) інформаційної системи на підставі ідентифікаційних даних про особу, зазначених в документах, складених в паперовій та/або електронній формі.

Регулювання транскордонної передачі персональних даних українським законодавством

У сучасному світі інформація вільно може вийти за межі національного кордону держави, оскільки світ характеризують глобалізаційні процеси, підкріплені високим ступенем науково-технологічних досягнень – кордони між державами не здатні стримувати інформаційні потоки.

Згідно ст. 29 Закону України «Про захист персональних даних» передача персональних даних іноземним суб'єктам відносин, пов'язаних із персональними даними, здійснюється лише за умови забезпечення відповідною державою належного захисту персональних даних у випадках, встановлених законом або міжнародним договором України. Держави - учасниці Європейського економічного простору, а також держави, які підписали Конвенцію Ради Європи про захист осіб у зв'язку з автоматизованою обробкою персональних даних, визнаються такими, що забезпечують належний рівень захисту персональних даних.

У тексті вищезгаданою Конвенції термін «персональні дані» означає будь-яку інформацію, яка стосується конкретно визначеної особи або особи, що може бути конкретно визначеною.

Персональні дані можуть передаватися іноземним суб'єктам відносин, пов'язаних з персональними даними, також у разі:

1) надання суб'єктом персональних даних однозначної згоди на таку передачу;

2) необхідності укладення чи виконання правочину між володільцем персональних даних та третьою особою - суб'єктом персональних даних на користь суб'єкта персональних даних;

3) необхідності захисту життєво важливих інтересів суб'єктів персональних даних;

4) необхідності захисту суспільного інтересу, встановлення, виконання та забезпечення правової вимоги;

5) надання володільцем персональних даних відповідних гарантій щодо невтручання в особисте і сімейне життя суб'єкта персональних даних.

Персональні дані не можуть поширюватися з іншою метою, ніж та, з якою вони були зібрані.

Міжнародно-правовий інститут захисту персональних даних та їх безперешкодної транскордонної передачі містить відповідні норми та принципи, що направлені на забезпечення прискорення темпів передачі інформації крізь кордони. Проте ця ціль нерозривно пов'язана із міжнародно-правовим захистом прав людини на приватність, що складає стрижневий елемент усього міжнародного-правового механізму регулювання транскордонної передачі персоніфікованої інформації [4]

Висновки

В межах євроінтеграційних намірів нашої держави відбуваються чималі зміни у нормативно-правовій складовій захисту інформації. З'являються нові послуги та можливості, що позитивно впливають на глобалізаційний аспект, але потребують належного регулювання з боку законодавця. Проте питання непорушності прав особистих прав особи не повинно незмінно головувати та забезпечувати інформаційну безпеку особи незалежно від законів та норм країни, з суб'єктом-резидентом якої відбувається зв'язок. Долаючи державні кордони, такі дані не повинні бути залишені поза увагою та захистом жодної держави.

Перелік використаних джерел

1. Горбенко Ю.І. Електронна ідентифікація: поняття, визначення, вимоги / Ю.І. Горбенко, Ю.В. Гончарова // Прикладна радіоелектроніка. – 2014 – N 176 – С. 138-144.
2. Закон України «Про захист персональних даних» - Відомості Верховної Ради України (ВВР), 2010, № 34, ст. 481.
3. Конвенція про захист осіб у зв'язку з автоматизованою обробкою персональних даних - Страсбург, 28 січня 1981 року. Дата ратифікації Україною: 06.07.2010.
4. Международное информационное право. – Кафедра кибермира. Режим доступу: http://cyberpeace.org.ua/ru/o_kafedre_kibermira .html. Дата звернення: 29/11/2014

МЕТОДИ ПОБУДОВИ ЗАГАЛЬНОСИСТЕМНИХ ПАРАМЕТРІВ ВЕЛИКИХ ПОРЯДКІВ ДЛЯ ЦИФРОВОГО ПІДПИСУ ЗА ДСТУ-4145-2002

І.Д.Горбенко, Р.С. Ганзя
Харківський національний університет радіоелектроніки, Україна
gorbenkoi@iit.kharkov.ua, ganzya@iit.kharkov.ua

Вступ

На даний момент в Україні використовується національний стандарт електронного цифрового підпису ДСТУ-4145, який прийнято у 2002 році, з розмірами загальносистемних параметрів від 163 до 509 бітів [1]. Проте враховуючи сучасний розвиток науки та техніки у напрямку квантової фізики, особливо у напрямку розробки квантових комп'ютерів, може виникнути ситуація, коли національний стандарт асиметричної криптографії України буде не стійким проти атак, що базуються на квантових алгоритмах [2].

Загрозою до усієї асиметричної криптографії, що базується на таких математичних проблемах, як складність факторизації великого цілого числа та складність рішення дискретного логарифму в полі та в групі точок еліптичної кривої є квантовий алгоритм Шора. Детальний опис цього алгоритму наведено у роботі [3]. В таблиці 1 наведено наші результати аналізу обчислювальної складності алгоритму Шора, що може бути запущений на квантовому комп'ютері з відповідними характеристики, та атаки методом ρ – Полларда.

Таблиця 1. Обчислювальна складність квантового та класичного методу розв'язку дискретного логарифму в групі точок еліптичної кривої

Розмір порядку базової точки, бітів	Кількість потрібних кубітів	Складність квантового алгоритму, кв. гейтів	Складність класичного алгоритму, оп. складання
110	808	$0.5\cdot10^{9}$	$3.6\cdot10^{16}$
163	1210	$1.6\cdot10^{9}$	$3.4\cdot10^{24}$
256	1834	$6\cdot10^{9}$	$3.4\cdot10^{38}$
509	3610	$4.7\cdot10^{10}$	$4\cdot10^{76}$
571	4016	$6.7\cdot10^{10}$	$8.8\cdot10^{85}$
1024	7218	$3.8\cdot10^{11}$	$1.3\cdot10^{154}$
2048	14390	$3.1\cdot10^{12}$	$1.8\cdot10^{308}$

Квантовий алгоритм Шора має поліноміальну складність розв'язку дискретного логарифму, в той час як найкращий

класичний алгоритм криптоаналізу криптосистеми на базі еліптичних кривих має експоненційну складність. Хоча збільшення розмірів базової точки не дає суттєвих підвищень стійкості проти квантового криптоаналізу, проте як показує аналіз алгоритму Шора, при збільшенні розмірів базової точки необхідна більша кількість кубітів квантового комп'ютера для реалізації атаки. На даний момент найбільша кількість кубітів реалізована в квантовому комп'ютері D-Wave Two, що складає 512 кубітів [4]. Проте такий комп'ютер насправді не може реалізувати квантові алгоритми, як наприклад Шора та Гровера. У зв'язку з цим одним з шляхів рішення загрози квантових комп'ютерів може бути збільшення розмірів загальносистемних параметрів асиметричних криптосистем. Таким чином нашою задачею було розроблення програмного засобу, який здійснює генерацію еліптичних кривих з використанням ефективних методів генерації.

Методи формування загальних параметрів

На момент створення національного стандарту ДСТУ-4145 найбільш придатним методом формування загальних параметрів у групі точок еліптичних кривих над полем $F(2^m)$ був метод Сатоха [5].

Для генерації загальних параметрів в групі точок еліптичних кривих можуть використовуватися l-адичні та p-адичні методи. L-адичні методи підрахунку кількості точок на еліптичній кривій можуть використовуватися для скінчених полей будь-яких характеристик, проте вони, як правило, застосовуються для підрахунку кількості точок в групі точок еліптичних кривих над великими простими полями $F(p)$. Для еліптичних кривих над скінченими полями малих характеристик використовуються p-адичні алгоритми.

Першим ефективними p-адичним методом для підрахунку кількості точок на еліптичній кривій був метод Сатоха, розроблений у 1999 році. В цьому методі було показано, що використання p-адичних методів для полей з невеликими характеристиками значно ефективніше, ніж використання l-адичних методів. На даний момент було запропоновано такі p-адичні методи: метод Сатоха, алгебро-геометричний метод та інші, що так чи інакше пов'язані з попередніми.

Метод Сатоха для визначення порядку кривої використовує властивості ендоморфізму Фробеніуса. Обчислення виконується у два етапи:

1) Підняття циклу ізогінеї та коефіцієнтів кривої;

2) Обчислення сліду ендоморфізму Фробеніуса, на базі якого визначається порядок кривої.

Підняття виконується послідовним підняттям j-інваріантів, коефіцієнтів кривої разом з підняттям підгруп l-крутіння з попереднім обчисленням циклу кривих та j-інваріантів j_1 [5].

Інший метод, який дуже сильно пов'язаний з методом Сатоха базується на алгебро-геометричному методі (AGM) та був запропонований Местре [6] та реалізований Харлі. Харлі також показав, що даний метод може бути надзвичайно ефективним. Обчислювальна складність метода Сатоха та АГМ, що був запропонований Местре складає $O(n^{3+\xi})$.

Запропонований метод Сатоха-Ск'єрна-Тагучі (SST) має складність $O(n^{2.5+\xi})$, проте це метод вимагає передобчислень [8].

Для полей характеристики 2 на даний момент AGM вважається швидким для великих криптографічних розмірів, через дуже малі розміри констант, він також є кращим алгоритмом для обчислення записів. Перший недолік алгоритму SST - це етап передобчислень, який не є допустимим для записів, але це зовсім не проблема для великих криптографічних розмірів, де це легко здійснити і зберігати з попередньо обчислюваними даними. Ще однією проблемою в методі SST є те, що заданий поліном для кільця є щільним, тим самим збільшується складність множення на коефіцієнт 3 у порівнянні з розрідженою структурою полінома, що використовується у AGM [7]. Основні принципи AGM будуть наведені нижче.

Для використання методу AGM необхідна крива E задана рівнянням $y^2 + xy = x^3 + a_6$ з елементом a_6 над F_q^* з j-інваріантом $j(E) = a_6^{-1}$. Необхідно позначити, що елемент a_6 є довільним елементом з Z_q, що зменшено до a_6 за модулем 2. Далі отримаємо рекурсивну формулу, що дасть нам цілком вірну послідовність (A_i, B_i) елементів з Z_q:

$$A_0 = 1 + 8a_6, \quad B_0 = 1,$$

$$A_{i+1} = \frac{A_i + B_i}{2}, \quad B_{i+1} = \sqrt{A_i + B_i},$$

де квадратний корінь вибирається так, щоб він був конгруентним до 1 по модулю 4. Таким чином у роботі [7] показано, що якщо

квадратний корінь один раз було задано конгруентним до 1 по модулю 4, то ця пропорція буде зберігатися на кроці $i+1$.

Послідовність (A_i, B_i) називається AGM-послідовністю і її можна привести до послідовності еліптичної кривої E, за допомогою виразу $y^2 = x(x - A_i^2)(x - B_i^2)$. Позначимо як j_1 j-інваріант еліптичної кривої E.

Добре відомо, що AGM пов'язаний з ізогінезисом степені 2 між еліптичними кривими. Цей крок дає зв'язок з канонічним підйомом у наступній теоремі.

Теорема 1. Нехай j_1 є j-інваріантом еліптичної кривої E, що є пов'язаним з AGM послідовністю. Тоді для послідовності j_i виконуються конгруенції:

$$j_0 \equiv a_6^{-2} \bmod 2, \quad j_{i+1} \equiv j_i^2 \bmod 2, \quad j_i \equiv j_i^{\uparrow} \bmod 2^{i+2}.$$

Перше співвідношення показує, що E є ізоморфною за сполученим підйомом з початковою еліптичною кривою E по модулю 2. Друге співвідношення стверджує, що всі еліптичні криві E_i також зменшуються за модулем 2 до сполучень кривої E (з точністю до ізоморфізму). Третя конгруенція є основною для AGM алгоритму: це означає, що, коли послідовність піднімається по AGM, ми все ближче наближаємось до канонічного підйому.

Це дає простий алгоритм для обчислення канонічного підйому. Починаючи з початкових значень (A_0, B_0), ми застосовуємо рекурсивну формулу для обчислення послідовних значень (A_i, B_i). Після k кроків, ми можемо обчислити j-інваріант асоційованої кривої, близький до канонічного підйому сполученої E, піднятої до точності 2^k [8].

Зв'язок цієї теореми зі слідом Фробеніуса дає нам ефективний метод для підрахунку кількості точок еліптичної кривої E.

Теорема 2. Нехай $i > 0$, та нехай c_i є $Norm_{Z_q/Z_p}(\frac{A_{i+1}}{A_i})$ тоді виконується рівність:

$$c_i + q / c_i = Tr(E) \bmod 2^{i+4}.$$

Алгоритм для підрахунку кількості точок на еліптичній кривій полягає у наступних кроках: розрахунок AGM-послідовності

необхідну кількість разів (кроків), далі розрахунок нормування дасть нам слід початкової кривої піднятої до деякої точності, що відповідає кількості шагів та деякої константи.

Висновки

Таким чином, на даний момент існують загрози для асиметричної криптографії, що пов'язані з квантовим алгоритмом Шора. Для підвищення стійкості національного стандарту цифрового підпису ДСТУ-4145 можна збільшити розмір базової точки. Для цього можуть використовуватися ефективні методи побудови загальносистемних параметрів, наприклад, алгебро-геометричний метод та його модифікації. Нами був розроблений програмний засіб на мові C++ з використанням бібліотеки NTL, який здатен будувати еліптичні криві для ДСТУ-4145-2002 з розміром базової точки 1031 біт.

Список використаних джерел

1. ДСТУ-4145-2002 "Інформаційні технології. Криптографічний захист інформації. Цифровий підпис, що грунтується на еліптичних кривих. Формування та перевіряння".
2. Shor, P. W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer [Text] / P. W. Shor //SIAM J. Comput. - 1997. - 26 (5). - pp. 1484–1509.
3. Ганзя, Р.С. Аналіз можливостей квантових комп’ютерів та квантових обчислень для криптоаналізу сучасних криптосистем / Р.С.Ганзя, Ю.І.Горбенко // Харків: «Восточно - Европейский журнал передових технологий». –2014. – Том 6 №1(67). – 8-15 с.
4. Launching the quantum artificial intelligence lab [Electronic resource] / Блог компанії Google: Режим доступа : URL: - http://googleresearch. blogspot.ru/2013/05/launching-quantum-artificial.html - 16.05.2012 р.
5. Горбенко, І. Д. Прикладна криптологія. [Текст]: монографія / І. Д. Горбенко, Ю. І. Горбенко; ХНУРЕ. – Х.: Форт, 2012. - 868 с.
6. Mestre, J. F. AGM pour le genre 1 et 2. [Text] / J. F. Mestre // Lettre à Gaudry et Harley - December 2000.
7. Gaudry, P. A comparison and a combination of SST and AGM algorithms for counting points of elliptic curves in characteristic 2 / P. Gaudry // ASIACRYPT 2002. 8th International Conference on the Theory and Application of Cryptology and Information Security Queenstown - New Zealand: Springer, 2002 – P.311-327.
8. Cohen, H. Elliptic and Hyperelliptic Curve Cryptography [Text]: handbook / H. Cohen, G. Frey – NW.: Chapman & Hall/CRC, 2006. - 807 p.

ПРОЕКТ НАЦІОНАЛЬНОГО СТАНДАРТУ СИМЕТРИЧНОГО БЛОКОВОГО ШИФРУВАННЯ

І.Д. Горбенко[1], Р.В. Олійников[1], В.І. Руженцев[1], О.В. Казимиров[1], О.О. Кузнецов[2], Ю.І. Горбенко[3], О.Є. Дирда[4], В.І. Долгов[2], А.І. Пушкарьов[4], Р.І. Мордвинов[1], Д.С.Кайдалов[1]

[1]Харківський національний університет радіоелектроніки, Україна

[2]Харківський національний університет ім. В.Н. Каразіна, Україна

[3]Приватне акціонерне товариство „Інститут інформаційних технологій", Харків, Україна

[4]Державна служба спеціального зв'язку та захисту інформації, Одеса, Україна

roliynykov@gmail.com, gorbenkoi@iit.kharkov.ua

Вступ

Тенденції розвитку сучасного постіндустріального суспільства демонструють необхідність подальшого інтенсивного впровадження нових інформаційних технологій. Децентралізація і віртуалізація обчислень, масове застосування безпровідних каналів зв'язку додатково збільшують існуючу критичну залежність всього суспільства від надійності та безпеки інформаційно-телекомунікаційних систем (ІТС). В таких умовах забезпечення інформаційної безпеки сучасних і перспективних ІТС стає однією з найпріоритетніших задач. Необхідною умовою для цього є застосування систем криптографічного захисту інформації, що дозволяють ефективно забезпечувати послуги конфіденційності та цілісності для розподілених обчислювальних систем. Водночас засоби криптографічного захисту інформації, які існують зараз, у ряді випадків не можуть забезпечити рівень пропускної здатності, який повністю відповідає сучасним вимогам. Найефективніше ця проблема може бути вирішена за допомогою підходу, який передбачає вдосконалення алгоритмів криптографічного перетворення, зниження їхньої обчислювальної складності при збереженні або збільшенні стійкості. А оскільки основний обсяг інформації, що передається каналами зв'язку, захищається за допомогою симетричних криптографічних примітивів, значну увагу необхідно приділяти саме цьому типу перетворень, зокрема, симетричним блоковим шифрам.

Огляд алгоритмів симетричного блокового шифрування, що зараз використовуються в Україні

Стандарт блокового шифрування ДСТУ ГОСТ 28147:2009 [1] був розроблений і введений в дію 25 років тому ще в СРСР. Він вже виведений із дії в Білорусії та планується до заміни в Росії. У відповідних умовах цей алгоритм ще забезпечує достатню

практичну стійкість, але є атаки, що відомі з літератури (теоретичні атаки) із складністю, значно меншою повного перебору. Для цього шифру існують великі класи слабких ключів (сеансових і довгострокових), що значно погіршують криптографічні властивості та навіть дозволяють виконувати практичні атаки на персональному комп'ютері. Крім того, швидкодія цього шифру суттєво нижча ніж у сучасних аналогів.

Крім ДСТУ ГОСТ 28147:2009, в Україні застосовуються TripleDES [2] (у банківських системах, що імпортовані або орієнтовані на застарілі стандарти), AES [3] та інші шифри (у складі операційних систем загального призначення).

TripleDES може забезпечувати практичну стійкість, але він також вразливий до теоретичних атак, а також суттєво повільніший навіть за ДСТУ ГОСТ 28147:2009.

AES має високу швидкодію (в т.ч. низку апаратних акселераторів у складі процесорів загального призначення), забезпечує практичну стійкість і є найбільш дослідженим у світі криптографічним алгоритмом. Водночас, для цього шифру відомі теоретичні атаки із складністю, меншою, ніж повний перебір, і він не може в повній мірі використати можливості 64-бітових платформ при програмній реалізації.

Таким чином, існуючі рішення мають певні недоліки, і в Україні потрібно введення нового стандарту блокового шифрування із властивостями, що дорівнюють або перевищують відомі іноземні рішення.

Вимоги до перспективного алгоритму симетричного блокового шифрування

При розробці проекту стандарту були висунуті наступні вимоги:

- високий рівень криптографічної стійкості із достатнім запасом на випадок появи нових атак протягом тривалого часу;
- висока швидкодія програмної реалізації на сучасних та перспективних платформах;
- компактність програмної і апаратної реалізації, можливість ефективної інтеграції декількох алгоритмів в одному засобі криптографічного захисту;
- прозорість проектування, консервативний підхід щодо забезпечення стійкості ;
- вища (або порівняна) ефективність щодо найкращих світових рішень;

- наявність низки режимів роботи, необхідних для ефективної реалізації сучасних засобів криптографічного захисту.

Загальна конструкція та властивості блокового шифру „Калина”

Криптографічне перетворення забезпечує високий і надвисокий рівень стійкості, підтримує розмір блоку 128, 256 і 512 біт та таку ж довжину ключа, має 10, 14 або 18 циклів SPN-перетворення. У алгоритмі шифрування застосована нова схема розгортання ключів, що використовує операції раундового перетворення та забезпечує додатковий захист від атак на реалізацію.

При розробці раундового перетворення застосовані чотири різних S-блоки із різних класів еквівалентності, із алгебраїчними властивостями, кращими ніж у AES, та нелінійністю вищою, ніж у білоруського та російського стандартів СТБ 34.101.31-2011 та ГОСТ Р 34.11-2012 “Стрибог” (в проекті національного стандарту України використане найкраще відоме у світі співвідношення для захисту від різних атак).

Шифр орієнтований на 64-бітові платформи, але забезпечує достатню ефективність і на сучасних 32-бітових. Реалізація передбачає наявність одного набору таблиць передобчислень, що є оптимізованим для швидкодіючої реалізації як зашифрування, так і формування циклових ключів (ці ж таблиці використовуються для ефективної реалізації алгоритму, що є проектом національного стандарту ґешування).

Криптографічне перетворення забезпечує стійкість до відомих методів криптоаналізу (диференційний, лінійний, інтегральний, бумеранг, усічені диференціали та інші), із суттєвим запасом.

Порівняння швидкодії блокового шифра “Калина” із іншими алгоритмами

На сучасних 64-бітових платформах перспективний алгоритм шифрування забезпечує вищу або порівняну ефективність щодо найкращих світових рішень. Результати вимірювання для системи на основі процесора Intel(R) Core(TM) i7-3770K CPU @ 3.50GHz під ОС Linux, компілятор gcc version 4.9.2 (Ubuntu 4.9.2-0ubuntu1~12.04, 30-Oct-2014), наведені у таблиці 1 та на рис.1.

Таблиця 1. Швидкодія блокових шифрів

№ з/п	Блоковий шифр	Мбіт/с
1	Kalyna-128/128	2844.29
2	Kalyna-128/256	1977.38
3	Kalyna-256/256	2153.48
4	Kalyna-256/512	1684.84
5	Kalyna-512/512	1483.29
6	AES-128	2684.65
7	AES-256	2140.85
8	GOST28147-89	741.95

Рис. 1. Порівняння швидкодії блокових шифрів

Таким чином, для оптимізованих версій перевага швидкодії „Калини" над AES складає від 1% до 6% (від 13 Мбіт/с до 160 Мбіт/с) при однаковій довжині ключа в залежності від процесору та довжини ключа (або порівняну швидкодію при 128-бітовому блоці та 256-бітовому ключі). На 256-бітовому ключі Калина виконує обробку швидше ніж ГОСТ 28147-89 від 2,6 до 3,16 разів (швидкість вища на 726-1412 Мбіт/с).

Додатково слід зазначити, що Калина забезпечує суттєво більш високий запас стійкости до криптоаналітичних атак ніж AES.

Режими роботи, визначені у проекті стандарту

Крім базового режиму (простої заміни, ECB), проект стандарту визначає дев'ять режимів роботи: гамування, гамування зі зворотнім зв'язком за шифртекстом, вироблення імітовставки, зчеплення шифрблоків, гамування зі зворотнім зв'язком за шифргамою, вибіркове гамування із прискореним виробленням імітовставки, вироблення імітовставки і гамування, індексованої заміни і захисту ключових даних. Частина режимів роботи охоплює вимоги міжнародного стандарту ISO/IEC 10116:2006. Інші режими потрібні для сучасних засобів криптографічного захисту (шифрування трафіку IP-мереж, в т.ч. Інтернет, носіїв інформації, ключових даних малого обсягу та ін.)

Висновки

Проект національного стандарту симетричного блокового перетворення забезпечує високий і надвисокий рівень стійкості із достатнім запасом на випадок появи нових атак та вдосконалення криптоаналітичних комплексів протягом тривалого часу. Алгоритм має високу швидкодію програмної реалізації на сучасних та перспективних платформах загального призначення, вищу або порівняну ефективність щодо найкращих світових рішень. Введені режими роботи дозволяють ефективну реалізацію сучасних засобів криптографічного захисту. Крім того, при розробці передбачена можливість ефективної інтеграції декількох національних алгоритмів в одному засобі криптографічного захисту.

Список використаних джерел

1. ГОСТ 28147–89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования [Текст]. – Введ. 01–07–1990. – М. : Изд-во стандартов, 1989. – 28 с.
2. Data Encryption Standard (DES) [Electronic resource] : FIPS PUB 46-3. – 1999 – Mode of access : www. URL: http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf.
3. Advanced Encryption Standard (AES) [Electronic resource] : FIPS PUB 197. – 2001 – Mode of access : www. URL: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

АНАЛІЗ МОЖЛИВОСТЕЙ ВИКОРИСТАННЯ ПОСТКВАНТОВИХ КРИПТОСИСТМ В СУЧАСНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ

Ю.І. Горбенко[1], *Р.С. Ганзя*[2]
[1]Харківський національний університет ім. В.Н.Каразіна, Україна
[2]Харківський національний університет радіоелектроніки, Україна
gorbenkou@iit.kharkov.ua, ganzya@iit.kharkov.ua

Вступ

Безпека сучасних інформаційних систем та технологій базується на стійкості криптографічних перетворень, які вони використовують для криптографічної обробки інформації. Так з використанням алгоритму Шора [1] складність криптоаналізу таких криптосистем як RSA, DSA, ECC буде поліноміальною, хоча з використанням класичних алгоритмів криптоаналізу, відомих на сьогодні, складність атаки на такі криптосистеми є субекспоненційною або експоненційною. Використовуючи алгоритм Гровера [2] можна зменшити складність атаки на симетричні криптосистеми та деякі криптосистеми на основі фактор-кілець.

Після винайдення Шором у 1994 році поліноміального алгоритму знаходження періоду функції, велика кількість криптографів та математиків почали працювати у напрямку постквантової криптографії, тобто криптографії, яка буде існувати після появи квантового комп'ютера. На даний момент вже існують певні класи криптосистем, які за своєю структурою будуть стійкими до квантового криптоаналізу на основі алгоритмів Шора та Гровера. Такі криптосистеми, як правило, презентуються та обговорюються на щорічній конференції PQCrypto, що проводиться з 2006 року. За останні роки вже було запропоновано певні класи криптосистем, що будуть стійкими до квантового криптоаналізу на основі алгоритмів Шора та Гровера [3].

На даний момент є актуальною задача дослідження стійкості постквантових криптосистем, а також розробка програмних засобів для можливостей їх подальшого використання у інформаційних системах на національному та міжнародному рівні.

Можливості протидії квантовому криптоаналізу

Основним напрямком протидії квантовим алгоритмам криптоаналізу є винайдення нових класів криптосистем, стійкість яких не ґрунтується на математичних проблемах складності факторизації цілого числа чи розв'язку дискретного логарифму. Напрямок таких досліджень названо терміном постквантова криптографія, тобто криптографія, яка буде стійкою навіть після

появи квантового комп'ютера, що буде мати велике число кубітів для своєї роботи. Цей напрямок робіт популяризувався після 2006 року, саме тоді була проведена перша конференція за цією тематикою (PostQuantumCrypto 2006). Така конференція проводиться щорічно і збирає для обговорення найкращі здобутки за рік у напрямку постквантової криптографії.

Що стосується симетричних криптосистем, то більшість сучасних симетричних шифрів та геш-функцій захищені від квантових комп'ютерів. Квантовий алгоритм Гровера може прискорити криптоаналіз таких криптосистем, але йому можна протидіяти збільшенням розміру ключа чи блоку повідомлення [4]. Короткий аналіз можливостей алгоритму Гровера, щодо криптоаналізу симетричних криптосистем, буде наведено нижче.

Отже, враховуючи останні досягнення у напрямку постквантової криптографії, можна виділити такі класи криптосистем, що будуть стійкими до квантового криптоаналізу [3]:

- Криптографія на основі решіток.
- Мультиваріативна криптографія.
- Криптографія на основі геш-функцій.
- Криптографія на основі кодів.
- Криптографія ізогінеї суперсингулярних еліптичних кривих.
- Симетрична криптографія.

Аналіз можливостей застосування криптографії на основі решіток у постквантовому світі

Криптографія на основі решіток є новітнім класом альтернативних схем відкритого ключа і в даний час являє собою активну область для досліджень. Є кілька складних проблем, які можуть бути використані для побудови криптосистеми на базі решіток, найпопулярнішою з них є проблема знаходження найкоротшого вектору.

Цей напрямок включає в себе такі криптографічні системи, як NTRU та GGH. Згідно до Micciancio і Регева [5], криптографію на основі решіток можна розділити на дві категорії: практичні криптосистеми, такі як NTRU, які не володіють доказами безпеки таких схем, і теоретичні, такі як матриці на основі навчання з помилками (Learning with Errors, LWE), які пропонують сильні докази безпеки, але використовують ключі, які є занадто великими для загального використання. З 2008 року ведуться дослідження по об'єднанню цих двох категорій, проте на даний момент ще не вдалося об'єднати ці дві категорії і створити новий клас

криптографічних систем на основі решіток, які б діяли ефективно як NTRU і були б доказово стійкими як LWE.

Протягом останніх років було запропоновано низку атак на криптосистему NTRU. Найкращою відомою класичною атакою є атака «зустріч посередині», що була запропонована у Одлижко у 2003 році, з використанням ідеї цієї атаки та квантового алгоритму Гровера у 2012 році була запропонована ефективна квантова атака на NTRU «зустріч посередині», яка в подальшому була удосконалена Вангом. Аналітична часова і просторова складність таких атак наведена у таблиці 1.

Таблиця 1. Аналітична складність атак на NTRU

	Класична атака зустріч посередині	Удосконалена квантова атака зустріч посередині
Часова складність	$O(\frac{C_{N/2}^{d/2}}{\sqrt{N}})$	$O(C_{\lfloor N/3 \rfloor}^{\lfloor d/3 \rfloor} \log C_{\lfloor N/3 \rfloor}^{\lfloor d/3 \rfloor}) + \overline{O}(\sqrt{C_{N-\lfloor N/3 \rfloor+1}^{d-\lfloor d/3 \rfloor}})$
Просторова складність	$O(\frac{C_{N/2}^{d/2}}{\sqrt{N}})$	$O(C_{\lfloor N/3 \rfloor}^{\lfloor d/3 \rfloor})$

Детальний аналіз стійкості криптосистеми NTRU проти квантового та класичного криптоаналізу наведено у [8]. Результати досліджень показали, що хоча квантова атака і потребує менше операцій, проте вона вимагає все одно субекспоненційної кількості квантових гейтів для реалізації на реальні криптосистеми.

Отже криптографічні конструкції на основі решіток мають великі перспективи для свого використання в інформаційній сфері, так криптосистема NTRU вже стандартизована (стандарт IEEE 1363.1) і використовується для надання послуги конфіденційності. Для України є важливим розвиток цього напрямку, враховуючи те, що ця криптосистема є стійкою до квантового криптоаналізу, має невеликі розміри загальносистемних параметрів і має великі швидкості за шифрування/розшифрування.

Криптографія на основі ізогінеї еліптичних кривих

Криптографія, що заснована на ізогінеї представляє цікаву альтернативу наведеним вище напрямкам постквантової криптографії [6]. Це пояснюється тим, що вона базується на природній обчислювальній проблемі теорії чисел, а саме на

проблемі обрахунку ізогінезису між еліптичними кривими. Ці системи можна віднести до одного з напрямків постквантової криптографії, такі системи базуються на теоретико-числовому припущенні. Враховуючи те, що національний стандарт вироблення та перевірки цифрового підпису базується на еліптичних кривих, то такий напрямок є надзвичайно важливим і перспективним для використання в Україні.

В роботі [6], наведено аналітичні показники стійкості такої криптосистеми, так показано, що складність криптоаналізу такої криптосистеми з використанням класичних комп'ютерів складає $O(\sqrt[4]{p})$, для квантових комп'ютерів – $O(\sqrt[6]{p})$.

Порівняльний аналіз квантового та класичного криптоаналізу криптосистем на базі ізогінеї еліптичних кривих наведено в таблиці 2.

Таблиця 2. Порівняльний аналіз стійкості криптосистеми на базі ізогінеї еліптичних кривих проти класичного та квантового криптоаналізу

Розмір поля *GF(p)*, біт	512	768	1024	2048
Класична атака, бітових операцій	$3\cdot10^{38}$	$6\cdot10^{57}$	10^{77}	10^{154}
Квантова атака, квантових гейтів	$5\cdot10^{25}$	$3\cdot10^{38}$	$2\cdot10^{51}$	$5\cdot10^{102}$

Отже криптосистеми на базі ізогінеї еліптичних кривих мають велику стійкість як проти квантового, так і проти класичного криптоаналізу, при цьому розміри поля мають невеликі розміри.

Висновки

Якщо коротко охарактеризувати постквантові криптосистеми, то криптосистеми на базі решіток, на даний момент, призначені більше для шифрування інформації, а схеми вироблення цифрових підписів на основі решіток є менш розвинуті, ніж схеми шифрування. В той же час криптосистеми на базі геш-функцій та мультиваріативні поліноми на даний момент призначені більше для вироблення цифрових підписів, ніж для шифрування інформації. Криптосистеми на базі ізогінеї зараз більш пов'язані з шифруванням та з протоколом автентифікації. Класично протоколи автентифікації суб'єкта пов'язується зі схемою Фіат-Шаміра, проте така схема є вразливою до можливостей квантового криптоаналізу.

Нещодавно було запропоновано схему на основі ізогінезису еліптичних кривих, яка пропонує схему виготовлення та перевірки цифрового підпису, автентифікації та може бути застосована у постквантовому світі [6]. На даний момент важко побудувати схеми цифрового підпису, які будуть стійкими до квантового криптоаналізу, єдина відома незаперечна схема підпису, що була запропонована, заснована на лінійних кодах.

Уся симетрична криптографія на даний момент є стійкою і не існує поліноміального за складністю алгоритму, який би скомпрометував такий клас криптосистем.

Список використаних джерел

1. Shor, P. W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer [Text] / P. W. Shor //SIAM J. Comput. - 1997. - 26 (5). - pp. 1484–1509.
2. Grover, L. K. A fast quantum mechanics algorithm for database search [Text] / L. K. Grover. // - Proceeding of the 28th ACM Symposium on Theory of Computation, New York: ACM Press. – 1996. - pp. 212-219.
3. Bernstein, D. Post-quantum cryptography [Text] / D. Bernstein, J. Buchmann, E. Dahmen. – Berlin: Springer, 2009. – 246 p.
4. Горбенко, Ю.І. Аналіз стійкості популярних криптоситем проти квантового криптоаналізу на основі алгоритму Гровера / Ю.І.Горбенко, Р.С.Ганзя // Київ: Захист інформації: Науково-практичний журнал,2014. – Том 16, №2. – 106–112 с.
5. Stefan Heyse. Post quantum cryptography: implementing alternative public key schemes on embedded devices : dissertation for the degree of doktor-ingenieur : 10.2013 / Stefan Heyse. – Bochum, 2013. – 235 p. – Bibliogr. : pp. 205–223.
6. De Feo, L. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies [Text] / L. De Feo, D. Jao, J. Plut // PQCrypto. – 2011. – 24 p.

СЦЕНАРІЇ СТВОРЕННЯ ТА ПЕРЕВІРКИ ВДОСКОНАЛЕНИХ ЕЛЕКТРОННИХ ПІДПИСІВ В МОБІЛЬНОМУ СЕРЕДОВИЩІ

Ю.І. Горбенко[1], К.В. Ісірова[2]
[1]ПАТ «Інститут інформаційних технологій», Харків, Україна
[2]Харківський національний університет імені В.Н. Каразіна, Україна
GorbenkoU@iit.kharkov.ua, katisa2008@mail.ru

Вступ

В Європейському Союзі (ЄС) вже 15 років успішно впроваджуються і застосовуються електронні послуги. Відповідно до Директиви 1999/93 / ЄС впроваджено і успішно застосовується електронний цифровий підпис (ЕЦП). Досвід його застосування дозволив зробити позитивні оцінки, а також виявити проблемні питання, необхідність його удосконалення та розвитку. Також зроблено висновок про необхідність розширення електронних послуг, з метою зробити їх в межах ЄС транскордонними. Результати досліджень і розробок були представлені і затверджені в Регламенті 2012 [1]. У 2014 році прийнято Регламент Європейського Парламенту та Ради з питань електронної ідентифікації і довірчих послуг (сервісів) для електронних операцій на внутрішньому ринку (Регламент 2014) [2]. В даному документі розширюється сфера застосування ЕЦП. Технологія ЕЦП використовується для надання електронних послуг мітки часу та електронної печатки. Крім того ЕЦП ставав основним елементом підтвердження інших електронних послуг - електронної ідентифікації, автентифікації, друку, електронного документа, надійної електронної доставки. В термінології Регламенту 2014 він отримав назву електронного підпису (ЕП).

Механізми електронної ідентифікації та електронного підпису у складі електронних довірчих послуг в Євросоюзі розглядаються як основа реалізації поставлених завдань. Впорядкувати положення, пов’язані з нормативним регулюванням вище зазначених питань покликаний так званий Мандат 460 [3].

Мобільні пристрої в рамках Мандату 460

В рамках Мандату 460 окремо вважається необхідним виділити положення, пов’язані з мобільними пристроями, оскільки в умовах сучасного світу, мобільність має суттєву роль. Детально ці позиції висвітлені в [4].

Нижче представлена основа для подальшої стандартизації, створення та перевірки вдосконалених електронних підписів в мобільному середовищі (тобто в умовах, коли мобільні пристрої,

підтримують мережеві послуги для створення та / або перевірки підпису) з урахуванням останніх поліпшень в можливостях мобільних пристроїв і їх суміщення з можливостями інших обчислювальних пристроїв.

Наведемо вимоги стандартизації для сценаріїв, які враховують наступне:

а) Локальне підписання, при якому підпис створюється з використанням ключа, який зберігається на мобільному пристрої користувача.

б) Дистанційне підписання, при якому підпис створюється з використанням ключа, який зберігається на віддаленому сервері.

в) Віддалену перевірка підпису, при якій перевірка відбувається на віддаленому сервері [4].

Характеристики

Опишемо особливості сценаріїв вдосконалених мобільних електронних підписів, які характеризують відмінності між сценаріями за допомогою відповідей на такі питання:

а) Чи створено документ на мобільному пристрої або надається стороннім додатком.

б) Де знаходиться документ, а також всі атрибути підпису.

в) Що відображається користувачеві (ґеш документа, весь документ, резюме основних елементів і т.д.); необхідно відзначити, що деяка інформація може бути доступна також на іншому пристрої (наприклад ПК), а не на самому мобільному пристрої.

г) Яким чином користувач контролює процес створення та / або перевірки підпису.

д) Де зберігатися особистий ключ і де формується сам підпис.

е) Як автентифікується підпис.

ж) Чи використовуються специфічні сервіси (наприклад SMS сервіс).

з) Чи можна застосувати сценарій до інших обчислювальних пристроїв або тільки до мобільних обчислювальних пристроїв [4].

Сценарії для мобільних пристроїв

Сценарій 1. Документ генерується мобільним пристроєм. Дайджест формується провайдером сервісів мобільного підпису. Цифровий підпис формується мобільним пристроєм. Удосконалений цифровий підпис формується провайдером сервісів мобільного підпису.

У даному сценарії документ може бути створений або за допомогою мобільного пристрою, або за допомогою віддаленого

додатку, який доступний користувачеві (крок 1). Клієнтський протокол провайдера сервісів мобільного підпису потім створює (крок 2) запит на генерацію вдосконаленого цифрового підпису в кооперативному. Потім клієнт подає запит провайдеру сервісів мобільного підпису (крок 3), який формує дайджест (крок 4), який повинен бути підписаний. Провайдер сервісів мобільного підпису посилає (крок 5) запит для підписання його на мобільний пристрій. Потім мобільний пристрій записує дайджест в пристрій для створення підпису та після авторизації користувача, обчислює (крок 6) цифрове значення підпису. Це значення потім представляється (крок 7), в якості підтвердження провайдеру сервісів мобільного підпису. Після того, як провайдер має це таке значення він може сформувати (крок 8) вдосконалений цифровий підпис та надати його (крок 9) клієнту.

Сценарій 2. Документ генерується провайдером додатків. Дії провайдера сервісів мобільного підпису ініційовані провайдером додатків. Дайджест формується мобільним пристроєм. Удосконалений цифровий підпис формується провайдером сервісів мобільної пошти.

В цьому сценарії користувач хоче використовувати віддалений додаток (крок 1). Постачальник додатків формує і передає (крок 2) запит для генерації вдосконаленого електронного підпису провайдеру сервісів мобільного підпису, який, в свою чергу, обчислює дайджест та інші атрибути, які повинні бути підписані (крок 3). Потім запит для підписання дайджесту відправляється на мобільний пристрій (крок 4). Після того, як користувач повідомляється і дозволяє обчислення цифрового підпису (крок 5), SCD виконує обчислення (крок 6) і мобільний пристрій надає відповідь, в тому числі значення цифрового підпису (крок 7) провайдеру сервісів мобільного підпису. Він генерує вдосконалений цифровий підпис (крок 8), включає його в свою відповідь постачальнику додатків, і надає відповідь (крок 9).

Сценарій 3. Документ генерується провайдером додатків. Діяльність провайдера сервісів мобільного підпису ініціюється провайдером додатків. Удосконалений цифровий підпис генерується мобільним пристроєм.

Користувач хоче взаємодіяти з провайдером додатків. Звернення може бути зроблено з мобільного пристрою або з іншого пристрою, наприклад ПК. В результаті такого звернення (крок 1), провайдер додатків робить висновок про необхідність вдосконаленого цифрового підпису даному користувачеві. Потім він формує і відправляє (крок 2) запит на генерацію підпису провайдеру сервісів мобільного підпису. Провайдер сервісів

мобільного підпису потім створює запит підпису (крок 3) і направляє його на мобільний пристрій (крок 4). Після того, як мобільний пристрій отримує цей запит, користувачеві надається деякий текст, що повідомляє запит (може бути весь документ), а також інші дані, які ясно вказують на наслідки задоволення цього прохання. Потім користувач дає згоду на підписання, наприклад, введенням PIN-коду SIM-карти (крок 5). SIM-карта потім переходить до генерації вдосконаленого цифрового підпису (крок 6), можливо, додаючи елементи, наприклад, сертифікат і мітку часу, якщо процедура не виконується на сервері підписання. Потім мобільний пристрій відправляє відповідь, включаючи в нього підпис (крок 7) на сервер підписання, який, в свою чергу, посилає провайдеру додатків вдосконалену цифровий підпис отриману від мобільного пристрою користувача (крок 8).

Сценарій 4. Документ генерується провайдером додатків. Підпис запитується мобільним пристроєм. Удосконалений цифровий підпис генерується провайдером сервісів підпису.

Користувач може використовувати додаток, наданий провайдером додатків, як результат того, що додаток генерує (крок 1) і являє документ, який повинен бути підписаний (крок 2). Якщо документ створюється локально ці кроки будуть пропущені.

Клієнтський протокол провайдера сервісів підпису працює в мобільному пристрої, генеруючи (крок 3) і відправляючи (крок 4) запит для генерації вдосконаленого цифрового підпису провайдеру сервісів підпису. Провайдер сервісів підпису генерує вдосконалений цифровий підпис і дає відповідь на запит (крок 5), після цього відправляє його назад клієнтові (крок 6). Потім користувач може обробляти прийняті вдосконалені цифрові підписи (крок 7), як дозволені додатки, запущеного на мобільному пристрої. У більшості випадків, підписаний документ буде відправлений провайдером додатків.

Сценарій 5. Удосконалений цифровий підпис генерується провайдером сервісів підпису під прямим контролем.

Обмін схожий на сценарій сервера підпису, який описаний вище з додатковим кроком 5, де "Дані активації підпису" надається з мобільного пристрою, який активує використання ключа підпису для конкретного документа (наприклад, за допомогою одноразового пароля).

Сценарій 6. Підпис створюється за допомогою спліт-ключа.

В цьому сценарії документ підготовлений локально або за допомогою віддаленого провайдера додатків (крок 1). "Попередній цифровий підпис" створюється на мобільному пристрої, використовуючи частину розділеного ключа, що зберігається

локально (крок 2). Попередній підпис відправляється провайдеру сервісів підпису (крок 3), створення цифрового підпису завершується за допомогою спліт- ключа, що зберігається у провайдера сервісів підпису для ідентифікованого підписувача (крок 4). Остаточний цифровий підпис потім повертається на мобільний пристрій.

Сценарій 7. Використання мобільного пристрою в процесі перевірки удосконаленого цифрового підпису.

Користувач, що має мобільний пристрій, отримує підписаний документ (крок 1). Він вимагає перевірки підпису (крок 2), використовуючи клієнтський протокол сервера, який працює в його мобільному пристрої. Клієнт генерує і відправляє запит на перевірку підпису (крок 3) до віддаленого серверу перевірки. Віддалений сервер перевірки проводить перевірку вдосконаленого цифрового підпису (підписів) в документі (крок 4), і формує відповідь-підтвердження для клієнта (крок 5). Результат перевірки належним чином надається користувачеві (крок 6) на його мобільний пристрій.

Висновки

Таким чином, ми бачимо, що розглянуті сценарії можна розділити на декілька груп: сценарії локального підписання (сценарії 1, 2, 3), сценарії віддаленого підписання (сценарії 4, 5, 6) та сценарій віддаленої перевірки підпису (сценарій 7). Дану класифікацію можливо провести, спираючись на основні характеристики, які вже були згадані вище.

Загалом, хочеться відзначити важливість таких тенденцій розвитку надання електронних довірчих послуг. Впровадження таких положень значно підвищить мобільність користувачів, що сприятиме розвитку системи електронних довірчих послуг.

Список використаних джерел

1. Пропозиція регламенту Європейського парламенту та Ради з питань ідентифікації та трастових послуг для електронних операцій на внутрішньому ринку (2012).
2. Регламент Європейського парламенту та Ради з питань ідентифікації та трастових послуг для електронних операцій на внутрішньому ринку (2014).
3. Mandate M460: «Standardisation Mandate to The European Standardisation Organisations CEN, CENELEC and ETSI in the Field of Information and Communication Technologies Applied to Electronic Signatures».
4. SR 019 020 Rationalised Framework of Standards for Advanced Electronic Signatures in Mobile Environment.

ОБҐРУНТУВАННЯ ВИМОГ ТА РОЗРОБКА СКРИПТОВОЇ МОВИ БЕЗПЕЧНОГО УПРАВЛІННЯ В ІНФОРМАЦІЙНО-ТЕЛЕКОМУНІКАЦІЙНИХ СИСТЕМАХ

Ю.І. Горбенко[1], В.А.Пономар[2]
[1]ПАТ «Інститут інформаційних технологій», Харків, Україна
[2]Харківський національний університет імені В. Н. Каразіна, Україна
GorbenkoU@iit.kharkov.ua, Laedaa@gmail.com

Вступ

В ряді інформаційно-телекомунікаційних систем (ІТС) для написання розширень програмних алгоритмів зазвичай застосовується скриптова мова, яка переважно використовується у випадках, коли:

1) Потрібно забезпечити можливість створення нових програм (алгоритмів, протоколів) без ризику дестабілізації системи.

2) Важливими є вимоги прозорості та зрозумілості коду. Крім того, скриптова мова, як правило, має власний проблемно-орієнтований набір команд, і один рядок скрипту може робити те саме, що і кілька десятків рядків на традиційній мові.

Вимоги до скриптової мови зі сторони захисту інформації

Основна вимога зі сторони захисту інформації полягає в тому, щоб нові програми, що створюються, не змогли нашкодити ІТС. Ця вимога задовольняється безпосередньо самим принципом виконання програм, написаних на скриптовій мові.

Також існують й інші вимоги, серед яких необхідно відзначити такі:

1) Користувач скриптової мови не повинен мати безпосередній доступ до скритих системних даних[1].

2) Скриптова мова повинна не давати можливості написання небезпечних команд управління.

3) Тільки автентифікований оператор, для якого в політиці безпеки надається така можливість, має право завантажувати написані команди.

4) Команди користувача не повинні замінити команди розробника.

Якщо захист інформації забезпечується за допомогою використання спеціальних засобів захисту, наприклад, модулів криптографічного захисту інформації (КЗІ), то виникає дві великих проблеми. По-перше, збільшується необхідний код, бо з'являється програмний прошарок, для забезпечення взаємодії обраної скриптової мови та криптографічного модуля. По-друге, через це необхідне значне збільшення швидкодії.

Тому конче необхідне створення перспективної спеціалізованої скриптової мови для використання в ІТС, у яких застосовуються модулі криптографічного захисту інформації. При цьому ми повинні дотримуватись необхідних правил в частині забезпечення сумісності з подібними чи при оновлені даної ІТС. На основі наведеного вище рекомендується при розробці перспективної скриптової мови використовувати мови С, С++ або Perl, але вибір мови залежить ще й від необхідних правил роботи програм, з якими необхідна сумісність, та методів перетворення [3].

Специфікація скриптової мови

Код виконуваної програми розбивається на відокремлені блоки. В блоках кожен запис закінчується символом «;», перенос рядків дозволяється, але даним символом не закінчуються записи тегів, що повідомляють про початок та кінець блока. Наведемо перелік можливих блоків: «PROGRAM», «COMMAND», «DEFINE», «PARAMETERS», «KEYREGISTERS», «INDATA», «OUTDATA», «SECUREDATA», «VARIABLES», «CODE». Тег про закінчення блоку має той самий вигляд, що початок, але до нього додається слово «END» без пробілу, наприклад, тег закінчення програми – «ENDPROGRAM».

Блок «PROGRAM» є головним блоком програми – він є самою програмою. При об'яві програми через пробіл необхідно вказати її тип: «NORMAL» чи «REMOVE». «NORMAL» – це звичайна програма добавлення нових команд до криптографічного модуля; «REMOVE» – це програма видалення команд з криптографічного модуля. Безпосередньо в блоці «PROGRAM» можуть знаходитися блоки «COMMAND» та «DEFINE».

Блок «COMMAND» описує команду (її параметри та код реалізації), що необхідно додати до криптографічного модуля, чи вказує, яку саме команду необхідно видалити. Через пробіл вказується ідентифікатор команди (її номер) та його розширення (якщо воно є). Якщо програма має тип «REMOVE», то на цьому блок команди закінчується, а якщо тип «NORMAL», то далі йдуть підблоки команди. Підблоками команди можуть бути всі інші блоки в будь-якій послідовності, але блок «PARAMETERS» повинен бути обов'язково і він повинен бути першим. Блок «CODE» теж повинен бути обов'язково і стояти останнім.

В блоці «PARAMETERS» вказуються за допомогою ключових слів параметри даної команди:

– стани модуля при яких команда може викликатися;

– ролі користувачів, які можуть її викликати;

– ключові маски (вказують, які ключі використовувати);

– границі довжин вхідних та вихідних даних.

Блоки «DEFINE», «KEYREGISTERS», «INDATA», «OUTDATA», «SECUREDATA», «VARIABLES» описують данні, що будуть використовуватися в коді програми. До них є спільна вимога – не повинно бути однакових імен та імен, що співпадають з системними змінними криптографічного модуля.

Блок «DEFINE» описує заміни та макроси. Якщо блок знаходиться безпосередньо в блоці «PROGRAM», то це є глобальними константами, і вони діють в усіх командах даної програми, а якщо він розташований у блоці «COMMAND» - то тільки в межах даної команди

Блок «KEYREGISTERS» вказує ключові регістри. Максимальна кількість регістрів залежить від криптографічного модуля для якого пишеться програма.

Блок «VARIABLES» описує змінні даної команди. Записується ім'я змінної, потім, в квадратних дужках, кількість елементів, розмір кожного елементу. При аналізі цього блоку контролюється умова, якою розмір елементів був в визначених границях, а також їх сумарний розмір не перевищував дозволений криптографічним модулем.

Блоки «INDATA», «OUTDATA», «SECUREDATA» описують структури вхідних, вихідних та захищених даних, що використовуються криптографічними функціями модуля. Вимоги до запису елементів структури ті ж самі, що й до запису елементів блоку «VARIABLES». В межах однієї структури імена повинні бути різними.

В блоці «CODE» знаходиться сама кодова реалізація алгоритму команди.

Правила написання коду команди

Кожен запис закінчується символом «;». Записом може бути виклик криптографічної функції модуля, виклик алгоритмічної функції, об'ява мітки, вираз присвоєння змінній нового значення.

Криптографічна функція може бути функцією виконання чи функцією повернення значення. Функція повернення значення повертає певне значення від криптографічного модуля (результат роботи чи певний параметр). Ця функція може бути як головною функцією запису, так і одним з його параметрів. Функція виконання вимагає від модуля виконання певної криптографічної дії. Ця функція може бути лише головною функцією запису. Всі криптографічні функції модуля описуються в додатковому файлі, що підключається компілятором та у своїх назвах мають признак криптографічної функції (пропонується, щоб їх назва починалася з символів «CRYPTO_» чи якогось скорочення). Параметрами функції можуть бути константи чи змінні, арифметичні вирази,

функції, що повертають значення. Параметри можуть бути вкладеними (одним з параметрів може бути функція з значенням, що повертається, параметром якої може бути теж функція, у якої теж параметри, і т.д.) [4].

Алгоритмічні функції – функції скриптової мови, кожна з них може бути лише головною функцією запису.

Функція безумовного переходу «GOTO». У ній через пробіл вказується ім'я мітки до якої треба перейти.

Функція повернення значення «EXIT». Вона завершує виконання команди та повертає значення, що вказується через пробіл.

Функції умовного переходу мають три параметри, що вказуються в дужках через кому. Перші два – значення, що порівнюються (це можуть бути константи, змінні, арифметичні вирази, криптографічні функції, що повертають значення), а третій мітка переходи, до якої необхідно перейти при виконанні умови. Усього є шість функцій даного типу, що передбачають шість можливих варіантів відношень значень

Мітка об'являється функцією «LABEL», у якій і через пробіл вказується її ім'я.

У арифметичних виразах операція присвоєння «=» може бути лише головною операцією (функцією) запису, всі інші операції можуть використовуватися тільки для визначення параметрів функції (або результату, що буде присвоєний). Значення може присвоюватися лише до змінної чи елементу масиву.

Наведемо перелік можливих операцій: арифметична сума, арифметична різниця, арифметичне множення, арифметичне ділення націло, побітове AND, побітове OR, побітове виключне OR (XOR), побітове NOT (інверсія), зсув вліво, зсув вправо.

При використані арифметичних операції можливе використання пріоритетних скобок. Індекс елементу масиву вказується в квадратних скобках після імені. Індекс може бути змінною. Під час виклику, імена елементів структури вказуються відокремленням символом «.» від імені самої структури. Через вимоги безпеки дані ключових регістрів та елементів структури даних криптографічного модуля користувач не може викликати в явному вигляді.

Структура вихідного байт-коду

Першими байтами бінарного файлу є спеціальні байти, що вказують модуль, для якого призначена дана програма, тип програми, тип завантаження та інше. Далі йде перелік команд – номер команди, флаги станів, флаги ролей, мінімальний та максимальний розміри вхідних та вихідних даних, номер байту

файлу з якого починається код команди, довжина коду. Після усього переліку йде спеціальне заповнення перед кодом. Код команди починається з двох спеціальних байтів, що вказують на початок коду нової команди. Далі йдуть байти запису одинарних операцій. На кожну операцію відводиться від 4 до 16 байтів.

Перший DWORD (4 байти) байт-коду містить в собі опис операції. Перший байт – ідентифікатор операції. Наступні два байти – байти флагів. Вони вказують тип операції (криптографічна функція, функція скриптової мови, арифметична операція), тип кожного параметру (константа, змінна, спеціальні данні), крім типу параметру вказується, ще його розмір в байт коді (для змінної розмір – це значення її зміщення). Опис розміру вводиться для оптимізації вихідного коду. Четвертий байт – резервний. Усі наступні можливі 0..12 байтів – значення параметрів.

Останні 4 байти бінарного файлу – його контрольна сума.

Алгоритм обробки рядка коду, що виконується

Після зчитування рядка скрипта в блоці CODE (рядок закінчується символом «;») виконується аналіз головної операції рядка. Це може бути операція присвоєння, мікрооперація засобу КЗІ, об'ява мітки переходу чи закінчення блоку CODE, функція переходу до мітки. Після цього починається формування байт-коду. Оскільки виклик операцій може бути рекурсивним, то спочатку повністю формується байт-код усіх вкладених операцій, а потім цей байт-код записується на вихід у порядку від операцій найнижчого рівня до головної операції. Алгоритм обробки в залежності від вихідної операції є наступним [4, 5]:

1. Мікрооперація засобу КЗІ. Починається формуватися байт-код. Порівнюється кількість наявних параметрів з необхідним. В тому разі, якщо є складний параметр, то спочатку він обробляється, а потім результат вкладається в байт-код.

2. Присвоєння. Аналізується об'єкт до якого відбувається присвоєння, це значення може бути змінною, елементом масиву чи регістром. Потім аналізується, що присвоюється, цей аналіз відбувається за тим самим алгоритмом, що й аналіз одного параметру мікрооперації.

3. Функція кінця виконуваного блоку. Звичайна функція з одним параметром – кодом помилки.

4. Об'ява мітки. Байт-код не формується, але у таблицю міток заноситься адреса знаходження даної мітки.

5. Функція переходу (умовного чи безумовного). Байт-код формується за тими ж правилами, що й мікрооперація засобу КЗІ, але параметр, що відповідає адресі мітки до якої треба перейти не заповнюється, а лише залишається місце для нього, а в таблицю

міток для даної мітки записується адреса, куди необхідно буде записати адресу мітки.

Особливість роботи з мітками полягає в тому, що операція переходу може виконуватися до об'яви мітки, тобто коли невідома її адреса. Саме для цього і формується таблиця міток переходу. Після закінчення обробки скрипта виконується прохід по таблиці міток, і в усі місця параметрів функцій переходу записується адреса необхідної мітки переходу. Арифметичні вирази обробляються за алгоритмом формування та обробки зворотнього польського запису. Дії над константами виконуються відразу, а над іншими типами даних записуються у байт-код у вигляді операції.

Оцінка нової скриптової мови

Для перевірки ефективності даних вимог була створена скриптова мова управління засобом КЗІ (мова УЗ КЗІ). На вхід інтерпретатора подавався текст скрипта, а на виході отримувався бінарний код для завантаження у криптографічний модуль.

Крім того, що є найважливішим в компіляторі цієї мови вже реалізовані необхідні вимоги безпечного управління, що були перелічені в попередніх розділах. В таблиці 3 наведено порівняння властивостей мови УЗ КЗІ з іншими скриптовими мовами та мовами програмування.

Таблиця 3. Порівняння властивостей мов програмування

Можливість	УЗ КЗІ	C	C++	Python	Perl	Ruby	Tcl
Статична типізація	+	+	+	-	+	-	-
Динамічна типізація	-	-	-	+	+	+	+
Явна типізація	+	+	+	-	-	-	-
Неявна типізація	-	-	-	+	+	+	+
Приведення типів	+	+	+	+/-	+	+/-	+
Некеруємі покажчики	+	+	+	-	-	-	-
Ручне керування памяттю	-	+	+	-	-	-	-
Команда goto	+	+	+	-	+	-	-
Команда break	-	+	+	+	+	+	+
Багатовимірні масиви	-	+	+	+	+	+	+
Динамічні масиви	-	+	+	+	+	+	+
Макроси	+	+	+	-	+	+	+

При порівняльному аналізу нової скриптової мови та існуючих при використанні для безпечного управління засобом КЗІ виявилось, що мова УЗ КЗІ має більшу швидкодію, вже забезпечує необхідні механізми безпеки управління і самі по собі транслятор з компілятором займають значно менше місця у порівнянні з універсальними мовами. Недоліки прогнозуються в тому, що при фундаментальній зміні засобу КЗІ виникає необхідність у значній переробці компілятора мови УЗ КЗІ, в той час коли універсальні мови, що були адаптовані для виконання задачі управління криптографічним модулем, потребують лише незначної зміни функцій парсингу програмного коду.

Висновки

Управління різноманітними ІТС – одна з областей інформаційної діяльності, в якій використовуються скриптові мови.

Скриптова мова безпечного управління модулем КЗІ створена у вигляді скриптової мови спеціального призначення, з універсальністю суб'єкта управління та оптимізацією вихідного байт-коду.

Інші скриптові мови мають більший функціонал (особливо якщо рахувати додаткові бібліотеки), але для модуля КЗІ це є зайвим функціоналом, в якому немає необхідності, і ті функції, що дозволяє скриптова мова безпечного управління модулем КЗІ є необхідними та достатніми для його повного функціонування.

Список використаних джерел

1. Turn your scripting language into a code generator [Electronic resource] / Some in-depth articles about programming, technology, science and mathematics. — Режим доступу: \www/ URL: http://www.gener8.be/site/articles/code_generation/ code_generation.html/ — 2011.
2. Ousterhout, J. Scripting: Higher-Level Programming for the 21st Century [Text] / J. Ousterhout // IEEE Computer. — 1998. — Т. 31, № 3. — P. 23–30.
3. Пратт, Т. Языки программирования: разработка и реализация [Текст]/ Т. Пратт, М. Зелковиц. — 4-е издание. — СПб.:Питер, 2002. — 688 с.
4. Ахо, А. Компиляторы: принципы, технологии и инструментарий [Текст]/ А. Ахо, М. Лам, Р. Сети, Д. Ульман. — М: ИД «Вильямс», 2008. – 723с.
5. Hoare C. A. R. Record Handling[Text] / C. A. R. Hoare // Programming Languages.—1968. — P. 291 – 347/

КОМПЬЮТЕРНЫЕ ТРЕНАЖЕРЫ В УЧЕБНОМ ПРОЦЕССЕ

Д.З. Дидманидзе, Н.О. Худжадзе
Батумский государственный университет имени Шота Руставели,
Батуми, Грузия
ddidari@mail.ru

В настоящее время существуют весьма жесткие требования к современным тренажерным системам и комплексам. Поэтому современная законченная тренажерная система должна включать в себя помимо средств "зрительной симуляции" средства "чувствительной симуляции". При обучении исключительно на компьютерных тренажерах, всегда есть и будет опасность подготовки не реальных, а "виртуальных специалистов", неспособных к профессиональному выполнению реальных задач. Немалую роль при производстве тренажеров также играет и программное обеспечение. В основном используется то же программное обеспечение, что используется при создании современных компьютерных игр. Требования к качеству графики могут быть различны и зависят от конкретного проекта. Немалую роль играет поддержка 3D графики на современном уровне, а также возможность использования 3D моделей, созданных в популярных программах для их создания. Также с помощью программного обеспечения могут имитироваться реальные физические законы, что позволяет существенно расширить спектр возможного применения тренажеров.

При разработке тренажеров могут использоваться как стандартные интерфейсы типа клавиатуры и мыши, так и более сложные, например, джойстики с системой обратной отдачи или даже перчатки и шлемы виртуальной реальности. Степень и качество визуализации в программных приложениях могут быть достаточно сложными. При разработке могут быть использованы сетевые технологии, позволяющие создавать комплексные тренажеры, на базе которых можно проводить обучение сразу нескольких специалистов.

Создание тренажеров на основе тех или иных реальных объектов может стоить очень дорого, а доверять молодым специалистам управление действующей аппаратурой – мероприятие весьма рискованное. Для того чтобы позволить таким специалистам отрабатывать свои профессиональные навыки, активно применяются виртуальные тренажеры.

Виртуальные интерактивные тренажеры ничем не уступают обычным тренажерам, однако позволяют отработать те или иные

профессиональные навыки без различных рисков и больших затрат. В них активно применяются различные современные мультимедийные технологии, технологии виртуального окружения и 3D графика, что позволяет с наибольшей эффективностью симулировать условия и законы реальной жизни в виртуальной реальности.

Интерактивные тренажеры дают возможность людям тренировать и вырабатывать те или иные специальные навыки. Такие тренажеры могут представлять собой как приложение для компьютера со стандартной конфигурацией, так и целые комплексы программного обеспечения, требующие наличия мощных компьютеров и другого специального оборудования. Тренировка происходит в автоматическом режиме, то есть обучаемый может самостоятельно, без посторонней помощи, проходить обучение. При взаимодействии с компьютерным тренажером, обучаемому передаются различные инструкции в виде обычного текста, звуковой или видео информации.

Список используемых источников

1. Софиев А.Э. Компьютерные обучающие системы / А.Э. Софиев, Е.А. Черткова – Москва: ДеЛи принт, 2006. — 296 с.
2. Дидманидзе И.Ш. Тренажёры в учебном процессе. / И.Ш. Дидманидзе, Д.З. Дидманидзе, Н.О. Худжадзе // XXIII international conference problems of decision making under uncertainties (PDMU – 2014). ABSTRACTS. May 12-16, 2014. – Mukachevo, 2014. – P. 95-96.
3. Дидманидзе И.Ш. Использование тренажеров в учебном процессе / И.Ш. Дидманидзе, Н.О. Худжадзе, Д.З. Дидманидзе // Материалы II международной научно-практической интернет конференций `СУЧАСНІ ТЕНДЕНЦІЇ РОЗВИТКУ МАТЕМАТИКИ ТА ІІ ПРИКЛАДНІ АСПЕКТИ – Донецк, 2013. – С. 205-206.

АЛГОРИТМЫ РЕАЛИЗАЦИИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ПОДДЕРЖКИ БЕЗОПАСНОСТИ БЕЗПРОВОДНЫХ СЕТЕЙ

И. Ш. Дидманидзе, З. Р. Беридзе
Батумский государственный университет имени Шота Руставели, Батуми, Грузия
ibraimd@mail.ru

Эффективность решений, связанных с автоматизированными системами поддержки безопасности беспроводных сетей, зависит от эффективности применяемого алгоритмического обеспечения. С этой целью были разработанны соответствующие алгоритмы.

На этапе алгоритмизации необходимо детализировать каждый блок таким образом, чтобы впоследствии облегчить задачи непосредственной реализации

В алгоритмические блоки включаются также те программы, которые управляют работой различных частей компьютера и позволяют пользователю решить поставленную задачи наиболее желаемым образом.

Для автоматизированной системы безопасности необходимо строить алгоритмы и описывать отдельные алгоритмические блоки в соответствии с их функциональными требованиями.

В соответствии с комбинированным алгоритмом шифрования символов происходит шифрование изображений, введенных пользователем, их распознавание и, соответственно, формирование баз данных.

На начальном этапе работы алгоритма происходит определение того, внёс ли пользователь информацию. На первом этапе внесенный пользователем пароль заносится в специальный массив, где определено начало и конец слова. Также происходит разбиение на символы и определение индекса для каждого символа. Далее в систему вводятся дополнительные переменные, по значениям которых определяются индексы дополнительных символов.

Затем происходит разбиение слов на отдельный символы и определение индекса для каждого из них, в результате получается слово, состоящее из объединения символов, в котором для каждого символа определен свой индекс. Из полученных и дополнительных символов получается «усложненное» слово (пароль) к которому добавляется значения текущей даты и времени, затем полученный результат разбивается на составные части (создание групп) согласно параметрам установленным в системе.

Зашифрованная таким образом информация передаётся на сервер по группам, если идентификация первой группы прошла успешно сервер запрашивает вторую группу и т. д. по очереди до завершения всех групп. Если какая-нибудь группа не была успешно идентифицирована, то сервер блокирует данного пользователя.

Список используемых источников

1. Beridze Z. Elaboration of dialogue procedures in safety support automated systems of wireless networks / Z. Beridze, N. Geladze // XXIV international conference problems of decision making under uncertainties (PDMU – 2014). ABSTRACTS. September 1-5, 2014. Cesky Rudolec, Czeck Republic, 2014. – P. 16-17.

ABOUT NEURAL NETWORKS

I. Sh. Didmanidze, G. A. Kakhiani, Z. N. Megrelishvili,
D. Z. Didmanidze
Batumi Shota Rustaveli State University, Batumi, Georgia
ibraimd@mail.ru

Introduction

It is clear that various initialization of synoptic communications can give various versions of the solution of a task. Here an important role is played by number of incidentally initialized communications.

For learning efficiency the architecture of a neural network plays large role. Dimension of entrance and output layers of a neural network depends on a solved task and structure of a set of entrance data.

As it is known a three-layer neural network approximation practically any function with any in advance set accuracy is possible. Thus the accuracy of approximation depends on quantity of neurons in the hidden layer.

Technical Requirements

For the purpose of achievement of necessary accuracy and a community of results possibly application of a neural network with two hidden layers, however it must be kept in mind that time of training of such network of many is more than usually

One of defects of a method of gradient descent is its "jamming" in local minima. For the purpose of overcoming of this defect possibly application of a so-called method of a heavy ball [1], [2].

In this case modification of synoptic communications of a neural network happens according to the following expression

$$\Delta w_{ij}(t+1) = -\alpha\gamma_j F'\left(S_j\right)y_i + \gamma\Delta w_{ij}(t)$$

Where γ –a constant called by moment parameter

Value of parameter of the moment undertakes from range [0,1]. introduction of this parameter allows to overcome rather small local minima.

On the basis above stated it is possible to draw the following conclusions:

1. The neural network with one hidden layer is able to display any entering signals in any proceeding signals;
2. The quantity of neurons in the hidden layer has to be less number of data in training selection;
3. Power of a neural network can be increased both at the expense of increase in neurons in the hidden layer and at the expense of increase in quantity of the hidden layers. if a

network influences #2 and the network isn't able to solve an objective that it is necessary to increase quantity of the hidden layers.

Thus initialization of synoptic communications in a network has to occur in rather narrow range of values

Theorem. Adaptive speed of training of neurons of the penultimate hidden layer can be counted as follows

$$\alpha_2(t) = \frac{\sum_i c_i \sum_j \left(y_j - t_j\right) w_{ij}}{\left(F'(0)\right)^2 \sum_j \left(\sum_i c_i w_{ij}\right)^2}$$

Where

$$c_i = \gamma_i F'(S_i)\left(1 + \sum_k y_k^2\right)$$

Here j number of an output layer of a neural network, i last hidden layer of a network, k penultimate hidden layer

proof:. For i of neuron of the last hidden layer the weighed sum can be written down as follows

$$S_i(t+1) = S_i(t) - \alpha_2 C_i,$$

where

$$c_i = \sum_k y_k \frac{\partial E}{\partial w_{ij}} - \frac{\partial E}{\partial T_i} = \gamma_i F'(S_i)\left(1 + \sum_k y_k^2\right),$$

then at decomposition in a row Taylor for proceeding signals of i of neuron we will have:

$$y_i(t+1) = F(0) + F'(0) S_i(t+1)$$

after transformations we will receive the following expression:

$$y_i(t+1) = y_i(t) - \alpha_2 F'(0) C_i \quad (1)$$

the weighed sum of j element can be presented as follows::

$$S_j(t+1) = \sum_i w_{ij} y_i(t+1) - T_j \quad (2)$$

after substitution (1) in expression (2) we will receive::

$$S_j(t+1) = S_j(t) - \alpha_2 F'(0) \sum_i w_{ij} C_j$$

output value j of an element is defined as follows:

$$y_j(t+1) = y_j(t) + F'(0) S_j(t+1) = y_j(t) - \left(F'(0)\right)^2 \alpha_2 \sum_j w_{ij} C_j$$

then

$$\frac{\partial E}{\partial \alpha_2} = \sum_j y_k - \left(y_j(t) - t_j - \alpha_2 \left(F'(0)\right)^2 \sum_i w_{ij} C_i\right) \times$$

$$\times\left(-\left(F'(0)\right)^2\right)\sum_j w_{ij}C_j$$

after equating of the last expression with zero and expression of $\alpha_2(t)$ we will receive:

$$\alpha_2(t)=\frac{\sum_i c_i\sum_j\left(y_j-t_j\right)w_{ij}}{\left(F'(0)\right)^2\sum_j\left(\sum_i c_i w_{ij}\right)^2}$$

as was to be shown.

The expressions given above allow to estimate the adaptive speed of training of a neural network with one or several hidden layers.

References

1. Didmanidze I. Multilayer neural network / I. Didmanidze, G. Kakhiani, D. Didmanidze // XXIV international conference problems of decision making under uncertainties (PDMU – 2014). ABSTRACTS. September 1-5, 2014. Cesky Rudolec, Czeck Republic, 2014. – P. 32-33.
2. Кратович П.В. Нейронные сети и модели типа ARIMA для прогнозирования котировок / П.В. Кратович // Программные продукты и системы. – 2011. – № 1(93). – С. 95 – 98.

COMPRESSION SCHEMA

I.Sh. Didmanidze, R.D. Tkhilaishvili
Batumi Shota Rustaveli State University, Batumi, Georgia
ibraimd@mail.ru

Let's discus one exemplar of choose of the compression schema. Let's suppose four-digit sequence.

$$v_{ri}^{i} = v_1 v_2 \dots v_i \dots v_r$$

Next transformation v_{ri}^{i} happens by

$$S = v_i v_{i+1} v_{i+2} \dots v_v^m + \frac{m}{2} v_{n+1} v_{n+2} \dots v_{2^n}$$

As result of it we have such sequence

$$S_N = S_1 S_2 \dots S_N$$

Let's introduce the notion of the expansion coefficient - K_N, which characterizes growth of the sequence N S_n^t with variations of the n i.e.

$$K_N = \frac{N(S)_{n>2}}{N(S)_{n=2}}$$

LEMMA 1. When t=1, $N=N_{max}$, n is equal or is close to

$$n = \frac{r+1}{4}$$

PROOF. Since t=1, the amount of block vectors k V_{kn} in the sequence S_n^t will be $k = (r-n)-(n-1) = r-2n+1$, and the amount of letters $N = rn - 2n^2 + n$.

The choice of the optimal notion of n, providing the maximum $N=N_{max}$, must be found $\frac{dN}{dn} = \frac{d}{dn}\left(rn - 2n^2 + n\right) = 0$, when

$$n = \frac{r+1}{4}.$$

THEOREMA. For four-digit sequence $S_N^t = S_1 S_2 \dots S_N$, $N \geq r$ minimal nomination of the N when t=1 is determined as

$$n = n_{\min \sqrt{\frac{N}{2}}}.$$

PROOF . According to the lemma 1 we have

$$N = r \cdot n - 2n^2 + n \text{ or}$$

$$r = \frac{N}{n} + 2n - 1.$$

For definition of providing maximum $N=N_{max}$, let's see $\frac{dr}{dn} = 0$ here

$$n = n_{\min \sqrt{\frac{N}{2}}}.$$

Conclusion 1 . When t=1 two adjacent codding layers of the length n have equal n-1 of equal elements.

As we already have seen algebraic sequence and feature of codding sequences let us to choose and to buy cyclical schema of the compression of data, built from these steps:

1) Function of the conversion $f : B \Rightarrow V$ transmits initial binary sequence $B^i_{r_i} = b_1^0 b_2^0 \ldots b_r^0$ into four-digit sequence $v^i_{r_i} = b_1^0 b_2^0 \ldots b_r^0$;

2) For t=1 sequence v_r^0 is founded according the theorem transmits into four-digit sequence S_r^2 when the length of each codding word n=2.

Note: choice of the notion n=2, i.e. introducing extra amount, when in two adjacent vectors border elements are similar, is due the demand of simple decoding.

3) On the ground of $N=N_{max}$ in S_r^2 defines minimal length r_i binary sequence $B^i_{r_i}$ and corresponding length of codding word n.

4) On the ground of (5) we transmit accounted notions r_i and n sequence S_r^2 in $S_{r_i}^{n-1}$, afterwards we can define concrete type of binary sequence b_1.

So finish first cycle of information compression.

The coefficient of the compression for end of the first cycle is

$$K_{comp}^{1} = \frac{\left(rn - 2n^2 + n\right)^2}{r}.$$

We must say there are a lot of variations concerning the choice of construction of modification of information compression schemas.

References

1. Nanobashvili N. Questions of the optimal compression of the discrete information within four-digit codding system / N. Nanobashvili // Tbilisi University's works. –1980. – T. 212(2).
2. Дидманидзе И.Ш. Представление информации в перестановочной схеме записи и сжатия / И.Ш. Дидманидзе, Р.Д. Тхилаишвили, К. Пинар // Материалы III международной научно-практической интернет конференций `СУЧАСНІ ТЕНДЕНЦІЇ РОЗВИТКУ МАТЕМАТИКИ ТА ІІ ПРИКЛАДНІ АСПЕКТИ – Донецк, 2014. – С. 65-66.

РЕАЛИЗАЦИЯ СЕТЕВОЙ БЕЗОПАСНОСТИ НА ПРИМЕРЕ ВИРТУАЛЬНЫХ СЕТЕЙ

М.В. Донадзе, Г.Т. Имнаишвили, Н.Ш. Сирабидзе
Батумский государственный университет имени Шота Руставели,
Батуми, Грузия
donadzem@gmail.com

Введение

Безопасность телекоммуникационных сетей во многом определяется размерами широковещательных доменов, внутри которых может происходить несанкционированный доступ к конфиденциальной информации. В традиционных сетях деление на широковещательные домены реализуется маршрутизатором.

Основная часть

Виртуальные сети созданы для того, чтобы реализовать сегментацию сети на коммутаторах. Таким образом, создание виртуальных локальных сетей (Virtual Local Area Networks – VLAN), которые представляют собой логическое объединение групп станций сети (рис.1), является одним из основных методов защиты информации в сетях на коммутаторах.

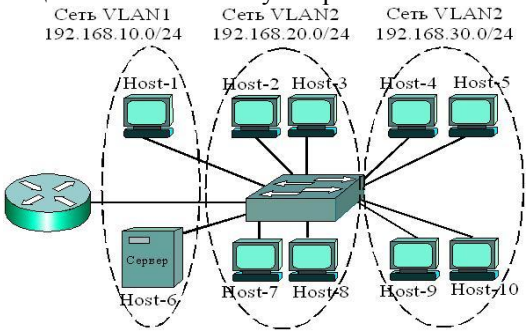


Рис. 1. Виртуальные локальные сети VLAN

Обычно VLAN группируются по функциональным особенностям работы независимо от физического местоположения пользователей. Обмен данными происходит только между устройствами, находящимися в одной VLAN. Обмен данными между различными VLAN производится только через маршрутизаторы.

Рабочая станция в виртуальной сети, например, Host-1 в сети VLAN1 (рис.1), ограничена общением с сервером в той же самой VLAN1. Виртуальные сети логически сегментируют всю сеть на

широковещательные домены так, чтобы пакеты переключались только между портами, которые назначены на ту же самую VLAN (другими словами, приписаны к одной VLAN). Каждая сеть VLAN состоит из узлов, объединенных единственным широковещательным доменом, образованным приписанными к виртуальной сети портами коммутатора.

Поскольку каждая виртуальная сеть представляет широковещательный домен, то маршрутизаторы в топологии сетей VLAN (рис.1) обеспечивают фильтрацию широковещательных передач, безопасность, управление трафиком и связь между VLAN. Коммутаторы не обеспечивают трафик между VLAN, поскольку это нарушает целостность широковещательного домена VLAN. Трафик между VLAN обеспечивается маршрутизацией, то есть общение между узлами разных виртуальных сетей происходит только через маршрутизатор.

Каждой виртуальной сети при конфигурировании должен быть назначен IP-адрес сети или подсети с соответствующей маской, для того, чтобы виртуальные сети могли общаться между собой.

Таким образом, сеть VLAN является широковещательным доменом, созданным одним или более коммутаторами. На рис. 2 три виртуальных сети VLAN созданы одним маршрутизатором и тремя коммутаторами. При этом существуют три отдельных широковещательных домена (сеть VLAN 1, сеть VLAN 2, сеть VLAN 3). Маршрутизатор управляет трафиком между сетями VLAN, используя маршрутизацию уровня 3.

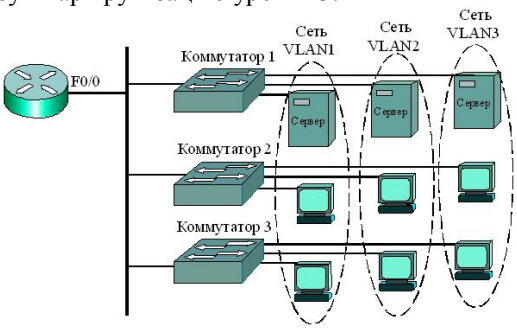


Рис. 2. Три виртуальных сети VLAN

При построении сети на нескольких коммутаторах необходимо выделить дополнительные порты для объединения портов разных коммутаторов, приписанных к одноименным виртуальным сетям (рис.3). Дополнительных пар портов двух коммутаторов должно быть выделено столько, сколько создано сетей VLAN.

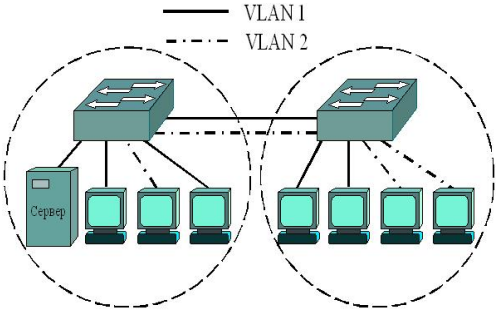


Рис. 3. Объединение виртуальных сетей двух коммутаторов

Поскольку кадры данных могут быть получены коммутатором от любого устройства, присоединенного к любой виртуальной сети, при обмене данными между коммутаторами в заголовок кадра добавляется уникальный идентификатор кадра – тег (tag) виртуальной сети, который определяет VLAN каждого пакета. Стандарт IEEE 802.1Q предусматривает введение поля меток в заголовок кадра, содержащего два байта (табл.1).

Таблица 1. Формат тега виртуальной сети

3 бита	1 бит	12 бит
Приоритет	CFI	VLAN ID

12 двоичных разрядов используются для адресации, что позволяет помечать до 4096 виртуальных сетей и соответствует нормальному и расширенному диапазону идентификаторов VLAN. Еще три разряда этого поля позволяют задавать 8 уровней приоритета передаваемых сообщений, то есть позволяют обеспечивать качество (QoS) передаваемых данных. Наивысший приоритет уровня 7 имеют кадры управления сетью, уровень 6 – кадры передачи голосового трафика, 5 – передача видео. Остальные уровни обеспечивают передачу данных с разным приоритетом. Единичное значение поля CFI показывает, что виртуальная сеть является Token Ring.

Пакет отправляется коммутатором или маршрутизатором, базируясь на идентификаторе VLAN и MAC-адресе. После достижения сети назначения идентификатор VLAN (tag) удаляется из пакета коммутатором, а пакет отправляется присоединенному устройству. Маркировка пакета (Packet tagging) обеспечивает механизм управления потоком данных.

Согласно принципу, представленному на рис. 3, в виртуальных локальных сетях для соединения нескольких коммутаторов между собой задействуют несколько физических портов. Совокупность физических каналов между двумя

устройствами может быть заменена одним агрегированным логическим каналом, получившим название транк (trunk).

Транковые соединения используются и для подключения маршрутизатора к коммутатору (рис.2). При этом на интерфейсе маршрутизатора формируются несколько субинтерфейсов (по количеству виртуальных сетей).

Пропускная способность агрегированного логического канала равна сумме пропускных способностей физических каналов. Транки применяют и для подключения высокоскоростных серверов.

На практике используются статические и динамические VLAN. Динамические VLAN создаются через программное обеспечение управления сети. Однако динамические VLAN не применяются широко.

Наибольшее распространение получили статические VLAN. Входящие в сеть устройства автоматически становятся членами VLAN-порта, к которому они присоединены. Для статического конфигурирования используется интерфейс командной линии CLI.

Выводы

Существующая в настоящее время система защиты информации при передаче по сети не в полной мере удовлетворяет требованиям по предотвращению несанкционированного доступа. Поэтому происходит постоянный поиск путей и способов создания конкретных аппаратно-программных средств и комплексов в рамках системы обеспечения информационной безопасности. Знание и учет особенностей построения и функционирования сети, конкретных стандартов и протоколов является основой успешного решения этой задачи.

Список используемых источников

1. Васин Н.Н. Построение сетей на базе коммутаторов и маршрутизаторов / Н.Н. Васин. – Москва: Интуит, 2011. – 388 с.

ВИКОРИСТАННЯ АВТОМАТИЗОВАНИХ ЗАСОБІВ ГЕНЕРАЦІЇ ПРОГРАМ ДЛЯ ГРАФІЧНИХ ПРИСКОРЮВАЧІВ У ЗАДАЧАХ МЕТЕОРОЛОГІЧНОГО ПРОГНОЗУВАННЯ

А.Ю. Дорошенко, О.А. Яценко, О.Г. Бекетов
Інститут програмних систем НАН України, Київ, Україна
doroshenkoanatoliy2@gmail.com, oayat@ukr.net, beketov.oleksii@gmail.com

Вступ

Регіональні та локальні моделі, що описують мезомасштабні та мікромасштабні метеорологічні процеси та явища, визначають формування погоди в окремих пунктах та регіонах, призначені для створення більш обґрунтованих та деталізованих як у просторі, так і у часі, прогнозів погоди. Математична модель виражається рівняннями руху Рейнольдса для турбулентних процесів, що зв'язують прискорення в певному напрямку з компонентами об'ємних і поверхневих сил, які діють у цьому напрямку, рівнянням збереження маси, термодинамічним рівнянням енергії і рівнянням стану Бойля-Шарля. Реалізація прогностичних моделей відбувається за допомогою чисельних методів. Побудовані в даній роботі алгоритми та програми прогнозування погоди ґрунтуються на регіональній математичній моделі стану атмосфери, що наведена в роботі [1]. Ядром такої моделі є рівняння конвективної дифузії, що розглянуті нижче.

Засоби автоматизації проектування паралельних програм

Для проектування послідовної та паралельної програм прогнозування було застосовано апарат алгебри алгоритмів на основі використання інструментарію проектування та синтезу програм (ІПС) [2]. В основу інструментарію покладено поняття систем алгоритмічних алгебр (САА) В.М. Глушкова. На САА ґрунтується алгоритмічна мова САА/1, що використовується для проектування програм в ІПС. Згадана мова призначена для багаторівневого структурного проектування і документування послідовних і паралельних алгоритмів і програм. Перевагою її використання є можливість опису алгоритмів у природно-лінгвістичній формі, яка є зручною для людини, це полегшує досягнення необхідної якості програм. Основними об'єктами мови САА/1 є абстракції операторів (перетворювачів даних) і умов (предикатів). Оператори й умови можуть бути базисними або складеними. Базисний оператор (умова) – це оператор (умова), що в САА-схемі вважається первинною атомарною абстракцією. В

паралельній схемі використовуються операції САА, що орієнтовані на формалізацію обчислень в CUDA-програмах, вони були запропоновані в роботі [2]: операції взаємодії із відеопам'яттю, запуску ядра та синхронізації потоків.

Задача конвективної дифузії

Рівняння Навьє-Стокса та тепло-, масо переносу, що складають основу сучасних метеорологічних моделей, є нелінійними тривимірними рівняннями конвективної дифузії:

$$\frac{\partial u}{\partial t} + \Lambda u = f\ ,\ (x_1, x_2, x_3) \in \Omega / \Gamma\ , t > 0\ , \qquad (1)$$

$$u(0, x_1, x_2, x_3) = u^0(x_1, x_2, x_3)\ ,\ (x_1, x_2, x_3) \in \Omega\ , \qquad (2)$$

$$u(t, x_1, x_2, x_3) = 0\ , \qquad (x_1, x_2, x_3) \in \Gamma\ ,\ t > 0\ , \qquad (3)$$

де $\Omega = [0, \ell_1] \times [0, \ell_2] \times [0, \ell_3]$ – просторова область визначення задачі, Γ – границя області Ω, $u = u(t, x_1, x_2, x_3)$ – залежна функція, $f = f(t, x_1, x_2, x_3)$ – вільний член рівняння, $\Lambda = \sum_{\alpha=1}^{3} \Lambda_\alpha$ – просторовий диференціальний оператор, що подається через суму більш простих операторів:

$$\Lambda_\alpha = v_\alpha \frac{\partial}{\partial x_\alpha} - \frac{\partial}{\partial x_\alpha}\left(\mu_\alpha \frac{\partial}{\partial x_\alpha} \right),$$

$$v_\alpha = v_\alpha(x_1, x_2, x_3),\ \ \mu_\alpha = \mu_\alpha(x_1, x_2, x_3) > 0\ .$$

Реалізація та експерименти

Розв'язання задачі (1)–(3) проводилось за допомогою адитивно-усередненого методу розщеплення [3] та методу явного підрахунку [4]. Застосовано алгоритм трирівневого паралелізму (модифікований адитивно-усереднений метод (МАУМ)), який представлено в [5]. Для проведення чисельного експерименту було розлянуто частковий випадок задачі (1)–(3) з відомим розв'язком. Було розроблено реалізацію наведеного підходу для архітектури відеографічного прискорювача засобами CUDA та OpenMP.

Створена реалізація МАУМ використовує рівномірну декомпозицію області Ω сіткою з кроком h розбиття часового проміжку з кроком τ. Розв'язок отримується в кожній точці сітки, і порівнюється із точним роз'язком в кінцевий момент часу.

Експеримент проводився із використанням графічного прискорювача *NVIDIA GeForce GTX 650 Ti (768 CUDA-ядер, базова частота 928MHz, об'єм глобальної пам'яті 1024Mb)* та процесора *Intel Core i5-3570 (4 ядра, базова частота 3.40GHz, 64-бітний набір інструкцій)* у 64-бітному форматі подання чисел з плаваючою крапкою.

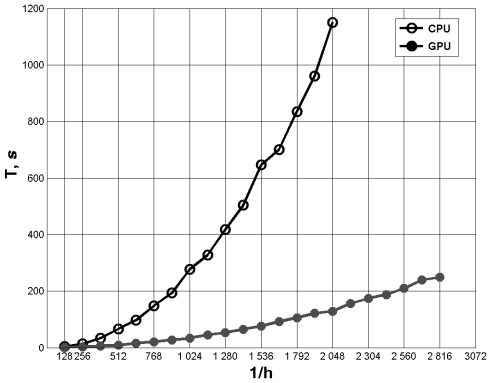


Рис. 1. Час розв'язання часткового випадку задачі (1)–(3) з використанням GPU та одного ядра CPU в залежності від розміру розбиття області при фіксованому кроку по часу

На поданому графіку зображена залежність часу виконання від розміру просторової сітки зі сталим часовим кроком для GPU та одного ядра CPU.

На основі побудованих САА-схем алгоритмів прогнозування для моделі, наведеної в роботі [1], було реалізовано послідовну та паралельну програми мовою C++ для виконання на CPU та GPU відповідно. Експеримент з виконання паралельної програми на відеографічному прискорювачі показав значне підвищення продуктивності обчислень порівняно із послідовною програмою (досягнуто 68-кратне мультипроцесорне прискорення із залученням згаданої вище експериментальної бази).

Далі наведено частину паралельної САА-схеми обчислень.

```
"Паралельне обчислення правих частин Q для (U)"
==== Запуск_Ядра(blocksPerGrid_3d, threadsPerBlock_3d)(
        "Виконати передрозрахунок для першої похідної (Pr1)";);
     Запуск_Ядра(blocksPerGrid_1d, threadsPerBlock_1d)(
        "Виконати обчислення першої похідної (Pr1)";);
     "Заповнити масив для Q у відеопам'яті значеннями (0)";
     Запуск_Ядра(blocksPerGrid_3d, threadsPerBlock_3d)(
        "Обчислити праві частини Q для (U) на Cuda";);
     "Занести в масив (RZ) значення для функції (U)";
     "Обчислити складові рівняння переносу в напрямку X на Cuda для (U)";
     "Обчислити складові рівняння переносу в напрямку Y на Cuda для (U)";
     "Обчислити складові рівняння переносу в напрямку Z на Cuda для (U)";
     "Скопіювати масив значень Q з відеопам'яті у пам'ять CPU";
```

```
      "Занести значення з лінійного масиву для Q у 4-вимірний масив";
      "Записати результуючі дані у файли для (U)";

"Обчислити складові рівняння переносу в напрямку X на Cuda для (RZ)"
==== "Оголошення та ініціалізація змінних";
      Запуск_Ядра(blocksPerGrid_3d, threadsPerBlock_3d)(
         "Виконати передрозрахунок для першої похідної (Pr1)";);
      Запуск_Ядра(blocksPerGrid_1d, threadsPerBlock_1d)(
         "Виконати обчислення першої похідної (Pr1)";);
      Запуск_Ядра(blocksPerGrid_3d, threadsPerBlock_3d)(
         "Виконати передрозрахунок для першої похідної (Pr2)";);
      Запуск_Ядра(blocksPerGrid_1d, threadsPerBlock_1d)(
         "Виконати обчислення першої похідної (Pr2)";);
      Запуск_Ядра(blocksPerGrid_3d, threadsPerBlock_3d)(
         "Виконати підсумовування (Pr1) та (Pr2)";);
      ЧЕКАТИ `Обробка у всіх потоках закінчена';
            "Звільнення зарезервованої пам'яті для змінних";
```

Висновки

Згідно отриманих даних можна зробити висновок, що запропонований підхід є ефективним при розв'язанні тестової задачі. Застосування розпаралелювання на відеокарті дає відчутне зменшення часу розв'язання задачі, дозволяє зменшувати часові й просторові кроки, тим самим покращувати точність розв'язку.

Оскільки алгоритм розпаралелювання для тестової задачі аналогічний алгоритму розв'язку складних рівнянь, що складають прогностичні метеорологічні моделі, можна стверджувати, що запропонований підхід є ефективним для реалізації останніх. Таким чином, паралельні обчислення на відеографічних процесорах та засоби CUDA доцільно застосовувати при реалізації математичних моделей циркуляції атмосфери.

Список використаних джерел

1. Прусов В. Моделювання природних і техногенних процесів в атмосфері / В. Прусов, А. Дорошенко. – К. : Наукова думка, 2006. – 542 с.
2. Формалізоване проектування та синтез паралельних програм для відеографічних прискорювачів / А.Ю. Дорошенко, О.Г. Бекетов, К.А. Жереб, О.А. Яценко // Проблеми програмування. – 2013. – № 3. – С. 38–46.
3. О моделировании третьей краевой задачи для многомерных параболических уравнений в произвольной области одномерными уравнениями / Д.Г. Гордезиани, Г.В. Меладзе // Ж. вычисл. матем. и матем. физ. – 1974. – Т. 14, № 1. – С. 246–250.

4. Эффективная разностная схема численного решения задачи конвективной диффузии / В.А. Прусов, А.Е. Дорошенко, Р.Н. Черныш, Л.Н. Гук // Кибернетика и сист. анализ. – 2007. – № 3. – С. 64–74.
5. Черниш Р.І. Модифіковане адитивно-усереднене розщеплення, його паралельна реалізація та застосування до задач метеорології : дис. ... канд. фіз.-мат. наук : 01.05.02 / Черниш Р.І. КНУ імені Тараса Шевченка. – К., 2010. – 150 с.

ПРИНЦИПЫ КООРДИНАЦИИ ИНТЕЛЛЕКТУАЛЬНЫХ АГЕНТОВ НА ОСНОВЕ НЕЧЕТКОЙ ЛОГИКИ

С.В. Ершов
Институт кибернетики имени В.М. Глушкова НАН Украины,
Киев, Украина
sershv@ukr.net

Введение

Интеллектуальные мультиагентные системы состоят из множества агентов – автономных программ, которые способны решать задачи либо самостоятельно, либо сотрудничая с другими агентами. Модели агентов с сильными интерактивными возможностями (коммуникация, сотрудничество и т.д.) могут быть использованы в качестве основных компонентов для проектирования сложных мультиагентных систем [1].

Из-за неопределенности характера взаимодействия и сотрудничества агентов при решении практических задач необходимо использование интеллектуальных агентов с нечеткой логикой [2-4], реализующих распределенную деятельность сложной мультиагентной системы. Интеллектуальные агенты могут вырабатывать более адекватные решения относительно своих дальнейших действий с учетом нечеткой информации в процессах на основе взаимодействия [5, 6].

Очень часто границы “информационных гранул”, представляемых нечеткими множествами, плохо различимы, выражены некоторыми зонами. В этом случае их можно выразить нечеткими множествами высшего типа. Так у нечетких множеств типа 2 отдельные значения принадлежности задаются функциями принадлежности, т.е. учитывается “размытость” определения принадлежности. Они выражают субъективную и лингвистическую неопределенность, связанную с различными оттенками смысла информационных гранул. При реальном взаимодействии

интеллектуальных агентов передаваемые значения (гранулы) имеют неточный, “размытый” смысл.

Уместно отметить, что создание агентов, использующих преимущества нечеткой логики высшего типа (в частности, типа 2), требует разработки методов координации и взаимодействия нечетких агентов и архитектурных моделей таких агентов, учитывающих особенности представления нечеткой информации. Это важно, например, при решении таких прикладных задач, как исследование эффективности процессов эвакуации людей в чрезвычайных ситуациях, поведения транспортных средств в условиях неопределенности (неполной информации) и ряда других задач.

Взаимодействие между нечеткими агентами

Взаимодействие $i \in I$ между двумя нечеткими агентами определяется кортежем $i = \langle r_s, r_t, \gamma \rangle$, где r_s – роль агента-источника взаимодействия, r_t – роль агента назначения взаимодействия, γ – сообщение, содержащее нечеткое значение. Целевой агент всегда оценивает нечеткое значение, содержащееся в сообщении, чтобы определить степень интереса, которую может для него представлять участие в этом взаимодействии.

С целью обмена информацией, запроса услуг, переговоров относительно нечетких значений агенты выражают свои намерения в соответствии с языковыми средствами теории речевых актов [7]. Основных речевых актов достаточно для того, чтобы нечеткие агенты воспринимали намерения о сотрудничестве, связанном с предложением, задаваемым сообщением. Такое взаимодействие задается протоколом, при котором для некоторых взаимодействий требуется формирование агентом ответного сообщения (рис. 1).

Коммуникация между двумя нечеткими агентами определяется в виде $\lambda_{s,t} = \langle \lambda, r_s, r_t, \tau, \overline{\mu}, \underline{\mu} \rangle$, что можно представить как $i = \langle r_s, r_t, \gamma \rangle$ и $\gamma = \langle \lambda_{s,t}, \tau, \eta \rangle$, где $\lambda_{s,t} \in \Gamma_{ri}$ – множество действий, которые нечеткий агент роли r_i может выполнить, $\lambda \in \Lambda$ – речевой акт, обозначаемый перформативным глаголом, r_s – нечеткий агент-источник коммуникации, r_t – нечеткий агент-получатель, $\tau \in T$ – это тип сообщения, $\langle \overline{\mu}, \underline{\mu} \rangle$ – нечеткое значение, представляющее собой нечеткое множество типа 2, которое может быть значением утверждения, вопроса,

ответа и т.д. Например, при получении сообщения, соответствующего речевому акту "информировать" $\lambda_{s,t} = \langle inform, r_s, r_t, \tau = FS2, \overline{\mu}, \underline{\mu} \rangle$ агент может выделить содержащееся в нем значение нечеткого множества типа 2: $(\overline{\mu}(0; 2; 4; 1), \underline{\mu}(1; 2; 3; 0{,}8))$.

Каждый нечеткий агент играет одну или несколько предустановленных ролей, границы которой определены его компетенцией в рамках мультиагентной системы. Поведение агента, соответствующее таким ролям, формализуется его нечеткими правилами принятия решений. Нечеткие правила принятия решений Δ_{a_i} нечеткого агента роли r_i, представляющие

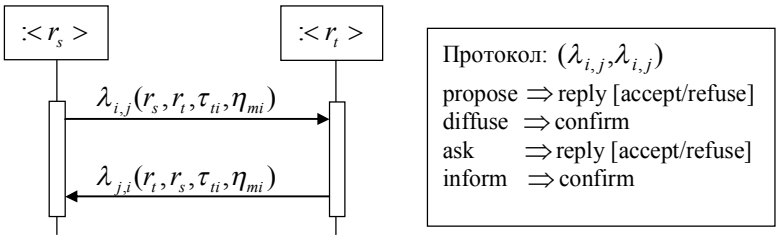


Рис. 1. Коммуникационный протокол между нечеткими агентами

его совокупную базу знаний, задаются в виде $\Delta_{ri} = \langle E_{ri}, X_{ri}, \Gamma_{ri} \rangle$, где E_{ri} – множество нечетких значений, соответствующих событиям, на которые агент r_i может реагировать, X_{r_i} – множество нечетких значений, связанных с внутренними состояниями агента r_i и Γ_{r_i} – множество действий, которые агент роли r_i может выполнять. В общем случае такими действиями могут быть посылки сообщений другому агенту, содержащие значение речевого акта и нечеткое значение, вырабатываемое системой нечетких правил.

Принципы координации при построении архитектуры нечетких мультиагентных систем

Проблемы, возникающие из-за частичного знания агентами своей среды, требуют разработки сложных координационных механизмов [1-5]. Архитектура мультиагентной системы контролирует и координирует взаимодействие между агентами системы, тем самым структурируя их виды деятельности с целью

достижения требуемого поведения. В общем случае архитектура мультиагентной системы рассматривается как совокупность ролей, которые состоят в определенных отношениях друг с другом, и которые принимают участие в систематических формах координирующего взаимодействия с другими ролями.

Архитектура мультиагентной системы задает совокупность агентов, поведение и функции которых определены нечеткими правилами. Такая совокупность может быть разделена на коалиции агентов, имеющих общие цели и характеристики. Чтобы выполнять свою роль, агент может взаимодействовать с агентами своей коалиции или других коалиций. Таким образом, роль является абстрактным представлением функции, выполняемой агентом в коалиции.

При динамической архитектуре роли нечетких агентов могут стать динамическими и определяться совокупностью $R = \{r_1, r_2, ..., r_n\}$. Тогда нечеткое множество ролей, которые выполняет агент ag, определяется выражением $R(ag) = \{\mu_{r_1}(ag), \mu_{r_2}(ag), ..., \mu_{r_n}(ag)\}$.

Пусть $\Omega_R : Ag \to \{R\}$ – функция "играть роль", тогда роли $r_j \in R$, которые играет нечеткий агент ag, задаются как $\Omega_R : (ag_i, r_j) \to \{\mu_{r_j}(ag_i)\}$, где $j \in J_R$, $J_R = \{1,2,...,n\}$ – множество всех ролей. Архитектура мультиагентной системы нечеткая и динамическая, поскольку распределение ролей, выполняемых нечеткими агентами, постоянно изменяется. Нечеткую систему на основе агентов можно разделить на множество коалиций следующим образом: $Ag_1 \subseteq Ag$,..., $Ag_i \subseteq Ag$, где i - число коалиций. Для каждой коалиции может быть определена основная роль, которую нечеткие агенты будут выполнять в этом сообществе. Считается, что каждый агент принадлежит к коалиции, в которой он играет свою основную роль: $\forall ag \in Ag \exists x \, (ag \in Ag_x \wedge \max(\Omega_R(ag, r_x))$.

Нечеткие агенты взаимодействуют, посылая сообщения агентам своей коалиции (как правило, они предназначены для выполнения их главной роли). Они также координируют работу нечетких агентов из других коалиций, и в этом случае они участвуют в других ролях. Если нечеткий агент ag_i осуществляет координирующее взаимодействие с агентом ag_j из другой

коалиции, то он участвует в выполнении его роли r_j:

$$\forall ag_i \in A \exists x(r_x \in R \wedge ag_j \in A_x \wedge \max(\Omega_R : (ag_j, r_x)) \wedge$$
$$\lambda_{i,j}(r_i, r_j, \tau, \overline{\mu}, \underline{\mu}) \Rightarrow \max(\Omega_R : (ag_i, r_x)))$$

Иерархическая нечеткая мультиагентная система представляет собой подробное представление отдельного нечеткого агента с целью упрощения нечетких баз правил с явным учетом отдельных агентов и координации между ними. Кроме того, дополнительная эффективность координации достигается за счет возможности лучшего представления одновременного воздействия уменьшенного количества входов на отдельные (упрощенные) базы правил. Однако, при этом теряется точность из-за накопления ошибок в результате повторяющегося применения фузификации, нечеткого вывода, редукции типа и дефузификации в течение нескольких циклов работы такой системы. Для повышения точности редукция типа с дефузификацией и дальнейшей фузификацией промежуточных переменных могут отсутствовать, нечеткое значение, представляющее собой нечеткое множество типа 2 непосредственно передается между агентами.

Обобщением такой архитектуры координации мультиагентных систем является ее представление как виртуальной сетки (grid). Каждый агент представлен вершиной сетки в то время как координирующие взаимодействия между агентами – соединениями между этими вершинами. Архитектура мультиагентной системы с pq вершинами $\{N_{11}\ldots N_{p1}\},\ldots,\{N_{1q}\ldots N_{pq}\}$, pq входами вершин $\{x_{11}\ldots x_{p1}\},\ldots,\{x_{1q}\ldots x_{pq}\}$, принимающими лингвистические значения из любых допустимых входных множеств, pq выходами вершин $\{y_{11}\ldots y_{p1}\},\ldots,\{y_{1q}\ldots y_{pq}\}$, принимающими лингвистические значения из любых допустимых множеств выходов, и состоящая из p горизонтальных уровней и q вертикальных ярусов (слоев), может быть описана формулой (1).

Ярус 1 …… Ярус q (1)
Уровень 1 $N_{11}(x_{11}, y_{11})$…….$N_{1q}(x_{1q}, y_{1q})$
…….
Уровень p $N_{p1}(x_{p1}, y_{p1})$…….$N_{pq}(x_{pq}, y_{pq})$

Структура сетки в формуле (1) определяет местоположение агентов, а также их входы и выходы. В этом случае, каждый вход и выход может быть как скаляром, так и вектором, представляющим набор нечетких значений типа 2. Уровни в этой структуре сетки представляют собой иерархии узлов в плане субординации в

пространстве, тогда как ярусы задают временную зависимость с точки зрения последовательности выполнения. Однако, формула (1) не дает никакой информации о соединениях между вершинами в виртуальной сетке. Такая информация содержится в структуре соединений в формуле (2), согласно которой $p(q\text{-}1)$ соединений узлов $\{z_{11,12}\ldots z_{p1,p2}\},\ldots,\{z_{1q-1,1q}\ldots z_{pq-1,pq}\}$ принимают лингвистические значения из допустимых множеств для соответствующих выходов и входов узла.

Ярус 1 ……... Ярус q-1 (2)

Уровень 1 $z_{11,12}=y_{11}=x_{12}\ldots\ldots\ldots z_{1q-1,1q}=y_{1q-1}=x_{1q}$

……

Уровень p $z_{p1,p2}=y_{p1}=x_{p2}\ldots\ldots\ldots z_{pq-1,pq}=y_{pq-1}=x_{pq}$

Для единообразия, мультиагентные системы, описываемые выражениями (1) и (2) предполагают наличие узлов в каждой ячейке базовой структуры сетки. Однако, не все ячейки в модели должны быть заполнены агентами, а соединения могут быть между узлами одного уровня или в соседних ярусами. Тем не менее, некоторые взаимодействия в мультиагентной системе могут быть между агентами на разных уровнях или агентами несоседних ярусов.

Выводы

Предложен метод взаимодействия нечетких агентов, базирующийся на значениях “речевых актов”, позволяющий обрабатывать намерения о сотрудничестве, связанном с предложением, сопровождаемым нечетким значением типа 2. Разработана ролевая модель координации поведения нечетких агентов, основанная на протоколах обмена сообщениями, задающая функциональность агентов в рамках мультиагентной системы. Разработанные методы координации позволяют моделировать и проектировать сложные, распределенные мультиагентные системы, основанные на методологии “вычислений со словами”. Они позволяют сделать представление знаний и моделирования рассуждений в мультиагентных системах более наглядным, выводя словесные заключения из посылок, выраженных средствами натурального языка.

Список используемых источников

1. Wooldridge M. An Introduction to Multi-agent Systems / M. Wooldridge. – Cambridge: MIT Press, 2002. – 366 p.
2. Парасюк И.Н. Нечеткие модели мультиагентных систем в распределенной среде / И. Н. Парасюк, С. В. Ершов // Проблемы программирования. – 2010. – № 2–3. – С. 330–339.

3. Парасюк И.Н. Моделе-ориентированная архитектура нечетких мультиагентных систем / И. Н. Парасюк, С. В. Ершов // Компьютерная математика. – 2010. – № 2. – С.139–149.
4. Ершов С.В. Принципы построения нечетких мультиагентных систем в распределенной среде / С. В. Ершов // Компьютерная математика. – 2009. – № 2. – С. 54–61.
5. Парасюк И.Н. Мультиагентные модели на основе нечеткой логики высшего типа для высокопроизводительной среды / И. Н. Парасюк, С. В. Ершов // Проблеми програмування. – 2012. – № 2–3. – С. 260–269.
6. Парасюк И.Н. Трансформационный подход к разработке интеллектуальных агентов на основе нечетких моделей / И. Н. Парасюк, С. В. Ершов // Проблемы программирования. – 2011. – № 2. – С. 62–78.
7. Bellifemine F. Developing Multi-Agent Systems with JADE / F. Bellifemine, G. Caire, D. Greenwood. – Chichester: John Wiley & Sons Inc., 2007. – 303 p.

МОДЕЛЬ ДАННЫХ С УНИВЕРСАЛЬНОЙ ФИКСИРОВАННОЙ СТРУКТУРОЙ

В.И. Есин
Харьковский национальный университет имени В.Н. Каразина
v.i.yesin@karazin.ua

Введение. Постановка проблемы

Известно, что разработка логической схемы базы данных (БД) является достаточно сложным технологическим процессом, хотя многие организации, отдельные проектировщики его недооценивают. Разрабатывая БД, они в незначительной степени полагаются на какие-либо методологии. Их действия в большей мере определяются опытом их предыдущей работы, осведомленностью, имеющимися «под рукой» средствами и некоторыми другими субъективными факторами. Именно это обстоятельство часто считается основной причиной неудач при разработке баз данных информационных систем (ИС). Но если даже воспользоваться существующими традиционными технологиями проектирования БД в том случае, если корпорация, предприятие или любая другая организация занимается несколькими видами деятельности, и ей необходимо хранить и обрабатывать большие объемы структурированных и слабо структурированных данных из разных существенно отличающихся предметных областей (к тому же вначале может быть неизвестно, какие из этих данных могут быть востребованы в будущем), то, следуя ее правилам, потребуется: уникальная разработка нескольких различных моделей данных (для каждой БД) с соответствующей на нее документацией и возможно различный программный инструментарий проектировщика. На это будут затрачены значительное время и средства, расход которых при усложнении моделируемых видов деятельности будет только увеличиваться.

Подход к решению проблемы

Оценив недостатки присущие такому подходу, было принято решение о поиске возможной универсальной (стандартной) для различных предметных областей (ПрО) модели данных, то есть инструмента моделирования логической модели ПрО (логической схемы БД), который позволял бы представлять данные различных предметных областей в стандартном (универсальном, общем, однообразным с точки зрения структуры конечного представления) и понятном (простом), как профессионалам в области разработки информационных систем, так и обычным пользователям виде.

Разработка универсальной (стандартной) модели данных в представляемом далее виде стала во многом возможной благодаря семантической модели данных «объект-событие» [1,2,3].

В соответствии с множествами отношений и функций модели «объект-событие» были уточнены и формализованы в виде соответствующих *n*-арных математических отношений все базовые понятия модели «объект-событие» («класс объектов», «класс событий», «экземпляр объектов», «тип характеристики объектов», «тип характеристики событий», «документ» и т.д.).

Так математическое отношение, содержащее данные о классах объектов и их иерархий для всех рассматриваемых предметных областей, приняло следующий вид:

$$C = \{(c_1, c_2, c_3, m_1) \mid c_1 \in C_1 \,\&\, c_2 \in C_2 \,\&\, c_3 \in (C_1 \bigcup \{null\}) \,\&\, m_1 \in M_1\}, \quad (1)$$

где C_1 – множество условных идентификаторов классов объектов; C_2 – множество условных имен классов объектов, причем $|C_1| \geq |C_2|$; M_1 – множество условных идентификаторов предметных областей; элемент кортежа c_3 –характеризует иерархию классов объектов.

Математическое отношение, содержащее данные об экземплярах объектов и их иерархии для всех типов объектов, приняло следующий вид:

$$O = \{(o_1, o_2, o_3, r_1, t_1) \mid o_1 \in O_1 \,\&\, o_2 \in O_2 \,\&\, o_3 \in (O_1 \bigcup \{null\}) \,\&\, r_1 \in R_1 \,\&\, t_1 \in T_1\}, \quad (2)$$

где O_1 – множество условных идентификаторов экземпляров объектов; O_2 – множество условных имен экземпляров объектов; T_1 – множество условных идентификаторов типов объектов; R_1 – множество условных идентификаторов разделов.

Математическое отношение, содержащее данные обо всех экземплярах событий определенного класса, которые произошло (происходит, будет происходить) с конкретными экземплярами объектов в определенный момент (интервал) времени, и их иерархии, приняло следующий вид:

$$B = \{(b_1, b_2, b_3, b_4, b_5, o_1) \mid b_1 \in B_1 \,\&\, b_2 \in B_2 \,\&\, b_3 \in B_3 \,\&\, b_4 \in B_4 \,\&\, b_5 \in (B_1 \bigcup \{null\}) \,\&\, o_1 \in O_1\}, \quad (3)$$

где B_1 – множество условных идентификаторов экземпляров событий; B_2 – множество условных имен экземпляров событий; B_3 – множество времен начала событий; B_4 – множество времен окончания событий ($|B_4| \leq |B_3|$).

Аналогичным образом, были уточнены и формализованы в виде соответствующих *n*-арных математических отношений все оставшиеся базовые понятия модели «объект-событие».

Такие математические отношения составили основу для дальнейшей разработки однообразной конечной структуры представления данных различных предметных областей в рамках создаваемой логической модели данных.

Анализ математических выражений подобных (1) – (3) позволил сделать вывод о том, что универсальную модель данных (УМД) легче всего реализовать в рамках реляционной модели:

$$(\mathrm{Rel(T)}, M_r, P, L), \tag{4}$$

где Rel(T) – реляционная модель данных над бинарным отношением $\mathrm{T} \subseteq \mathrm{N} \times \mathrm{V}$, где N – счетное множество имен атрибутов, элементами которого являются имена всех атрибутов, входящих в отношения УМД, V – счетное множество значений, элементами которого являются значения всех этих атрибутов;
M_r – конечное множество имен математических отношений УМД:
M_r={*name(C),name(T),name(O),name(M),name(R),name(H),…*};
P – предикат (условия) обеспечения целостности данных в УМД;
L – язык манипулирования данными.

Такое представление УМД не противоречит определению модели данных, в котором выделяется три компоненты: структурная, манипуляционная и целостная. Оно полностью с ним согласуется.

Формализовано структура УМД определяется как множество возможных отображений $\chi : M_r \to \mathrm{Rel(T)}$, результатом которого есть конечное множество реляционных отношений различной арности: {R_1, R_2,...,R_{19}}={*C,T,M,R,O,H,...*} (где R_i – *i*-е реляционное отношение), основой которых являются математические отношения подобные (1) – (3).

Для явного указания конкретного типа отношения используется форма записи: $R_i(A_{i1}, A_{i2}, ..., A_{i\alpha_i})$. В результате схема универсальной модели данных определяется как совокупность множества типов отношений:

$$\begin{aligned}
&R_1(A_{11}, A_{12}, ..., A_{1\alpha_1}) = C(c_1, c_2, c_3, m_1),\\
&R_2(A_{21}, A_{22}, ..., A_{2\alpha_2}) = T(t_1, t_2, c_1),\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&R_{19}(A_{19_1}, A_{19_2}, ..., A_{19_{\alpha 19}}) = K(k_1, k_2, j_1, o_1, v_1)
\end{aligned} \tag{5}$$

Множество типов отношений (5) является инвариантом.
Со схемой УМД связан конечный набор имен атрибутов:

$at(БД_{УМД}) = \bigcup_{i=1}^{n} at(R_i)$, где n=19, и конечное множество доменов: $D=\{D_1, D_2, ..., D_m\}$.

Для каждого D_l найдется имя атрибута $A_{i\alpha_j} \in at(БД_{УМД})$ такое, что $D_l = dom\ A_{i\alpha_j}$, где l=1..m, а j=1..n.

Такая структура УМД позволяет осуществлять объектное представление предметной области в рамках реляционной модели, что является ее существенным преимуществом перед другими логическими моделями, создаваемыми традиционными методами в рамках реляционной модели.

Целостность данных в УМД обеспечивается как путем поддержания целостности сущностей и ссылочной целостности (как двух важных правил целостности, которые, по сути, являются ограничениями для всех допустимых состояний логической схемы базы данных), так и с помощью соответствующих ограничений доменов (ограничения на допустимые списочные значения характеристик объектов, событий, параметров объектов и т.д.), а также ограничений, упрощающих организацию, так называемых, корпоративных ограничений целостности (например,

- ограничения на количество экземпляров объектов, входящих в определенный класс объектов (преобразованное выражение (1)):

$$C = \{(c_1, c_2, c_3, c_4, m_1) \mid c_1 \in C_1 \,\&\, c_2 \in C_2 \,\&\, c_3 \in (C_1 \bigcup \{null\}) \,\&\, (c_4 \le \Delta) \,\&\, m_1 \in M_1\},$$

где Δ – значение, ограничивающее количество экземпляров объектов для класса объектов, заданного первым элементом картежа – c_1; c_4 – максимальное число экземпляров объектов, входящих в определенный класс;

- ограничения на события: с одним экземпляром объекта в один и то же момент (интервал) времени может происходить только одно событие одного класса;

- ограничение на максимальное количество характеристик события с одним именем и т. д.

Средством манипулирования данными в УМД (элемент L в четверке (4)) может служить как математический аппарат реляционной алгебры, так и специальный непроцедурный язык модели данных (ЯМД) [4].

Выводы

Разработанная универсальная модель данных – это инструмент моделирования, реализованный в рамках реляционной модели данных, который позволяет избежать создания каждый раз

для новой предметной области новой логической схемы базы данных.

Набор неизменных, фиксированных стандартных (с точки зрения структуры конечного представления данных предметной области) реляционных отношений УМД может быть использован при моделировании данных различных предметных областей. Результатом такого моделирования является логическая схема конкретной разрабатываемой базы данных, представленная в виде экстенсионала фиксированных неизменных отношений (путем явного указания элементов рассматриваемой предметной области).

Список используемых источников

1. Есин В. И. Семантическая модель данных "объект-событие" // Вісник Харківського національного університету. Сер.: Математичне моделювання. Інформаційні технології. Автоматизовані системи управління. – 2010. – № 925. – С. 65-73.
2. Сорока Л. С. Формализованное представление модели данных «объект-событие» / Л. С. Сорока, В. И. Есин // Вісник Академії митної служби України. Сер.: Технічні науки. – № 2(46). – С. 49–62.
3. Модель данных «объект-событие» и ее возможности / В. И. Есин, В. Г. Юрасов // Вестник Воронежского государственного технического университета. – 2014. – Т. 10, № 4. – С. 38–43.
4. Есин В. И. Язык для универсальной модели данных / В. И. Есин, М. В. Есина // Системи обробки інформації. – 2011. – № 5(95) – С.193–197.

МЕТОД ЗАХИСТУ ВІД НЕСАНКЦІОНОВАНОГО ДОСТУПУ ДО ІНФОРМАЦІЇ НА ОСНОВІ БАГАТОФАКТОРНОЇ АВТЕНТИФІКАЦІЇ ТА УМОВИ ЙОГО ВИКОРИСТАННЯ

М.В. Єсіна
Харківський національний університет імені В.Н. Каразіна, Україна
rinayes20@gmail.com

Вступ

На сьогоднішній день у технологічно розвинених державах особлива увага приділяється проблемам кібербезпеки, розроблені та реалізуються доктрини інформаційної безпеки. Основним змістом доктрин є протидія загрозам безпеки, насамперед – захисту від несанкціонованого доступу (НСД) до інформації та інформаційних ресурсів.

Стосовно даної проблематики розроблюється та впроваджується достатня кількість міжнародних стандартів. Одним з таких стандартів є стандарт ISO/IEC 19790, який прийнято і в Україні (ДСТУ ISO/IEC 19790).

Забезпечення безпеки інформації в інформаційно-телекомунікаційній системі (ІТС) є одним з найважливіших завдань в ході її експлуатації. Суттєво важливою послугою з безпеки інформації є послуга доступності. По суті її можна трактувати як психологічно сприйнятні механізми доступу до інформації та ресурсів легальних користувачів та суттєві трудності в отриманні доступу порушників та зловмисників, які роблять спроби несанкціонованого доступу (НСД). Зважаючи на важливість цієї проблеми на міжнародному рівні прийнято ряд стандартів відносно захищенності засобів криптографічного захисту від НСД. В першу чергу до них необхідно віднести стандарт ДСТУ ISO/IEC 19790, який визначає вимоги безпеки для криптографічних модулів. В ньому визначено 4 рівні безпеки, причому безумовною вимогою на 4 рівні є застосування захисту від НСД методів та механізмів багатофакторної автентифікації [1-3]. Аналіз вказаних та інших джерел підтвердив недостатній рівень теоретичної та практичної розробки методів та механізмів багатофакторної автентифікації.

Метою доповіді є аналіз стану та розробка моделей оцінки захищенності від НСД в ІТС при реалізації механізмів засобом застосування багатофакторної автентифікації.

Фактори автентифікації

Багатофакторна автентифікація – це автентифікація з хоча б двома незалежними факторами автентифікації. При її реалізації

можуть використовуватись фактори різної природи [3,4]. До них відносять [2]:

- властивість, яку має суб'єкт (наприклад, біометричні дані);
- знання – інформація, яку має суб'єкт (наприклад, пароль чи пін-код);
- володіння – річ, яку має суб'єкт (наприклад, електронна або магнітна картка, флеш-пам'ять, електронний ключ).

Механізми автентифікації пов'язані із захистом інформації взагалі – повідомлень чи ресурсів, від їх модифікації, підміни чи створення хибних даних. Самі механізми автентифікації можна розглядати як заходи захисту, що призначені для встановлення достовірності передачі повідомлення чи відправника або засобів верифікації санкціонування індивідуума для отримання конкретних категорій інформації. По суті, автентифікація являє собою процедуру чи процес встановлення достовірності твердження, що суб'єкт або об'єкт має заявлені (очікувані) властивості. Механізми автентифікації можна розглядати як методи захисту в ІТС від різних видів обману її користувачів.

До переваг багатофакторної автентифікації можна віднести здатність захистити інформацію, як від внутрішніх загроз, так і від зовнішніх вторгнень.

Певною її слабкістю можна вважати необхідність використання додаткових програмно-апаратних комплексів, засобів зберігання і зчитування даних.

Критерії оцінки захищеності від НСД

Проведений аналіз дозволив визначити, що в якості основних показників захищенності від НСД можна вибрати такі [2]:

1. $P_{НСД}(n)$ – ймовірність НСД у n спробах, де n – кількість спроб отримати НСД;
2. $t_б$ – безпечний час;
3. $\Delta T = T_Д$ – допустимий час НСД;
4. n – кількість спроб, які можливо здійснити.

Модель оцінки захищенності від НСД

Аналіз використання схем багатофакторної автентифікації показує, що в ході їх побудови можуть використовуватися певні механізми з послідовним, паралельним або комбінованим з'єднанням елементів захисту. Її вид та послідовність застосування і перелік факторів у схемі може змінюватися на практиці, вони залежать від призначення системи, що захищається, та вимог, які до неї висуваються [1-3]. У випадку використання в

багатофакторному механізмі автентифікації двох факторів можливі такі комбінації: пароль(і) та ключ(і), пароль(і) і біометрична ознака(и), ключ(і) і біометрична ознака [3]. Комбінована схема багатофакторної автентифікації з факторами «пароль і ключ» наведена на рис. 1.

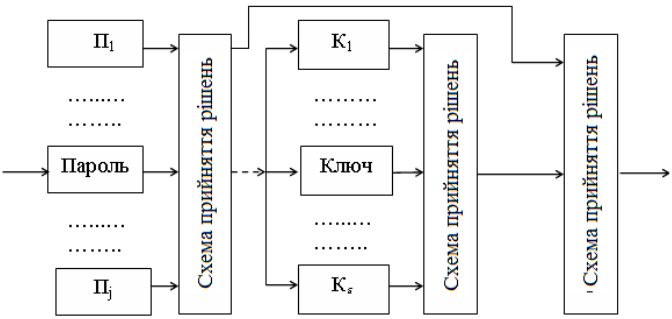


Рис.1. Комбінована схема багатофакторної автентифікації «пароль і ключ».

Таблиця 1. Співвідношення для оцінки $P_{нсд}(P_{зах})$ механізмів комбінованої двохфакторної автентифікації

Механізм пароль/ключ	$P_{НСД} = \left(1 - \prod_{i=1}^{j} P_i^n\right)\left(1 - \prod_{\partial=1}^{\varepsilon} P^к{}_{\partial}\right)$
Механізм пароль/біометрія	$P_{НСД} = \left(1 - \prod_{i=1}^{j} P_i^n\right)\left(1 - \prod_{l=1}^{\mu} P^б{}_{l}\right)$
Механізм ключ/біометрія	$P_{НСД} = \left(1 - \prod_{\partial=1}^{\varepsilon} P^к{}_{\partial}\right)\left(1 - \prod_{l=1}^{\mu} P^б{}_{l}\right)$
Механізм пароль/ключ	$P_{зах} = \prod_{i=1}^{j} P_i^n \cdot \prod_{j=1}^{\varepsilon} P^к{}_{j}$
Механізм пароль/біометрія	$P_{зах} = \prod_{i=1}^{j} P_i^n \cdot \prod_{l=1}^{\mu} P^б{}_{l}$
Механізм ключ/біометрія	$P_{зах} = \prod_{j=1}^{\varepsilon} P^к{}_{j} \cdot \prod_{l=1}^{\mu} P^б{}_{l}$

В таблиці 1 наведено часткові випадки моделей оцінки для трьох варіантів двофакторного механізму. В першому рядку наведено значення, які відповідають моделі, що наведена на рис. 1.

В наведених формулах прийнято такі позначення:

- P^{n} – імовірність правильної роботи механізму паролювання;
- $P^{к}$ – імовірність правильного застосування механізму ключа;
- $P^{б}$ – імовірність правильного застосування механізму біометричних даних.

В таблиці 2 в якості прикладу наведено значення $P_{нсд}$ для системи «пароль+ключ». В якості ключа були використані асиметричні пари, що ґрунтуються на перетвореннях в групі точок еліптичних кривих.

Таблиця 2. Значення $P_{нсд}$ для системи «пароль+ключ»

$P^{к}_{нсд}$ / $P^{n}_{нсд}$	10^{-19}	10^{-32}
$3,9\times10^{-3}$	$3,9\times10^{-22}$	$3,9\times10^{-35}$
$3,73\times10^{-9}$	$3,73\times10^{-28}$	$3,73\times10^{-41}$
$1,778\times10^{-13}$	$1,778\times10^{-32}$	$1,778\times10^{-45}$
$5,42\times10^{-20}$	$5,42\times10^{-39}$	$5,42\times10^{-52}$
$2,94\times10^{-39}$	$2,94\times10^{-58}$	$2,94\times10^{-71}$
$1,59\times10^{-58}$	$1,59\times10^{-77}$	$1,59\times10^{-90}$

Програмна модель механізму автентифікації

Програмну модель для генерування паролів було реалізовано за допомогою пакету прикладного програмування MathCad15.

Було генеровано чотири послідовності: за допомогою вбудованого генератору випадкових чисел, на основі m-послідовності та на основі послідовностей, сформованих за допомогою LLR і LCG. Для всіх отриманих послідовностей були проведені тести на випадковість. Всі послідовності пройшли тести.

Отриманий файл з паролем можна передавати користувачу для експлуатації. При передачі цей файл можна також захистити від зловмисників. Захист можна реалізувати за допомогою шифрування файлу за допомогою будь-якого криптоалгоритму чи використовуючи стеганографічні методи.

Висновки

Стандарт ISO/IEC 19790 встановлює вимоги безпеки для криптографічних модулів, що використовуються в системі безпеки для захисту критичної інформації. Головною вимогою безпеки Рівня Захисту 4 є застосування багатофакторної автентифікації.

В якості основних факторів автентифікації можна використовувати: властивість, яку має суб'єкт; знання – те, що знає суб'єкт; володіння – сутність, яку має суб'єкт.

Для кожного із факторів автентифікації необхідно визначити повний перелік атак зі сторони атакуючих чи потенційних криптоаналітиків (порушників), сформулювати критерії та показники, які дозволили б їх порівняти та обрати такі атаки, які можуть бути реалізовані та забезпечували б досягнення максимальних значень ймовірностей НСД, тобто максимального значення $P_{нсд}$, потім сформулювати пропозиції та рекомендації, в тому числі для застосування у механізмах багатофакторної автентифікації.

Необхідно також відмітити недостатній рівень теоретичної та практичної розробки тематики, що розглянута, та її актуальність.

Список використаних джерел

1. Горбенко І. Д. Прикладна криптологія. Теорія. Практика. Застосування: монографія. / І. Д. Горбенко, Ю. І. Горбенко // Х. : «Форт», 2012. – 870 с.
2. Інформаційні технології – Методи забезпечення безпеки – Вимоги до безпеки для криптографічних модулів (ISO/IEC 19790:2012(E)): ДСТУ ISO/IEC 19790. – [Чинний від 2012-08-15]. – К. : Держспоживстандарт України, 2012. – 98 с. – (Національні стандарти України).
3. Єсіна М. В. Багатофакторна автентифікація: використання механізмів двофакторної автентифікації для захисту від несанкціонованого доступу / М. В. Єсіна, І. Д. Горбенко // Компьютерное моделирование в наукоемких технологиях (КМНТ-2014): Труды науч.-техн. конф. с международным участием, 28-31 мая 2014 г., г. Харьков: Харьковский национальный университет им. В.Н. Каразина, 2014. – С. 159–162.
4. Симонс Г. Д. Обзор методов аутентификации информации. / Г. Д. Симонс. – Т. 76, №5. – М. : ТИИЭР, 1988. – С. 105-125.

ПРИМЕНЕНИЕ АДАПТИРОВАННЫХ СТРУКТУР ДАННЫХ В ПРОГРАММНЫХ СИСТЕМАХ

Г.В. Забула, В.И. Шинкаренко
Днепропетровский национальный университет имени В. Лазаряна,
Україна
zabulus12@gmail.com, shinkarenko_vi@ua.fm

Введение

Современные программные информационные системы становятся все более интеллектуальными и приобретают адаптивные способности. Адаптация широко применяется в интерфейсах пользователя, значительно реже в части обработки и преобразования данных [1, 2], но практически не применяется при хранении данных на различных носителях.

Физическая реализация структур данных (СД) может в значительной степени влиять на временную эффективность программных систем, использующих структурированные данные. В [3, 4] показано, что время операции доступа к позиции в зависимости от физического размещения данных в оперативной памяти (ОП) может отличаться на два порядка. Следовательно, целесообразно проводить автоматическую адаптацию используемых физических реализаций для улучшения временной эффективности программных систем.

Формирование эффективных структур данных

Структуры данных (СД) не обладают функциональностью, поэтому говорить об их временных характеристиках в этом смысле некорректно. Временная эффективность структур данных определяется временной эффективностью алгоритмов обработки данных и порядком их применения. Под алгоритмами в данном случае понимаются алгоритмы операций связанных с доступом к данным, таких как: поиск, добавление, удаление (исключаются операции преобразования данных). Порядок обработки данных назовем сценарием.

Построение физической реализации выполняется путем преобразования логической структуры. Логическая структура представляет собой описание концептуальной модели, отображает составляющие модели, их свойства и связи между ними.

Для формирования конкретной физической СД используются программные шаблоны [3]. Программный шаблон – полный набор методов доступа к элементам СД, которые определяют порядок размещения данных, обеспечивают формирование СД, добавление/удаление элементов, поиск и т.п.

Для реализации программных шаблонов используются механизмы платформы .NET под собирательным названием «рефлексия». Данные механизмы позволяют создавать программные средства, результатом работы которых являются программы или их составляющие. В нашем случае, разработанное программное средство создает множество реализаций структур данных [5], поддерживающих одинаковые операции (интерфейсов). Реализации этих интерфейсов основываются на реализациях структур данных из библиотеки .NET.

Адаптация структур данных на основе генетического алгоритма

Адаптация предполагает поиск предпочтительной по временной эффективности структуры данных, состоящей из программных шаблонов, при использовании заданного сценария работы.

Для адаптации был выбран генетический алгоритм (ГА) с разработанными параметрами поиска [6]. Применение ГА продиктовано значительными: временем генерации конкретной физической структуры и процессом выполнения сценариев с использованием этой структуру. ГА позволяет ускорить подбор эффективной структуры из сконструированных за счет сокращения количества выполнения сценариев и количества генерируемых физических реализаций. Для обеспечения этих сокращений приходится жертвовать полнотой выборки – генетический алгоритм останавливает поиск в случае если кандидат на наилучшую структуру данных не изменяется в течении нескольких поколений. В этом случае считается что поиск сошелся.

Для адаптации структур данных применялись известные подходы на основе генетического алгоритма [7] с особенностями, обусловленными спецификой решаемой задачи.

Представим детализацию обобщенной схемы работы ГА.

Оптимизируемыми параметрами являются физические реализации СД каждого отношения логической СД.

Кодирование хромосом выполнено в виде вектора чисел. Каждое число (аллель) представляет собой порядковый номер в списке шаблонов. Позиция числа в хромосоме обозначает индекс отношения, к которому применяется шаблон. Гены в хромосоме не являются зависимыми или взаимозаменяемыми. В старших битах чисел могут кодироваться дополнительные свойства шаблонов. Так для связных массивов указывается количество элементов в каждом массиве.

Целевой функцией является время выполнения набора сценариев на совокупности файлов, к которым выполняется адаптация.

Инициализация начальной популяции выполняется следующим образом. Каждая хромосома заполняется одинаковыми аллелями. Количество хромосом в первом поколении соответствует количеству различных шаблонов. Например, для пяти отношений и четырех шаблонов в первом поколении будет присутствовать четыре особи: [0, 0, 0, 0, 0], [1, 1, 1, 1, 1], [2, 2, 2, 2, 2], [3(25), 3(25), 3(25), 3(25), 3(25)]. Здесь 0 – динамический массив; 1 – связный список; 2 – хэш-таблица; 3 – связный список массивов, 25 – количество элементов в связных массивах. Для повышения разнообразия хромосом, в первое поколение также добавляются несколько потомков существующих хромосом.

Оператор отбора выбирает хромосомы для следующего поколения и скрещивания со значением целевой функции выше среднего для хромосом текущего поколения.

Оператор скрещивания выполняется на основе нескольких родительских хромосом, количество которых задается параметром ГА.

Скрещивание генов хромосомы производится следующим образом: для каждого локуса генерируется случайное число, значение которого определяет номер родителя в наборе родительских хромосом: $y_{1,i} = x_{\xi_i,i}$, где $X_i = [x_{i,1}, x_{i,2} \ldots x_{i,n}]$ – родительские хромосомы, $Y_1 = [y_{1,1}, y_{1,2} \ldots y_{1,n}]$ – дочерняя хромосома; ξ – вектор случайных чисел; n – кол-во отношений.

Оператор мутации изменяет выборочные гены сложением по модулю m (кол-во шаблонов) текущего значения с единицей.

Критерий остановки – отсутствие изменений лучшей хромосомы в течении заданного количества поколений.

Параметры ГА алгоритма: вероятность участия хромосом в скрещивании, предельное количество поколений, минимальное количество хромосом в популяции, вероятность мутации, количество родительских хромосом для скрещивания, количество уникальных хромосом для скрещивания.

Разработанная методика адаптации структур данных [3] апробирована экспериментальными исследованиями адаптации СД для растровых изображений в ОП.

Установлено, что время выполнения стохастических сценариев обработки данных адаптированных СД в 1,5 – 3 раза меньше времени выполнения соответствующих сценариев на контрольных СД.

В работе определено одно из ключевых условий, при которых адаптация является целесообразной. В зависимости от

количества методов обработки данных в сценарии и конкретных программно-аппаратных сред эксплуатации СД, адаптация оправдана, если количество подобных сценариев при эксплуатации превышает 150. В этом случае предполагается достаточно быстрая окупаемость адаптации по временным затратам.

Формализация адаптации структур данных

В процессе адаптации существует проблема представления адаптирующих механизмов. Существующие способы формализации не позволяют в достаточной мере представить процесс адаптации.

В данной работе для формализации представления этих процессов предлагается использовать конструктивно-продукционные структуры (КПС) [8]. КПС – это инструмент описания множества конструкций с помощью правил подстановки, операций подстановки и операций связывания.

Используя ОКПС представляется возможным формализация всех процессов адаптации: построение сценариев, логической структуры данных, физической структуры данных, применение генетического алгоритма для поиска, всего процесса адаптации в целом. Формализация этих процессов позволит автоматизировать процесс построения адаптированных структур данных.

Выводы

Выполнены всесторонние исследования процесса адаптации структур данных на физическом уровне: теоретические, на разработанных моделях средствами КПС, и экспериментальные. Полученные результаты показывают эффективность и условия целесообразности адаптации.

Список использованной литературы

1. Зиглер К. Методы проектирования программных систем / К. Зиглер. – М.: Мир, 1985. – 328 с.
2. Касперски К. Техника оптимизации программ. Эффективное использование памяти / К. Касперски. – СПб.: БХВ-Петербург, 2003. – 464 с.
3. Шинкаренко В. И. Повышение временной эффективности структур данных в оперативной памяти на основе адаптации / В. И. Шинкаренко, Г. В. Забула // Проблеми програмування. – 2012. – № 2-3. – С. 211-218.
4. Шинкаренко В. И. Экспериментальные исследования алгоритмов в программно-аппаратных средах : монография / В. И. Шинкаренко. – Д.: Изд-во Днепропетр. нац. ун-та ж.-д. трансп. им. акад. В. Лазаряна, 2009. – 279 с.
5. Вирт Н. Алгоритмы и структуры данных / Н. Вирт – М.: ДМК, 2010. – 274 с.

6. Шинкаренко В. И. Применение генетического алгоритма в задачах адаптации структур данных / В. И. Шинкаренко, Г. В. Забула // Искусственный интеллект. – 2012. – № 3. – С. 323-331.
7. Гладков Л. А. Генетические алгоритмы / Л. А. Гладков, В. В. Курейчик, В. М. Курейчик — М: Физматлит, 2006. — 320 с.
8. Шинкаренко В. И. Конструкционно-продукционная модель структур данных на логическом уровне / В. І. Шинкаренко, В. М. Ильман, Г. В. Забула // Проблеми програмування. – 2014. – № 2-3. – С. 10-16.

КОМПЬЮТЕРНЫЕ ОБУЧАЮЩИЕ СИСТЕМЫ И ОБУЧЕНИЕ ИНОСТРАННЫМ ЯЗЫКАМ

Н.М. Зоидзе, К.М Зоидзе
Батумская государственная морская академия, Батуми, Грузия
nataliazoidze@gmail.com

При обучении иностранным языкам широко распространено использование учебных компьютерных программ. Это облегчает обучение, так как такие готовые продукты уже адаптированы с учебными процессами, и имеется опыт практической работы, что облегчает их применение. При этом нужно отметить, что применение готовых систем имеет и недостатки, из которых выделим моральное устаревание системы. Естественно, что пользователь должен владеть информацией об обновлении готовой системы (даже в случае ее полного изменения), чтобы не произошло использование такой системы в учебном процессе.

Известно, что количество компьютерных программ, обучающих иностранным языкам, уже превышает количество «обычных» школьных учебников. Эти программы основываются на применение готовых учебных курсов, которые предлагают учащимся материалы по фонетическим, грамматическим и лексическим упражнениям.

Похожие материалы обучающих «учебников» представлены в виде интерактивных упражнений и часто составлены из частей теоретических и практических аспектов, которые включены в обязательный минимум образовательного содержания иностранного языка. Готовые мультимедиа продукты и компьютерные учебные системы рекомендованы на всех этапах обучения - в процессе выработки грамматических, фонетических и лексических навыков и в процессе улучшения.

На сегодня особенно популярны следующие программы „Essential Gramar in Use“, „Britannica“, „Macmillan English Dictionary“, „Bridge to English“.

Большое значение имеет создание собственных мультимедиа программ. Значение этих процессов возрастает, если в их разработке принимают участие студенты/учащиеся.

Мультимедиа может быть простого или комплексного вида. Простой вид – когда имеем дело с представлением одного изображения или слайда, комплексный – когда одновременно идёт передача звука, изображения и текста. Такая комбинация информационных средств на основе существующей в настоящий момент компьютерной техники и информационно-коммуникационных технологий вполне возможна. Это будет иметь эффектное применение в образовательной системе, в частности в процессе изучения иностранных языков.

Современная интерактивная медиа даёт возможность пользователям самим определить условия обмена информации. Это значит, что качественно выполненный мультимедийный программный пакет оснащён разнообразным линком и даёт возможность пользователю выбрать свой метод получения/переработки информации.

Известно, что человек 80% информации получает посредством зрения. Естественно, что если учебная информация будет красиво оформлена посредством технологий изучения иностранного языка, сила её восприятия будет более эффектной, простой и приятной. Исходя из сказанного, разработчики должны заботиться о подготовке слайдов, оформления, содержимого, наличии необходимых изображений и диаграмм. Предпочтительно использовать цветной фон и шрифт среднего размера.

Список используемых источников

1. Didmanidze I., Zoidze N, Zoidze K. Technical and programming supportive means of foreign language learning / I. Didmanidze, N. Zoidze, K. Zoidze // XXIV international conference problems of decision making under uncertainties (PDMU – 2014). ABSTRACTS. September 1-5, 2014. Cesky Rudolec, Czeck Republic, 2014. – P. 31-32.

ON CLASSES OF QUASI-ARY FUNCTIONS

Ie. V. Ivanov
Taras Shevchenko National University of Kyiv, Kyiv, Ukraine
ivanov.eugen@gmail.com

Introduction

The composition-nominative approach [1] aims to propose a mathematical basis for development of formal methods of analysis and synthesis of software systems. It is grounded on several principles [1] including the *Development principle* (from abstract to concrete), *Principle of priority of semantics over syntax*, *Compositionality principle*, *Nominativity principle*. The latter Nominativity principle states that *nominative data* [1] are adequate mathematical models of various forms of data that are processed and stored by computing systems. There are several types of nominative data [1,2], and all of them are based on naming relations that associate names and values. The simplest type of nominative data is a *nominative set* (or named set) [1] which is a partial function from a set of names to a class values.

On the abstract level a computing system is modeled as a partial function that maps nominative sets (input data) to nominative sets (output data). Such functions are called biquasi-ary. A slightly more general notion is a quasi-ary function which means a function defined on nominative sets (but which may return objects of a different kind). Such functions (models) can be composed in various ways (e.g. by sequential composition, branching, etc.). Operations that construct composed systems from constituents are called compositions.

A set of compositions available to a system developer together with a set of functions obtained from some chosen set of basic functions by applications of compositions form a certain algebraic system (program algebra) which is considered as a semantic model of a computing system development language. The syntax of this development language follows naturally from this semantic model: programs are represented as terms of the mentioned algebra.

The relation of the above mentioned notions to semantics of programming languages can be illustrated on a simple educational programming language *WHILE* described in [3]. This is an imperative language in which programs are composed from statements that involve Boolean and arithmetic expressions. A program state is an assignment of values to variable names. It can be modeled as a nominative set (which is defined on assigned variables and is undefined on unassigned variables). Semantics of a statement can be represented as a (partial) biquasi-ary function (a mapping from states to states) and semantics of a boolean or arithmetic expression can be represented as quasi-ary

function which takes boolean or, respectively, integer values. Semantics of statements are composed to obtain semantics of a program.

The composition nominative approach allows one to investigate semantic properties of programs and develop methods of synthesis of systems with desired properties. In the works [4,5] on the composition-nominative approach logical systems for reasoning about computing systems and solving verification problems which extend Floyd-Hoare logic [6] were proposed.

Although applications are important, the composition-nominative approach is largely a mathematical framework and it has a number of associated theoretical questions. One of them is the problem of investigation of properties and classification of quasi-ary functions. This problem is the topic of this talk. Although some classes of these functions have been investigated in previous works on the composition-nominative approach, a general classification has not been given.

In this talk we propose a classification of quasi-ary functions over nominative sets on the basis on the following observation. In the case of a finite set of names an important difference between quasi-ary functions and ordinary n-ary functions (i.e. $f(x_1, x_2, ..., x_n)$) is that the value of a quasi-ary function may be defined even if not all its arguments are defined. However, in some cases evaluation of a quasi-ary function on a nominative set can be reduced to a number of simpler evaluations. We will define a series of classes of quasi-ary functions such that each class consists of all quasi-ary functions evaluation of which can be reduced to evaluations of n-ary function using a particular method. Then we will give necessary and sufficient conditions for membership of a particular quasi-ary function in these classes.

Nominative sets

A nominative set (or named set) is a partial function from a non-empty set of *names* V to a set of *values* A. We consider nominative sets equal if their graphs are equal. We will denote by ${}^{V}A$ the set of all nominative sets (partial functions) from V to A and by A^V denote the set of all total functions from V to A (indexed families). We will use an expression of the form $[i_1 \mapsto a_1, i_2 \mapsto a_2, ... i_n \mapsto a_n]$, where $i_1, i_2, \ldots$ are distinct names to denote a nominative set d such that the graph of d is the set $\{(i_1, a_1), (i_2, a_2), \ldots, (i_n, a_n)\}$.

For each partial function f we will denote by $dom(f)$ the set of all arguments x such that $f(x)$ is defined. By $im(f)$ we will denote the image of f, i.e. $im(f) = \{x \mid x \in dom(f)\}$. By $f(x) \cong g(y)$ we will denote the strong equality, i.e. if at least one of the values $f(x)$ and $g(y)$ is defined, then the another one is defined and they both are equal.

For each nominative set d and a set of names U we will denote by $d|_U$ the restriction of d (as a function) on U.

A nominative set d_1 is included in a nominative set d_2 (denoted as $d_1 \subseteq d_2$), if the graph of d_1 (as a function) is a subset of the graph of d_2.

Classes of Quasi-ary Functions

Let V be a *finite* nonempty set of names and A, Y be non-empty sets (set of input values and output values respectively). Let us denote: $G_V(A,Y)$ is the set of all quasi-ary functions from ${}^VA$ to Y and $G^0{}_V(A,Y)$ is the set of all functions $f \in G_V(A,Y)$ such that $dom(f) \subseteq A^V$;

The class $G^0{}_V(A,Y)$ consists of quasi-ary functions that are defined on total nominative sets (i.e. nominative sets in which all names have defined values). After choosing an arbitrary total ordering of the elements of V, evaluation of a function of this class can be straightforwardly reduced to evaluation of an n-ary function, where n is the cardinality of the set V. Then the following question arises: how can one reduce evaluation of a quasi-ary function that does not belong to $G^0{}_V(A,Y)$ (i.e. which is defined on some non-total data) to a series of evaluations of functions from $G^0{}_V(A,Y)$ (and thus can be reduced to evaluations of n-ary functions) ? This question can be formalized in different ways, one of which gives a rise to the following classes of quasi-ary functions:

Definition 1 (Constructive definitions of function classes).

(1) $\boldsymbol{G}^e{}_V(A,Y)$ is the set of all $f \in G_V(A,Y)$ such that there exist $g_U \in G^0{}_V(A,Y)$ for all $U \in 2^V \setminus \{\varnothing\}$ such that (i) $f(d) \cong g_U(d|_U)$ for each d and $U \in 2^V \setminus \{\varnothing\}$ such that $g_U(d|_U)$ is defined; (ii) $f(d)$ is undefined, if $g_U(d|_U)$ is undefined for all $U \in 2^V \setminus \{\varnothing\}$.

(2) $\boldsymbol{G}^l{}_V(A,Y)$ is the set of all $f \in G_V(A,Y)$ such that there exist $g_U \in G^0{}_V(A,Y)$ for all $U \in 2^V \setminus \{\varnothing\}$ such that (i) $f(d)$ is defined and equal to $g_U(d|_U)$ for each d and $U \in 2^V \setminus \{\varnothing\}$ such that $g_U(d|_U)$ is defined; (ii) $f(d)$ is undefined, if $g_U(d|_U)$ is undefined for all $U \in 2^V \setminus \{\varnothing\}$; (iii) for each $d \in {}^VA$ there exists not more than one $U \in 2^V \setminus \{\varnothing\}$ such that $g_U(d|_U)$ is defined.

(3) $\boldsymbol{G}^t{}_V(A,Y)$ is the set of all $f \in G_V(A,Y)$ such that either f is nowhere defined, or $dom(f) \neq \varnothing$ and $U_0 = \bigcap_{d \in dom(f)} dom(d)$, and for each $U \in 2^V \setminus \{\varnothing\}$ there exist $g_U \in G^0{}_U(A,Y)$ and $h_U \in G^0{}_U(A, 2^V \setminus \{\varnothing\})$ such that (i) $f(d) \cong g_{U_k}(d|_{U_k})$, if $d \in {}^VA$ and there exist $U_1, U_2, \ldots, U_{k+1} \in 2^V \setminus \{\varnothing\}$ ($k \geq 1$) such that $U_1 = U_0$, $U_k = U_{k+1}$, and $U_i = h_{U_{i-1}}(d|_{U_{i-1}})$ for $i = 2, 3, \ldots, k+1$; (ii) $f(d)$ is undefined otherwise.

Informally, the class $\boldsymbol{G}^e_V(A,Y)$ contains quasi-ary functions, evaluation of which on a nominative set can be reduced to parallel evaluation of functions from $G^0_U(A,Y)$, $U \in 2^V \setminus \{\varnothing\}$.

The class $\boldsymbol{G}^l_V(A,Y)$ contains quasi-ary functions, evaluation of which on a nominative set can be reduced to parallel evaluation of functions from $G^0_U(A,Y)$, $U \in 2^V \setminus \{\varnothing\}$ with the following restriction: not more than one of these evaluations must return an answer.

The class $\boldsymbol{G}^t_V(A,Y)$ contains quasi-ary functions f, evaluation of which on a nominative set can be reduced to a sequence of evaluations of functions from $G^0_U(A,Y)$, $U \in 2^V \setminus \{\varnothing\}$. The evaluation of $f(d)$ starts with reading values corresponding to a set of names U_0 that are guaranteed to have values in all nominative sets from the domain of f, and then proceeds to reading values that correspond to increasingly wide subsets U_i until all values relevant for determining $f(d)$ are obtained.

Definition 2. A quasi-ary function f on ${}^V\!A$ is called *equitone*, if for all d_1, d_2 such that $d_1 \subseteq d_2$, if $f(d_1)$ is defined, then $f(d_2)$ is defined and $f(d_1) = f(d_2)$.

For any quasi-ary function f on ${}^V\!A$ denote by $marg(f)$ the set $\{d \in {}^V\!A \mid \{d' \in {}^V\!A \mid d' \subseteq d\} \cap dom(f) = \{d\}\}$, i.e. $marg(f)$ is the set of minimal elements of $dom(f)$ with respect to nominative set inclusion.

Definition 3 (Descriptive definitions of function classes).

(1) $G^e_V(A,Y)$ is the set of all equitone functions $f \in G_V(A,Y)$ such that f([]) is undefined (where [] is a nowhere defined nominative set);

(2) $G^l_V(A,Y)$ is the set of all functions $f \in G^e_V(A,Y)$ such that for each $d \in dom(f)$ the set $\{d' \subseteq d \mid f(d') \text{ is defined}\}$ has the least element with respect to nominative set inclusion;

(3) $G^t_V(A,Y)$ is the set of all functions $f \in G^l_V(A,Y)$ such that for each set $D \subseteq marg(f)$ with at least 2 elements the set $\{d|_S \mid d \in D\}$ has at least 2 elements, where $S = \bigcap_{d \in D} dom(d)$.

Theorem 1 (Equivalence of constructive and descriptive definitions of classes of quasi-ary functions)

(1) $\boldsymbol{G}^e_V(A,Y) = G^e_V(A,Y)$;

(2) $\boldsymbol{G}^l_V(A,Y) = G^l_V(A,Y)$;

(3) $\boldsymbol{G}^t_V(A,Y) = G^t_V(A,Y)$.

Potential Applications

In the domain of embedded systems low-level programming languages (e.g. C or Assembly language) are frequently used, as they are well-supported by software and hardware platform vendors and allow developers to achieve good performance of the resulting executable code using the available compilers. However, manual mechanisms of data/memory management lead to the risk of introducing software errors

that are related to memory/data access such as uninitialized memory access, incorrect pointer dereferencing, and so on. The common outcomes of invalid data/memory access operations (e.g. an attempt to read from a missing, protected, etc. region) are:

- an error flag or an interrupt/exception is raised, the program continues execution and becomes aware of the data access error;
- the data access operation never terminates, but the program continues execution (if the operation is executed asynchronously);
- the operation yields an unpredictable result, or a fatal error or deadlock occurs after which normal program execution is impossible.

In the case of safety critical software, the last outcome is the most undesirable situation, so it makes sense to take special measures to minimize the risk of its occurrence. Software verification gives a possible solution, but it may be difficult to apply it to low-level platform-specific code. For this reason it makes sense to investigate software synthesis methods that minimize the risk of data access errors.

Consider a quasi-ary function as a specification of the required behavior of a program and its input nominative set as a representation of a memory state, where the names with defined values represent memory addresses of the existing and accessible values, and the names that do not have a corresponding defined value in this nominative set represent addresses which the program must not try to access during its operation. Then Theorem 1(3) can be interpreted as a condition under which this specification can be implemented correctly and deterministically, i.e. the quasi-ary function can be evaluated using a series of denamings such that no name with an undefined value is ever accessed. Moreover, its proof gives a method for constructing such an implementation.

References

1. Nikitchenko N. S. A composition nominative approach to program semantic / N. S. Nikitchenko.–Technical University of Denmark, Report IT-TR: 1998-020. – 1998. – 103 p.
2. Nikitchenko M.S. Stability and monotonicity of programs with respect to structural transformations of data / M.S. Nikitchenko, Ie.V. Ivanov. – Problems of Programming. – 2010. – No. 2–3. – P. 58-67. (In Ukrainian)
3. Nielson H.R. Semantics with Applications: A Formal Introduction. / H.R. Nielson, F. Nielson. – John Wiley & Sons, Inc., New York, NY, USA. – 1992.
4. Nikitchenko M. Satisfiability in composition-nominative logics / M. Nikitchenko, V. Tymofieiev // Central European Journal of Computer Science. – 2012. – No. 2. – P. 194–213.

5. Kryvolap A. Extending Floyd-Hoare logic for partial pre- and postconditions. / A. Kryvolap, M. Nikitchenko, W. Schreiner – In: CCIS 412. – Springer. – 2013. – P. 355–378.
6. Hoare, C. An axiomatic basis for computer programming. / C. Hoare // Communications of the ACM. – 1969. – P. 576–580.

АНАЛИЗ СТРОЕНИЯ МНОЖЕСТВА ИМЕННЫХ МНОЖЕСТВ

Е.В. Иванов[1], Н.С. Никитченко[1], В.Г. Скобелев[2], С.С. Шкильняк[1]
[1]Киевский национальный университет имени Тараса Шевченко
[2]Институт прикладной математики и механики НАН Украины, Донецк
ivanov.eugen@gmail.com, nikitchenko@unicyb.kiev.ua

Введение

Композиционный подход в программировании [1] явился фундаментом для систематического исследования логических исчислений, построенных на основе композиционно-номинативного подхода [2], представляющих собой математический аппарат, предназначенный для разработки формальных методов синтеза, анализа и верификации программных средств. Значение указанного направления существенно возрастает в связи с интенсивно проводимыми во всем мире исследованиями в области разработки теоретических основ построения решателей, предназначенных для автоматизированной проверки формул разрешимых теорий 1-го порядка.

Композиционно-номинативный подход представляет собой существенную детализацию теории равенства и неинтерпретируемых функций, основы которой были заложены в [3]. Эта детализация основана на том предположении, что основным объектом, на котором осуществляется построение тех или иных конструкций являются не обычные абстрактные множества, а именные множества, элементами которых являются упорядоченные пары вида (имя, значение) (т.е. любое именное множество представляет собой частичное отображение). Основными конструкциями, построенными на именных множествах являются квазиарные функции, т.е. отображения именных множеств в заданное множество или в заданные множества. Таким образом, любое логическое исчисление, построенное на основе композиционно-номинативного подхода, представляет собой теорию частичных отображений. А сам

композиционно-номинативный подход с математической точки зрения представляет собой исследование формальных методов синтеза, анализа и верификации программных средств с позиции теории категорий.

Целью настоящей работы является анализ строения множества именных множеств с позиции теории множеств и теории решеток [4, 5].

Основные понятия

Пусть V ($|V| \geq 2$) и A ($A \neq \varnothing$) – не более, чем счетные множества, соответственно, имен и данных. Обозначим через $\boldsymbol{F}_{V,A}$ множество всех (возможно частичных) отображений множества V в множество A. В соответствии с [2] элементы множества $\boldsymbol{F}_{V,A}$ назовем *V-именными множествами* над множеством A.

Замечание 1. Из этого определения вытекает, что:

1) если $|A| = 1$, то множество $\boldsymbol{F}_{V,A}$ изоморфно множеству всех подмножеств множества V;

2) если $|V|=1$, то множество $\boldsymbol{F}_{V,A}$ изоморфно множеству, состоящему из пустого множества и всех одноэлементных подмножеств множества A.

Поэтому всюду в дальнейшем считаем, что $|V| \geq 2$ и $|A| \geq 2$.

Так как каждый элемент $f \in \boldsymbol{F}_{V,A}$ однозначно определяется своим графиком

$$graph(f) = \{ (v,a) \in Dom\, f \times Val\, f \mid f(v) = a \},$$

то при исследовании строения множества $\boldsymbol{F}_{V,A}$ естественно использовать множество $\boldsymbol{G}_{V,A} = \{ graph(f) \mid f \in \boldsymbol{F}_{V,A} \}$.

Представления элементов множества $\boldsymbol{F}_{V,A}$ элементами множества $\boldsymbol{G}_{V,A}$ дает возможность следующим образом определить отношение частичного порядка на множестве $\boldsymbol{F}_{V,A}$

$$f_1 \leq f_2 \Leftrightarrow graph(f_1) \subseteq graph(f_2).$$

Из этого определения вытекает, что:

1) наименьшим элементом множества $\boldsymbol{F}_{V,A}$ является нигде не определенное V-именное множество, которое обозначим как $0_{V,A}$;

2) множеством максимальных элементов множества $\boldsymbol{F}_{V,A}$ является множество $\boldsymbol{F}^{(\mathrm{ttl})}_{V,A}$ всюду определенных V-именных множеств над множеством A.

Замечание 2. Так как $|A| \geq 2$, то при любом множестве имен V в частично-упорядоченном множестве $(\boldsymbol{F}_{V,A}, \leq)$ нет наибольшего элемента. Следовательно, при $|A| \geq 2$ частично-упорядоченное множество $(\boldsymbol{F}_{V,A}, \leq)$ не может быть изоморфно никакой булевой алгебре.

Как это обычно делается, положим

$$f_1 < f_2 \Leftrightarrow f_1 \le f_2 \,\&\, f_1 \ne f_2 \;\; (f_1, f_2 \in \boldsymbol{F}_{V,A}).$$

Теоретико-множественный анализ множества $\boldsymbol{F}_{V,A}$

Охарактеризуем вначале замкнутость множества $\boldsymbol{F}_{V,A}$ относительно тех операций, которые, по своей сути, являются теоретико-множественными операциями. Операции пересечения $\cap$ и разности $\setminus$ именных множеств могут быть определены обычными равенствами, а именно, для всех $f_1, f_2 \in \boldsymbol{F}_{V,A}$

$$f_1 \cap f_2 = f \Leftrightarrow graph(f_1) \cap graph(f_2) = graph(f);$$
$$f_1 \setminus f_2 = f \Leftrightarrow graph(f_1) \setminus graph(f_2) = graph(f).$$

Отсюда непосредственно вытекает, что:

1) для любых $f_1, f_2 \in \boldsymbol{F}_{V,A}$ истинны включения
$$Dom\,(f_1 \cap f_2) \subseteq Dom\, f_1 \cap Dom\, f_2$$
$$Dom\, f_1 \setminus Dom\, f_2 \subseteq\; Dom\,(f_1 \setminus f_2) \subseteq Dom\, f_1;$$
2) для любых $f_1, f_2 \in \boldsymbol{F}_{V,A}$ и любых $X, Y \subseteq V$ истинно равенство
$$f_1|_X \cap f_2|_Y = (f_1 \cap f_2)|_{X \cap Y}.$$

Результат теоретико-множественной операции $\cup$ объединения множеств, примененной к элементам $graph(f_1)$, $graph(f_2) \in \boldsymbol{G}_{V,A}$, может не принадлежать множеству $\boldsymbol{G}_{V,A}$, т.е. не определять именное множество.

Замечание 3. Точнее,

$$graph(f_1) \cup graph(f_2) \in \boldsymbol{G}_{V,A} \Leftrightarrow f_1|_{Dom\, f_1 \cap Dom\, f_2} = f_2|_{Dom\, f_1 \cap Dom\, f_2}.$$

Отсюда, в частности, вытекает, что при $|A| \ge 2$ формула

$$f_1 \cup f_2 = f \Leftrightarrow graph(f_1) \cup graph(f_2) = graph(f) \;\; (f_1, f_2, f \in \boldsymbol{F}_{V,A})$$

определяет на множестве $\boldsymbol{F}_{V,A}$ частичную операцию.

В соответствии с [2] в качестве аналога теоретико-множественной операции объединения множеств будем использовать бинарную операцию ∇ врезания (наложения) именных множеств на $\boldsymbol{F}_{V,A}$, определяемую следующим образом:

$$f_1 \nabla f_2 = f \Leftrightarrow graph(f_1) \cup graph(f_2|_{Dom\, f_2 \setminus Dom\, f_1}) = graph(f).$$

Отсюда непосредственно вытекает, что:

1) неравенство $f_1 \le f_1 \nabla f_2$ истинно для всех $f_1, f_2 \in \boldsymbol{F}_{V,A}$;
2) если $f_1 \le f_2$ $(f_1, f_2 \in \boldsymbol{F}_{V,A})$, то $f_1 \nabla f_2 = f_2 \nabla f_1 = f_2$;
3) нигде не определенное V-именное множество $0_{V,A}$ является нейтральным элементом для операции ∇.

Теорема 1. Алгебраическая система $(\boldsymbol{F}_{V,A}, \nabla, \cap)$ (для $|A| \ge 2$) характеризуется следующим образом:

1) $(\boldsymbol{F}_{V,A}, \nabla)$ – некоммутативный моноид;
2) $(\boldsymbol{F}_{V,A}, \cap)$ – коммутативная полугруппа без нейтрального элемента, но с нулем $0_{V,A}$, который является нейтральным элементом моноида $(\boldsymbol{F}_{V,A}, \nabla)$;

3) операция $\cap$ не дистрибутивна относительно операции ∇;
4) для операции ∇ относительно операции $\cap$ выполняется односторонний дистрибутивный закон $f_1 \nabla (f_2 \cap f_3) = (f_1 \nabla f_2) \cap (f_1 \nabla f_3)$.

Корме того, выполняется $(f_1 \cap f_2) \nabla f_3 \geq (f_1 \nabla f_3) \cap (f_2 \nabla f_3)$.

Таким образом, алгебраическая система ($\boldsymbol{F}_{V,A}$, ∇, $\cap$) не является модельной алгеброй с двумя операциями (т.е. полем, кольцом, полукольцом, булевой алгеброй). Отсюда вытекает, что конструкции, построенные в терминах алгебраической системы ($\boldsymbol{F}_{V,A}$, ∇, $\cap$), могут иметь свойства, существенно отличающиеся от свойств их аналогов, построенных в терминах модельных алгебр.

Анализ строения множества $\boldsymbol{F}_{V,A}$ с позиции теории решеток

Исследуем строение частично-упорядоченного множества ($\boldsymbol{F}_{V,A}$, $\leq$). Легко проверить, что ($\boldsymbol{F}_{V,A}$, $\leq$) – такая нижняя полурешетка, что

$$\inf \{f_1, f_2\} = f_1 \cap f_2 \quad (f_1, f_2 \in \boldsymbol{F}_{V,A}).$$

Следовательно, на частично-упорядоченном множестве ($\boldsymbol{F}_{V,A}$, $\leq$) могут быть построены все теоретико-множественные конструкции, которые определяются на нижних полурешетках.

В частично-упорядоченном множестве ($\boldsymbol{F}_{V,A}$, $\leq$) для любых таких элементов f_1, $f_2 \in \boldsymbol{F}_{V,A}$, что $f_1 \leq f_2$ интервал определяется равенством $[f_1, f_2] = \{ f \in \boldsymbol{F}_{V,A} \mid f_1 \leq f \leq f_2 \}$.

Теорема 2. Интервал $[f_1, f_2]$ ($f_1, f_2 \in \boldsymbol{F}_{V,A}$; $f_1 \leq f_2$) является полной решеткой.

Следствие 1. Алгебраическая система ($[f_1, f_2]$, $\{\cup, \cap\}$) ($f_1, f_2 \in \boldsymbol{F}_{V,A}$; $f_1 \leq f_2$) является дистрибутивной решеткой.

Следствие 2. Алгебраическая система ($[f_1, f_2]$, $\{\cup, \cap\}$) ($f_1, f_2 \in \boldsymbol{F}_{V,A}$; $f_1 \leq f_2$) является дедекиндовой решеткой.

Определим на интервале $[f_1, f_2]$ ($f_1, f_2 \in \boldsymbol{F}_{V,A}$; $f_1 \leq f_2$) унарную операцию $C_{[f_1, f_2]}$ следующим образом: для любого $f \in [f_1, f_2]$

$$C_{[f_1, f_2]}(f) = f' \Leftrightarrow graph\,(f') = graph\,(f_2) \setminus graph\,(f) \cup graph\,(f_1).$$

Следствие 3. Алгебраическая система ($[f_1, f_2]$, $\{\cup, \cap, C_{[f_1, f_2]}\}$) ($f_1, f_2 \in \boldsymbol{F}_{V,A}$; $f_1 \leq f_2$) является булевой алгеброй.

Отображение $\varphi : \boldsymbol{F}_{V,A} \to \boldsymbol{F}_{V,A}$ назовем изотонным на непустом подмножестве $S \subseteq \boldsymbol{F}_{V,A}$, если неравенство $\varphi(f_1) \leq \varphi(f_2)$ истинно для всех таких $f_1, f_2 \in S$, что $f_1 < f_2$.

Следствие 4. Пусть $[f_1, f_2]$ ($f_1, f_2 \in \boldsymbol{F}_{V,A}$; $f_1 \leq f_2$) – произвольный интервал, а $\varphi : \boldsymbol{F}_{V,A} \to \boldsymbol{F}_{V,A}$ – любое такое отображение, что $\varphi_{[f_1, f_2]}$ – изотонное отображение, для которого

истинно включение $\mathrm{Val}\varphi_{[f_1,f_2]} \subseteq [f_1, f_2]$. Тогда отображение $\varphi_{[f_1,f_2]}$ имеет неподвижную точку.

Следствие 5. Интервал $[f_1, f_2]$ ($f_1, f_2 \in \boldsymbol{F}_{V,A}; f_1 \leq f_2$) изоморфен интервалу $[0_{V,A}, f_2 \setminus f_1]$.

Максимальными интервалами в частично-упорядоченном множестве ($\boldsymbol{F}_{V,A}$, $\leq$) являются интервалы вида $[0_{V,A}, f]$, где $f \in \boldsymbol{F}^{(\mathrm{ttl})}_{V,A}$. Эти интервалы характеризуются следующим образом.

Следствие 6. Максимальные в частично-упорядоченном множестве ($\boldsymbol{F}_{V,A}$, $\leq$) интервалы изоморфны друг другу.

Заключение

В настоящей работе исследуется строение множества именных множеств с позиции теории множеств и теории решеток. Из полученных результатов вытекает, что частично-упорядоченное множество ($\boldsymbol{F}_{V,A}$, $\leq$) является объединением попарно пересекающихся изоморфных максимальных интервалов $[0_{V,A}, f]$ ($f \in \boldsymbol{F}^{(\mathrm{ttl})}_{V,A}$). Из этой формулы, в частности, следует, что для любых двух элементов $f^{(1)}, f^{(2)} \in \boldsymbol{F}^{(\mathrm{ttl})}_{V,A}$ ($f^{(1)} \neq f^{(2)}$) изоморфизм φ максимальных интервалов $[0_{V,A}, f^{(1)}]$ и $[0_{V,A}, f^{(2)}]$ определяется через семейство отображений $\{g|_{Dom\, f}\}_{f \in [0_{V,A}, f^{(1)}]}$, т.е. имеет достаточно сложную структуру.

Именно эти два указанные выше обстоятельства, во многом, обуславливают большую внутреннюю сложность различных конструкций, определяемых на частично-упорядоченном множестве ($\boldsymbol{F}_{V,A}$, $\leq$).

Список используемых источников

1. Редько В.Н. Основания композиционного программирования / В.Н. Редько // Программирование. – № 3. – 1979.
2. Нікітченко М.С. Математична логіка та теорія алгоритмів. / М.С. Нікітченко, С.С. Шкільняк – Київ: Видавничо-поліграфічний центр "Київський університет". – 2008. – 528 с.
3. Ackermann W. Solvable cases of the decision problem / W. Ackermann // The Journal of Symbolic Logic. – 1957. – N. 1. – P. 68-72.
4. Биркгоф Г. Теория решеток / Г. Биркгоф. – М.: Наука, 1984. – 568 с.
5. Скорняков Л.А. Элементы теории структур / Л.А. Скорняков – М.: Наука. – 1982. – 160 с.

ОПТИМИЗАЦИЯ ЗАПРОСОВ В РЕЛЯЦИОННЫХ БАЗАХ ДАННЫХ

И.С. Канарская
Киевский национальный университет имени Тараса Шевченко
iren_kiss@mail.ru

Введение

В настоящее время системы управления базами данных (СУБД) широко используются во многих областях деятельности человека. Наиболее распространенной является реляционная модель данных, которая была впервые предложена Э. Коддом в 1970 году [1]. С математической точки зрения реляционная база данных является конечным множеством конечных отношений различной арности между заранее определёнными множествами элементарных данных. Однако, реляционные СУБД, как и другие системы, ориентированные на ненавигационный непроцедурный интерфейс, с одной стороны, в большей степени нуждаются в оптимизации, а с другой стороны, предоставляют большие возможности оптимизации, при этом оптимизация запросов является актуальной и очень важной задачей [2]. Поэтому оптимизационные приемы в реляционных СУБД наиболее развиты.

Путь обработки запроса в реляционной СУБД

Рассмотрим типичный для современных реляционных СУБД путь обработки запроса, поступившего в СУБД на SQL-подобном языке запросов. Обработка запроса, состоит из пяти этапов.

На первом этапе запрос подвергается лексическому и синтаксическому анализу. При этом вырабатывается его внутреннее представление, отражающее структуру запроса и содержащее информацию, которая характеризует объекты базы данных, упомянутые в запросе (отношения, поля и константы). Информация о хранимых в базе данных объектах выбирается из каталогов базы данных. Внутреннее представление запроса используется и преобразуется на следующих стадиях обработки запроса. Существенным является выбор внутреннего представления, которое должно быть достаточно удобным для последующих оптимизационных преобразований.

На втором этапе запрос в своем внутреннем представлении подвергается логической оптимизации. При этом могут применяться различные преобразования, «улучшающие» начальное представление запроса, например, эквивалентные преобразования, после проведения которых получается внутреннее представление, семантически эквивалентное начальному, в частности, приведение запроса к некоторой канонической форме. Преобразования могут

быть и семантическими, когда получаемое представление не является семантически эквивалентным начальному, но гарантируется, что результат выполнения преобразованного запроса совпадает с результатом запроса в начальной форме при соблюдении ограничений целостности, существующих в базе данных. В любом случае после выполнения этого этапа обработки запроса его внутреннее представление остается непроцедурным, хотя и является более эффективным, чем начальное.

Третий этап обработки запроса состоит в выборе на основе информации, которой располагает оптимизатор, набора альтернативных процедурных планов выполнения данного запроса в соответствии с его внутреннем представлением, полученным на втором этапе. Кроме того, для каждого плана проводится временная оценка выполнения запроса по этому плану. При оценках используется статистическая информация о состоянии базы данных, доступная оптимизатору. Из полученных альтернативных планов выбирается тот, который имеет наименьшую «стоимость», и именно его внутреннее представление теперь соответствует обрабатываемому запросу.

На четвертом этапе по внутреннему представлению наиболее оптимального плана выполнения запроса формируется выполняемое представление плана, которое может быть программой в машинных кодах или быть машинно-независимым, но более удобным для интерпретации.

Наконец, на последнем, пятом этапе обработки запроса происходит его реальное выполнение в соответствии с выполняемым планом запроса. Это либо выполнение соответствующей подпрограммы, либо вызов интерпретатора с передачей ему для интерпретации выполняемого плана.

Таким образом, процесс обработки запроса можно разделить на подготовительную (включающую этапы 1 – 4) и исполнительную (этап 5) части. При этом некоторые методы оптимизации (и даже подходы к оптимизации) довольно сильно зависят от общей организации обработки запроса. При отрыве во времени процесса компиляции от реального выполнения запроса оптимизатор располагает меньшей и менее достоверной информацией, чем в том случае, когда этап компиляции тесно привязан к этапу выполнения (выполняется в рамках транзакции пользователя) [3].

Понятие «стоимости» запроса

Для оптимизации обработки запросов очень важным является понятие «стоимости» запроса. Выделяют следующие составляющие этого понятия:

1) коммуникационная стоимость, то есть сумма стоимости передачи данных из их местоположения во вторичную память, из которой загружаются данные для вычисления и стоимости перемещения результата в его местоположение;

2) стоимость доступа ко вторичной памяти, то есть стоимость загрузки порций данных из вторичной памяти в основную память, используемую при вычислении;

3) стоимость запоминания, то есть стоимость времени использования вторичной памяти и памяти буферов;

4) стоимость вычисления, то есть стоимость времени, используемого непосредственно для вычисления.

Таким образом, оптимизация обработки запросов представляет собой нетривиальную многокритериальную задачу [4].

Современные направления оптимизации запросов

На настоящее время сформировались следующие три направления исследования оптимизации обработки запросов.

1. Разработка методов оптимизации обработки последовательности запросов. Формирование этого направления обусловлено попыткой снижения стоимости обработки запроса за счет использования того фактора, что, как правило, в последовательности составных запросов содержится значительное количество общих подвыражений. Ощутимый прорыв произошел в связи с разработкой методов оптимизации обработки последовательности запросов, основанных на использовании представления запросов ориентированными ациклическими И-ИЛИ графами. Именно разработка этих методов дала возможность выработать ряд общих рекомендаций увеличения производительности SQL Microsoft Server 6.5.

2. Разработка методов лексической и семантической оптимизации запросов. Лексическая оптимизация запроса состоит в его преобразовании, направленном на устранение избыточности на основе анализа ограничений и условий, содержащихся в нем. Существуют следующие два подхода к решению этой задачи:

а) преобразование запроса, представленного на нереляционном языке, к реляционной форме, и применение к ней методов лексической оптимизации, разработанных для реляционных СУБД;

б) построение новых алгебр, предназначенных для представления запросов с учетом особенностей новых языков, и разработка методов оптимизации выражений этих алгебр.

Семантическая оптимизация запроса представляет собой валидацию и преобразование синтаксического дерева запроса к виду, который будет оптимальным для выполнения дальнейших

шагов оптимизации. При этом осуществляется преобразование запросов в каноническую форму – раскрытие представлений, преобразование подзапросов в соединения, спуск предикатов, упрощение условий, распределение предикатов и преобразование дерева условий в пути выборки.

3. Разработка методов оптимизации обработки запросов в параллельной распределенной связке «База данных – СУБД». Именно эти исследования определили следующие три направления, наиболее интенсивно развивающиеся в настоящее время:

а) разработка детерминированных и вероятностных методов «оптимизации» запросов для различных моделей параллельных распределенных связок «База данных – СУБД»;

б) построение моделей связок «База данных – СУБД» на основе использования графических процессоров;

в) разработка моделей и методов обработки и хранения больших данных, характеризующиеся большим объемом, разнообразием и скоростью, с которой структурированные и неструктурированные данные поступают по сетям передачи в процессоры и хранилища, а также наличием эффективных процессов переработки данных в требуемую информацию.

Выводы

В работе исследованы этапы обработки запросов в реляционных СУБД, показаны составляющие «стоимости» запросов, с помощью которой можно оценивать их оптимальность, и представлены современные направления оптимизации запросов. Проведенный анализ показывает, что современное состояние этого направления оптимизации требует поиска новых и нетривиальных решений, при этом, в связи с резким увеличением потоков данных, задача оптимизации запросов является актуальной.

Список используемых источников

5. Codd E.F. A Relational Model of Data for Large Shared Data Banks / E. F. Codd // Communications of the ACM. – 1970. – V. 13, N. 6. – P. 377–387.
6. Мендекович Н.А. Обзор развития методов лексической оптимизации запросов / Н.А. Мендекович, С.Д. Кузнецов // Труды ИСП РАН. – 2012. – Т. 23. – С.195 – 214.
7. Кузнецов С.Д. Методы оптимизации выполнения запросов в реляционных СУБД / С.Д. Кузнецов // Итоги науки и техники. Вычислительные науки. Т.1. – М.: ВИНИТИ, 1989. – С. 76-153.
8. Буй Д.Б. Сложность операций в базах данных (обзор) / Д.Б. Буй, В.Г. Скобелев // Радіоелектронні і комп'ютерні системи. – 2014. – №6 (70). – С.53 – 59.

МОДЕЛЮВАННЯ ПРОЦЕСІВ КРИПТОГРАФІЧНОГО ПЕРЕТВОРЕННЯ В ФАКТОР - КІЛЬЦІ ТА ЙОГО ВЛАСТИВОСТІ

А.Г. Карпов
Харківський національний університет імені В.Н. Каразіна,
Україна
andrey.karpov1994@gmail.com

Вступ

На нинішній час суттєве розповсюдження отримали крипто перетворення направленого шифрування (НШ) в кільці, в скінченному полі та в полі груп точок еліптичних кривих [1]. Недоліком перетворень в кільці та в скінченному полі є неможливість забезпечення NP - повної (субекспоненційної) складності атаки «повне розкриття». Вказані перетворення та перетворення в групах точок еліптичних кривих також мають загальний недолік – велика складність прямого та зворотного перетворення, що приводить до низької швидкодії. Останні пошуки та дослідження дозволили обґрунтувати та на нинішній час стандартизувати у вигляді стандарту США ANSI X9.98 алгоритм НШ, в основу якого покладено криптографічне перетворення в фактор - кільці. У роботі порівняльний аналіз перетворення типу X9.98 (NTRU) по критеріям стійкості та складності.

Математичні основи алгоритму NTRU

У таблиці 1 наведені параметри основного алгоритму NTRU та відповідні пояснення до них. У таблиці 2 наведені значення параметрів для різних рівнів безпеки, які рекомендує NTRU Cryptosystem [2]. При відповідному виборі параметрів, імовірність помилки розшифрування може дорівнювати 10^{-25} або менше. Значення параметрів, що наведені в таблиці 2, мають малу ймовірність помилки розшифрування. Ці параметри рекомендовані NTRU Cryptosystem для використання в комерційних застосуваннях.

Таблиця 1. Ключі та параметри алгоритму NTRU

Параметр	Коротке пояснення параметру
N	Розмір усіченого кільця многочленів R. Елементи кільця подані у вигляді поліномів степеня $N-1$ (параметр відкритий)
q	Великий модуль, згідно з яким зводиться за модулем кожний коефіцієнт многочлена в кільці R (відкритий)
p	Малий модуль, за яким зводиться кожний

	многочлен (відкритий)
f	Поліном, що є особистим (таємним) ключем
g	Поліном, який використовується для генерації відкритого ключа h з особистого F(таємний, але знищується після першого використання)
h	Відкритий ключ, також поліном
r	Випадковий поліном «забілювання» (таємний, але знищується після першого використання)
d_f	Поліном f повинен мати d_f коефіцієнтів 1 та d_f-1 коефіцієнтів – 1
d_g	Поліном g повинен мати d_g коефіцієнтів 1 та d_g коефіцієнтів – 1
d_r	Поліном r повинен мати d_r коефіцієнтів 1 та d_r коефіцієнтів – 1

Таблиця 2. Значення розмірів параметрів для різних рівнів безпеки NTRU

Рівень безпеки	N	q	p
Стандартний	251	128	3
Високий	347	128	3
Надвисокий	503	256	3

Генерування асиметричної пари ключів. Генерування асиметричної пари розпочинає, наприклад, абонент В. Для цього він вибирає, згідно з вимогами табл. 1 два випадкових поліноми f і g. Причому поліноми f і g повинні мати мультиплікативно зворотні поліноми в кільці за модулем загальних параметрів p і q. Позначимо такі мультиплікативно зворотні значення для полінома f як $f_p^{-1}(F_p)$ та $f_q^{-1}(F_q)$ відповідно. Вони мають бути розраховані точно тільки для обраного f,

$$f = (1 + pF) \bmod q$$

де p = 3 – загальний (доменний параметр), а F – випадковий поліном, алгоритм генерування якого визначено в стандарті [2].

Далі абонент В обчислює свій відкритий ключ згідно з правилом:

$$h = p(f_q^{-1} \cdot g) \bmod q$$

Поліноми f і g повинні мати конкретну кількість коефіцієнтів, 0, +1 і –1 згідно табл. 1, тобто обрані випадково із множини поліномів з фіксованим числом символів коефіцієнтів 0, +1 і -1. Особистий (таємний) ключ у вигляді многочлена f разом із

зворотними $f_p^{-1}(F_p)$ та $f_q^{-1}(F_q)$ є таємними ключовими даними, h – відкритим ключем. Асиметричною парою ключів є пара (f, h) особистого і відкритого ключів, що є поліномами степені не вище N–1.

Алгоритм зашифрування. Нехай абонент А бажає направлено зашифрувати для абонента В двійкове повідомлення m. Для цього А повинен знати справжні загальні параметри N, p і q та відкритий ключ абонента В – h_b. Далі абонент А генерує «випадково» згідно з табл. 1, у кільці R_q многочлен r_a (по суті, ключ сеансу). Безпосередньо абонент А зашифрування виконує шляхом обчислення значення

$$c = (r_a \cdot h_b + m) \bmod q$$

Алгоритм розшифрування. Для розшифрування криптограми c абонент В повинен мати справжні загальні параметри N, p і q. Розшифрування здійснюється способом згортки поліномів N–1 степеня в кільці $(Z/\ qZ)\ [X]/(X^{N-1})$ особистого ключа f_b та криптограми c як

$$a = (f_b \cdot c) \bmod q$$

Надалі коефіцієнти полінома a для отримання повідомлення m зводяться за модулем в інтервалі [A, A+q-1], а також по модулю p для отримання кандидата в розшифроване повідомлення m'.

f_b та f_q^{-1} є мультиплікативно зворотними за модулем q. На останок, при зведенні полінома a за модулем p, отримаємо, що (при $f_p^{-1} = F_p$

$$(F_p \cdot a) \bmod p = (F_p \cdot pr \cdot g + F_p \cdot f \cdot m) \bmod p = m$$

Оцінка складності криптоаналізу методом повного розкриття

Відносно асиметричних криптоперетворень криптографічна стійкість до атак «повне розкриття», коли визначається особистий (конфіденційний) ключ, зводиться до розв'язання деяких математичних задач [1]. Так, для RSA-перетворення задача повного розкриття в основному зводиться до факторизації модуля перетворення, для перетворення в полі Галуа – до дискретного логарифмування в полі Галуа, для перетворення в групі точок еліптичних кривих – до дискретного логарифмування на еліптичній кривій. Далі, для перетворень у кільцях зрізаних поліномів задача зводиться до розв'язання певних задач в алгебраїчних решітках. Особливу групу складають задачі криптоаналізу криптоперетворень зі спарюванням точок еліптичних кривих тощо. Зрозуміло, що ці задачі є субекспоненційно або експоненційно

складними, особливо для відповідно обґрунтованих вибором параметрів і ключів.

Стійкість атаки проти загрози «повне розкриття» визначається складністю розв'язання рівнянь відносно особистого ключа. Складність розв'язування цього рівняння набагато вища, ніж у кільці та полі. У полі – субекспоненційна, а в групі точок еліптичних кривих – експоненційна складність.

Порівняльний аналіз відомих перетворень НШ та NTRU

Оцінки часу криптоаналізу відносно NTRUEncrypt, RSA та EC криптосистем для різних рівнів безпеки наведено в табл. 3.

В таблиці 4 наведено порівняння швидкодії асиметричних криптоперетворень.

Таблиця 3. Тимчасова складність криптоаналізу асиметричних перетворень

Рівень безпеки (довжина симетричного шифру, бітів)	Оцінка часу криптоаналізу, MIPS-years	Криптосистема		
		RSA	NTRU X9.98-2010	EC
80(2TDEA)	10^9	1024	N=263	f =160 – 223
112(3TDEA)	10^{17}	2048	N=401	f = 224 – 255
256(AES-256)	10^{63}	15360	N=1171	f = 512 +

Таблиця 4. Порівняння швидкостей NTRU, RSA та EC

Рівень стійкості (бітів БСШ)	Операцій / секунда		
	NTRU	EC	RSA
112	10638	951	156
128	9901	650	12
256	5000	116	1

Таким чином, основною перевагою NTRU є суттєво підвищена, у порівнянні з RSA, DSA та EC, швидкодія. За даними компанії Security Innovation, що займається розробкою NTRU, алгоритм NTRU в режимі направленого шифрування забезпечує збільшення швидкодії до 200 разів.

Можливості криптоаналізу на квантовому комп'ютері

В кінці 20 століття стало зрозумілим, що суттєве підвищення швидкодії обчислень при здійсненні криптоаналізу може бути досягнуто при використанні алгоритмів, що ґрунтуються на квантових комп'ютерах. У 1992 році Дойтч та у 1994 році Шор розробили квантовий алгоритм факторизації модуля RSA перетворення та розв'язку дискретного логарифмічного рівняння відповідно.

В 1996 році Гровером було запропоновано узагальнений квантовий алгоритм пошуку у великій базі даних. Він дозволяє зменшити обчислювальну складність поточного вичерпного пошуку з $O(2^n)$ до $O(2^{n/2})$, тобто за складністю близький, або по суті реалізує, метод створення колізій.

Значні зусилля криптоаналітиків по пошуку ефективних методів криптоаналізу NTRU донині залишаються безуспішними і носять експоненційний характер складності. У 2003 році Людвіг застосував квантовий алгоритм пошуку Гровера на основі решіток. Було виявлено, що час роботи квантового алгоритму скороченого пошуку (QRS) різко скоротився в порівнянні з класичними методами, але все ще мав експоненційну часову складність. У 2005 році Грехем довів, що обчислювальна складність QRS більша, ніж у атаки «зустріч посередині». Також ще у 2009 році NTRU вважався безпечним від квантових атак, оскільки вимагав експоненційної складності. У 2010 році Ванг застосував квантовий алгоритм для пошуку особистого ключа NTRU. Системи на базі NTRU можуть бути уразливі до квантового криптоаналізу, хочу ще не так давно зазначалося, що такі схеми будуть стійкі проти нього.

Висновки

Таким чином, криптоперетворення в фактор - кільці має суттєві переваги. Основними з них є менша складність криптографічних перетворень, тобто підвищена швидкодія, а також експоненційна складність здійснення атаки типу «повне розкриття». Основний недолік - збільшення розміру шифротексту.

Список використаних джерел

1. Горбенко І.Д. Прикладна криптологія: теорія, практика, застосування. Монографія / І.Д. Горбенко, Ю.І. Горбенко. – Харків: Форт, 2012. – 880 с.
2. Качко Е.Г. Исследование методов вычисления инверсии в алгоритме NTRU / Е.Г. Качко, Д.С. Балагура, К.А. Погребняк, Ю.И. Горбенко // Прикладная радиоэлектроника. – 2013. – N 2. – С. 254-257.

3. Горбенко Ю.І. Аналіз можливостей квантових комп'ютерів та квантових обчислень для криптоаналізу сучасних криптосистем / Ю.І. Горбенко, Р.С. Ганзя // Східно-Європейський журнал передових технологій. – 2014. – Т.1, №9 (67). – С. 8 – 15.

ІНТЕЛЕКТУАЛЬНИЙ МЕТОД ПРОГНОЗУВАННЯ УСПІШНОСТІ ПРОГРАМНИХ ПРОЕКТІВ НА ОСНОВІ АНАЛІЗУ СПЕЦИФІКАЦІЇ ВИМОГ

А.В.Красій
Хмельницький національний університет, Україна
andriy-krasiy@yandex.ua

Вступ

Наразі людство все частіше покладається на програмне забезпечення при вирішенні складних задач управління ракетами, літаками, атомними реакторами, стрімко зростає кількість програмних проектів з високою вартістю, тому актуальним і дуже важливим є вміння оцінити можливу успішність програмного проекту на ранніх етапах життєвого циклу, а також допомогти замовнику обрати ймовірно успішний програмний проект з множини альтернативних програмних проектів.

Успішність програмного проекту на етапі проектування можна ймовірнісно оцінити на основі прогнозованих значень основних характеристик розроблюваного за проектом програмного забезпечення (ПЗ) - тривалості проекту, вартості, ефективності, складності, зручності використання, кросплатформності, якості та надійності ПЗ. Аналіз специфікації програмного проекту надає множину показників для подальшого розрахунку (прогнозування) значень основних характеристик розроблюваного за проектом ПЗ, а також для вибору прийнятної моделі життєвого циклу ПЗ [1, 2]. Доведено необхідність та розроблено штучну нейронну мережу (ШНМ), яка опрацьовує множину показників специфікації, здійснює апроксимацію показників та надає прогнозовані кількісні значення характеристик ПЗ [2] - рис.1.

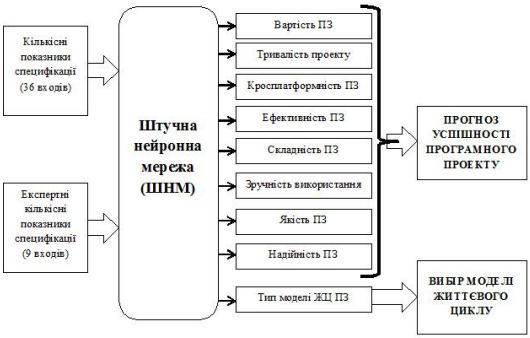


Рис.1. Концепція нейромережного прогнозування характеристик ПЗ на основі аналізу специфікацій

Інтелектуальний метод прогнозування успішності програмного забезпечення на основі аналізу специфікації вимог

ШНМ прогнозування характеристик ПЗ на основі аналізу специфікацій видає множину $SCH_{Sp} = \{Cs, Dsp, Ecef, Cp, Cx, Ub, Qs, Rs, Slcm\}$ значень основних характеристик програмного проекту Sp: значення вартості (Cs), тривалості програмного проекту (Dsp), економічної ефективності ($Ecef$), кросплатформності (Cp), складності (Cx), зручності використання (Ub), якості (Qs), надійності (Rs), а також номеру прийнятної моделі життєвого циклу ($Slcm$). ШНМ прогнозування характеристик ПЗ на основі аналізу специфікацій навчена так, що всі значення характеристик ПЗ належать інтервалу (0;1]. На основі отриманих з ШНМ прогнозування значень основних характеристик ПЗ замовнику складно комплексно оцінити успішність проекту, оскільки складно вірно інтерпретувати одержані значення характеристик. Тому необхідно розробити метод, результати якого допомагатимуть замовнику оцінити успішність програмного проекту, інтерпретуючи та інтегруючи одержані значення характеристик ПЗ.

Значення характеристик програмного проекту Sp з підмножини $SCH_{Min_{Sp}} = \{CS, Dsp, Cx\} \subset SCH_{Sp}$, що є близькими до 1, вказують на низьку успішність програмного проекту Sp. Значення характеристик програмного проекту Sp з підмножини $SCH_{Max_{Sp}} = \{Ecef, Cp, Ub, Qs, Rs\} \subset SCH_{Sp}$,

що є близькими до 1, вказують на високу успішність програмного проекту Sp.

Інтегративним показником успішності $Ipup_{Min_{Sp}}$ *проекту* Sp називатимемо кількісний показник успішності програмного проекту Sp на основі прогнозованих характеристик ПЗ, значення яких, що є близькими до 1, негативно впливають на успішність програмного проекту. *Інтегративним показником успішності* $Ipup_{Max_{Sp}}$ *проекту* Sp називатимемо кількісний показник успішності програмного проекту Sp на основі прогнозованих характеристик ПЗ, значення яких, що є близькими 1, позитивно впливають на успішність проекту.

Для отримання графічного представлення інтегративного показника успішності $Ipup_{Min_{Sp}}$ створено систему координат, яка має три вісі - Cs, Dsp, Cx; для отримання графічного представлення інтегративного показника успішності $Ipup_{Max_{Sp}}$ створено систему координат, яка має п'ять осей - $Ecef$, Ub, Cp, Qs, Rs [3]. Нехай ШНМ прогнозування характеристик ПЗ на основі аналізу специфікацій надала нам наступну множину значень характеристик для програмного проекту Sp:

$$SCH_{ANN_{Sp}} = \{Cs_{ANN}, Dsp_{ANN}, Cx_{ANN}, Ecef_{ANN}, Cp_{ANN}, Ub_{ANN}, Qs_{ANN}, Rs_{ANN}\}.$$

Тоді отримаємо наступні графічні представлення інтегративних показників успішності: $Ipup_{Min_{Sp}}$ (рис.2) - площа виділеного суцільною жирною лінією трикутника $Cs_{ANN}Dsp_{ANN}Cx_{ANN}$; $Ipup_{Max_{Sp}}$ (рис.3) - площа виділеного суцільною жирною лінією п'ятикутника $Ecef_{ANN}Ub_{ANN}Cp_{ANN}Qs_{ANN}Rs_{ANN}$. Для визначення ймовірності успішності програмного проекту за інтегративними показниками успішності $Ipup_{Min_{Sp}}$ та $Ipup_{Max_{Sp}}$ знадобляться також максимальні значення інтегративних показників успішності: $Ipup_{Min_{bad}}$ (рис.3) - площа трикутника $CsDspCx$, окресленого

пунктирною лінією та $Ipup_{Max_{best}}$ (рис.4) - площа п‘ятикутника $EcefUbCpQsRs$, окресленого пунктирною лінією.

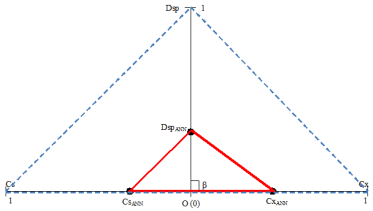


Рис.3. Графічне представлення інтегративного показника успішності $Ipup_{Min_{Sp}}$ та його максимального значення $Ipup_{Min_{bad}}$

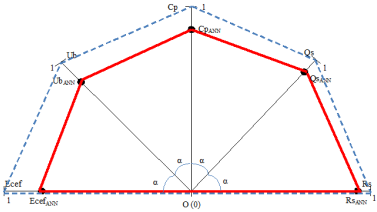


Рис.4. Графічне представлення інтегративного показника успішності $Ipup_{Max_{Sp}}$ та його максимального значення $Ipup_{Max_{best}}$

Для знаходження інтегративного показника успішності $Ipup_{Min_{Sp}}$ скористаємось формулою площі трикутника за відомою стороною ($Cs_{ANN}Cx_{ANN}$) та висотою ($ODsp_{ANN}$):

$$Ipup_{Min_{Sp}} = \frac{1}{2} \cdot (Cs_{ANN} + Cx_{ANN}) \cdot Dsp_{ANN} . \qquad (1)$$

Для знаходження інтегративного показника успішності $Ipup_{Max_{Sp}}$ розіб’ємо п’ятикутник на 4 трикутники - $Ecef_{ANN}OUb_{ANN}$, $Ub_{ANN}OCp_{ANN}$, $Cp_{ANN}OQs_{ANN}$, $Qs_{ANN}ORs_{ANN}$ і знайдемо площу для кожного з трикутників за

двома відомими сторонами та кутом між ними (кут $\alpha = 45°$), після чого додамо отримані площі трикутників:

$$Ipup_{Max_{Sp}} = \frac{1}{2} \cdot \frac{\sqrt{2}}{2} \cdot (Ecef_{ANN} \cdot Ub_{ANN} + \\ + Ub_{ANN} \cdot Cp_{ANN} + Cp_{ANN} \cdot Qs_{ANN} + \\ + Qs_{ANN} \cdot Rs_{ANN}) \quad (2)$$

Знайдемо максимальні значення інтегративних показників успішності $Ipup_{Min_{bad}}$ та $Ipup_{Max_{best}}$ за формулами (1) і (2):

$$Ipup_{Min_{bad}} = \frac{1}{2} \cdot (1+1) \cdot 1 = 1 \; ; \quad (3)$$

$$Ipup_{Max_{best}} = 0.3536 \cdot (1 \cdot 1 + 1 \cdot 1 + \\ + 1 \cdot 1 + 1 \cdot 1) = 1.4144 \quad (4)$$

Інтегративні показники успішності програмного проекту дають змогу визначити ймовірність його успішності.

Оскільки характеристики підмножини $SCH_{Min_{Sp}}$ вимагають мінімізації, то значення $Ipup_{Min_{bad}} = 1$ - це найгірше значення інтегративного показника $Ipup_{Min_{Sp}}$. Характеристики підмножини $SCH_{Max_{Sp}}$ навпаки вимагають максимізації, тому значення $Ipup_{Max_{best}} = 1.4144$ - це найкраще значення інтегративного показника $Ipup_{Max_{Sp}}$.

Тоді за інтегративним показником $Ipup_{Min_{Sp}}$ ймовірність успішності $P_{Min_{Sp}}$ програмного проекту Sp складає:

$$P_{Min_{Sp}} = 1 - \frac{Ipup_{Min_{Sp}}}{Ipup_{Min_{bad}}} = 1 - Ipup_{Min_{Sp}} \; . \quad (5)$$

За інтегративним показником $Ipup_{Max_{Sp}}$ ймовірність успішності $P_{Max_{Sp}}$ програмного проекту Sp складає:

$$P_{Max\,Sp} = \frac{Ipup_{Max\,Sp}}{Ipup_{Max_{best}}} = 0.7070 \cdot Ipup_{Max\,Sp} \,. \qquad (6)$$

Оскільки інтегративний показник $Ipup_{Min\,Sp}$, на основі якого обчислюється ймовірність успішності $P_{Min\,Sp}$ проекту, залежить від трьох основних характеристик ПЗ, а інтегративний показник $Ipup_{Max\,Sp}$, на основі якого обчислюється ймовірність успішності $P_{Max\,Sp}$ проекту, залежить від п'яти основних характеристик ПЗ, тоді усереднене значення ймовірності успішності P_{Sp} обчислимо за формулою:

$$P_{Sp} = \frac{3 \cdot P_{Min\,Sp} + 5 \cdot P_{Max\,Sp}}{8} \qquad (7)$$

Формалізація інтелектуального методу прогнозування успішності програмного забезпечення на основі аналізу специфікації вимог матиме наступний вигляд:

$$SCH_{Sp} = \{SCH_{Min\,Sp}, SCH_{Max\,Sp}\} \Rightarrow$$

$$\Rightarrow \begin{cases} Ipup_{Min\,Sp} = f_1(SCH_{Min\,Sp}) \\ Ipup_{Max\,Sp} = f_2(SCH_{Max\,Sp}) \end{cases} \Rightarrow$$

$$\Rightarrow \begin{cases} P_{Min\,Sp} = f_3(Ipup_{Min\,Sp}, Ipup_{Min_{bad}}) \\ P_{Max\,Sp} = f_4(Ipup_{Max\,Sp}, Ipup_{Max_{best}}) \end{cases} \Rightarrow \qquad (8)$$

$$\Rightarrow P_{Sp} = f_5(P_{Min\,Sp}, P_{Max\,Sp})$$

де функція f_1 обчислюється за формулою (1), функція f_2 – за формулою (2), функція f_3 – за формулою (5), функція f_4 – за формулою (6), функція f_5 – за формулою (7); показники $Ipup_{Min_{bad}}$ та $Ipup_{Max_{best}}$ обчислюються за формулами (3) та (4) відповідно.

Розглянемо *для прикладу* програмний проект «Альфа» софтверної компанії "СТУ-Електронікс" (м. Хмельницький). В результаті аналізу наданої специфікації ШНМ сформувала наступні

висновки щодо прогнозованих характеристик ПЗ, яке буде розроблене за специфікацією: $Cs_{ANN} = 0.2$; $Dsp_{ANN} = 0.1$; $Cx_{ANN} = 0.15$; $Ecef_{ANN} = 0.8$; $Ub_{ANN} = 0.85$; $Cp_{ANN} = 0.87$; $Qs_{ANN} = 0.89$; $Rs_{ANN} = 0.91$.

Згідно методу прогнозування успішності програмних проектів, за формулами (1), (2), (5), (6), (7) обчислимо інтегративні показники успішності $Ipup_{Min_{Sp}}$ і $Ipup_{Max_{Sp}}$ для програмного проекту, а також ймовірності успішності цього проекту:

$$Ipup_{Min_{Sp}} = \frac{1}{2} \cdot (0.2 + 0.15) \cdot 0.1 = 0.0175;$$

$$Ipup_{Max_{Sp}} = 0.3536 \cdot (0.8 \cdot 0.85 + 0.85 \cdot 0.87 + 0.87 \cdot 0.89 + 0.89 \cdot 0.91) = 1.0621;$$

$$P_{Min_{Sp}} = 1 - 0.0175 = 0.9825 = 98.25\%$$

$$P_{Max_{Sp}} = 0.7070 \cdot 1.0621 = 0.7509 = 75.09\%$$

$$P_{Sp} = \frac{3 \cdot 0.9825 + 5 \cdot 0.7509}{8} = 0.8376 = 83.76\%$$

Отже, ймовірність успішності програмного проекту «Альфа» складає близько 0.84.

Висновки

Прогнозовані за допомогою ШНМ кількісні значення характеристик ПЗ дають можливість обчислювати інтегративні показники успішності для різних проектів. Розроблений метод прогнозування успішності програмного забезпечення на основі аналізу специфікації вимог дозволяє отримати інтегративні показники успішності $Ipup_{Min_{Sp}}$ і $Ipup_{Max_{Sp}}$ різних програмних проектів на основі прогнозованих значень характеристик ПЗ для цих проектів, отриманих в результаті аналізу специфікації штучною нейронною мережею. Крім цього, розроблений метод дозволяє обчислити ймовірності успішності програмних проектів. Метод відрізняється від відомих тим, що дозволяє прогнозувати успішність програмних проектів, порівнювати програмні проекти комплексно за основними характеристиками проекту і розроблюваного ПЗ та прогнозованим значенням ймовірності успішності (а не тільки за вартістю та

тривалістю, як відбувається наразі) та виконувати обґрунтований вибір програмного проекту замовником для подальшої реалізації.

Список використаних джерел

1. Говорущенко Т.О. Визначення характеристик та вибір моделі життєвого циклу програмного забезпечення на основі аналізу специфікацій / Говорущенко Т.О., Красій А.В. // Вісник Хмельницького національного університету – Хмельницький: ХНУ, 2013. - №6, с.201-208
2. Красій А.В. Моделювання процесу прогнозування характеристик програмного забезпечення на основі аналізу специфікацій // Комп'ютерно-інтегровані технології: освіта, наука, виробництво - Луцьк: ЛНТУ, 2014 - с.66-76
3. A.Krasiy. The Concept of Prediction of Software Characteristics and Software Projects Success Based on the Analysis of Software Requirements Specifications / A.Krasiy, T.Hovorushchenko // Матеріали IX Міжнародної науково-технічної конференції CSIT-2014. - Львів: Вид. НУ "Львівська політехніка", 2014.

ВЛАСТИВОСТІ СИСТЕМ ВИВОДУ МОНОТОНИХ ЛОГІК ФЛОЙДА-ХОАРА НАД ІЄРАРХІЧНИМИ ДАНИМИ

А.В. Криволап
Київський національний університет імені Тараса Шевченка, Україна
krivolapa@gmail.com

Вступ

Верифікація є однією з найважливіших та найскладніших проблем в процесі розробки програмних систем. Одним з підходів до вирішення даної проблеми є застосування систем автоматичного доведення теорем. Основою більшості таких систем є логіка Флойда-Хоара. Це пов'язано зі зручністю представлення тверджень у вигляді трійок – передумова та післяумова, представлені формулами логіки та програма. Проте, класична логіка Флойда-Хоара визначена для тотальних предикатів

Монотонна логіка Флойда-Хоара над простими номінативними даними

У статті [1] приведено розширення логіки Флойда-Хоара на випадок часткових предикатів. Семантика даного розширення подана за допомогою програмних алгебр квазіарних предикатів та функцій. Відповідні алгебри задані над простими номінативними даними. Для представлення семантики трійок Флойда-Хоара використовується спеціальна композиція. При перенесенні визначення істинності трійка Флойда-Хоара для тотальних предикатів на випадок часткових при визначенні композиції Флойда-Хоара, виявляється, що композиція не буде монотонною. Монотонність в даному випадку розуміється як монотонність за визначеністю предикатів та функцій. Тому в [1] було запропоновано визначення композиції Флойда-Хоара, яка була б монотонною та неперервною. Відповідне розширення логіки Флойда-Хоара на часткові предикати будемо для зручності називати в подальшому монотонною логікою Флойда-Хоара.

Всі поняття, що не визначені в даній роботі, будемо розглядати в сенсі [1]. Будемо позначати V – множина імен, A – множина базових значень, ${}^{V}A$ – номінативна (іменна) множина, $Pr^{V,A}$ – множина квазіарних предикатів над ${}^{V}A$, $Prg^{V,A}$ – множина бікваазіарних функцій над ${}^{V}A$.

Тоді композиція Флойда-Хоара матиме тип $FH : Pr^{V,A} \times Prg^{V,A} \times Pr^{V,A} \to Pr^{V,A}$ та буде визначена наступним чином:

$$FH(p,pr,q)(d)=\begin{cases} T, \text{ якщо } p(d)\downarrow = F \vee q(pr(d))\downarrow = T, \\ F, \text{ якщо } p(d)\downarrow = T \wedge q(pr(d))\downarrow = F, \\ \text{невизначено в інших випадках.} \end{cases}$$

Для того, щоб побудоване розширення можна було використовувати для верифікації, необхідно задати систему виводу. Розглянемо приклад простої імперативної мови програмування, мову WHILE, та систему виводу для неї, запропоновану в [2]. Істинність будемо розуміти, як неспростовність. Можна побачити, що в монотонній логіці Флойда-Хоара відповідна система виводу не буде ані коректною, ані повною. Існує декілька шляхів вирішення даної проблеми – додавання обмежень до правил виводу, або ж розгляд обмеженого класу трійок Флойда-Хоара. Відповідні системи виводу та їх властивості було розглянуто в роботах [3].

Ієрархічні дані та асоціативні номінативні алгоритмічні алгебри Глушкова

Монотонна логіка Флойда-Хоара, разом з повними та коректними системами виводу, була побудована для простих номінативних даних. Проте, в мовах програмування широко застосовуються складні дані. В статті [4] розглянуто ієрархічні номінативні дані, та на їх основі побудовано асоціативні номінативні алгоритмічні алгебри Глушкова. Серед зазначених вище алгебр розглянемо алгебру над номінативними даними зі складними іменами та складними значеннями. Під складними іменами розуміється скінчена послідовність елементів множини V. В той же час складні значення можуть бути як елементами множини базових значень, так і номінативними даними.

Асоціативною номінативною алгоритмічною алгеброю предикатів та функцій над номінативними даними зі складними іменами та складними значеннями називається наступна двохосновна алгебра:

$$NGA^{a}_{CC}(V,A) =< Pr_{CC}(V,A), Fn_{CC}(V,A); \vee, \neg, \bullet, IF, WH, \cdot,$$
$$(Asg^{u})_{u\in V^{+}}, (S^{\bar{u}}_{F})_{\bar{u}\in\bar{U}}, (S^{\bar{u}}_{P})_{\bar{u}\in\bar{U}}, Id, True, \perp_{F}, \perp_{P}, (u!)_{u\in V^{+}},$$
$$Empty, IsEmpty >$$

Тут $Pr_{CC}(V,A)$ – множина предикатів над складними номінативними даними, $Fn_{CC}(V,A)$ – множина біномінативних функцій, $\vee, \neg$ – композиції диз'юнкції та заперечення, $\bullet, IF, WH$, – композиції послідовного виконання, галуження та циклу, $\cdot$ композиція передбачення, $(Asg^u)_{u\in V^+}, (S_F^{\bar{u}})_{\bar{u}\in\bar{U}}, (S_P^{\bar{u}})_{\bar{u}\in\bar{U}}$ – параметричні композиції присвоєння та суперпозиції, $Id, True, \bot_F, \bot_P$ – композиції тотожної функції, константи істина, всюди невизначеної функції та предикату, $(u!)_{u\in V^+}, Empty, IsEmpty$ – композиції предикату перевірки імені та константи порожнього номінативного даного.

Будемо використовувати дану алгоритмічну алгебру для подання семантики монотонної логіки Флойда-Хоара над ієрархічними даними. Синтаксис та функція інтерпретації випливає з семантики природнім чином, лише додаються формули спеціального вигляду – трійки Флойда-Хоара: якщо p, q – формули, та pr – терм, то $\{p\}pr\{q\}$ – спеціальна формула Флойда-Хоара. І тоді семантика її визначається наступним чином: $\square\{p\}pr\{q\}\square = \square p\square \to \square pr\cdot q\square$.

Якщо ми розглянемо систему виводу з *T*-зростаючими ассерціями для простих номінативних даних та адаптуємо її до випадку ієрархічних номінативних даних, то ми отримаємо наступну систему виводу:

$$\{S_P^x(p,h)\}\ Asg^x(h)\ \{p\}$$

$$\{p\}\ Id\ \{p\}$$

$$\frac{\{p\}\ pr_1\ \{q\}, \{q\}\ pr_2\ \{r\}}{\{p\}\ pr_1 \bullet pr_2\ \{r\}}$$

$$\frac{\{r\wedge p\}\ pr_1\ \{q\}, \{\neg r\wedge p\}\ pr_2\ \{q\}}{\{p\}\ IF(r, pr_1, pr_2)\ \{q\}}$$

$$\frac{\{r\wedge p\}\ pr\ \{p\}}{\{p\}\ WH(r, pr)\ \{\neg r\wedge p\}}$$

$$\frac{\{p'\}\ pr\ \{q'\}}{\{p\}\ pr\ \{q\}}, p\models_T p', q'\models_T q$$

Тут під $p\models_T p'$ розуміємо логічний наслідок за істиною.

Теорема 1. Система виводу для *T*-зростаючих ассерцій в монотонній логіці Флойда-Хоара над ієрархічними даними коректна.

Аналогічно за всіма коректними системами виводу для простих номінативних даних, розглянутими в [3], можна побудувати коректні системи виводу для монотонних логік Флойда-Хоара над ієрархічними даними. Також даний результат зберігається і для випадку ієрархічних даних зі складними іменами та простими значеннями і для випадку ієрархічних даних з простими іменами та складними значеннями.

Висновки

В роботі було розглянуто монотонну логіку Флойда-Хоара над ієрархічними номінативними даними. Продемонстровано, як можна для логіки над ієрархічними даними побудувати систему виводу по аналогії з системою виводу для випадку простих номінативних даних. Показано, що такі системи будуть коректні. Повнота побудованих систем потребує подальшого вивчення.

Список використаних джерел

1. A.V. Kryvolap, M.S. Nikitchenko, W. Schreiner Extending Floyd-Hoare logic for partial pre- and postconditions / A.V. Kryvolap, M.S. Nikitchenko, W. Schreiner // 9th International Conference ICTERI 2013 Revised Selected Papers.– 2013 .–CCIS vol. 412.– pp. 355-378.
2. H.R. Nielson, F. Nielson Semantics with applications: a formal introduction / H.R. Nielson, F. Nielson. – John Wiley & Sons Inc. – 1992. – 240 p.
3. M.S. Nikitchenko, A.V. Kryvolap Inference Systems for Floyd-Hoare logic with partial predicates / M.S. Nikitchenko, A.V. Kryvolap // Proceedings of the Twelfth International Conference Informatics 2013. –2013. – pp. 88-93.
4. V.G. Skobelev, M.S. Nikitchenko, I. O. Ivanov On Algebraic Properties of Nominative Data and Functions / V.G. Skobelev, M.S. Nikitchenko, I. O. Ivanov // CCIS. – 2014 (In Print).

АНАЛИЗ СВОЙСТВ ПСЕВДОСЛУЧАЙНЫХ ПОМЕХОУСТОЙЧИВЫХ КОДОВ

Т.В. Лавровская, С.Г. Рассомахин
Харьковский национальный университет им. В.Н. Каразина, Украина
lavrovska92@gmail.com

Введение

Информационные технологии характеризуются быстрой сменой концептуальных направлений технических и программных средств, методов, а также сфер применения. Наблюдается постоянное возрастание роли систем и сетей мобильного доступа и массового вещания. Эффективность этих технологий в значительной степени определяется пропускной способностью канала связи, а также удельной частотной и энергетической эффективностью систем передачи информации, использующих помехоустойчивые коды. Несмотря на огромные достижения, полученные в последние годы в области сетевых технологий, рациональное использование физического ресурса каналов передачи остается наиболее "узким" местом. Доказательством этого является тот факт, что производительность вычислительных средств возрастает, примерно, на 1 – 2 порядка за каждые 10 лет, в то время как скорость передачи данных, ограниченная физическими характеристиками каналов, остается, практически, неизменной [1].

Анализ существующих проблем применения псевдослучайных кодов

В настоящее время в процессе развития сетей передачи данных постоянно повышаются требования к скорости передачи информации и качеству предоставляемых сервисов. В связи с этим возрастают удельные энергетические и частотные затраты для достижения требуемых скоростей [2]. Причиной этого является тот факт, что при существующих методах кодирования и построения физических переносчиков данных (сигналов) повышение скорости информационного обмена в стремлении приблизиться к пропускной способности каналов, как правило, достигается за счет расширения требуемой полосы частот или увеличения мощности передатчиков. Поэтому целью данной работы является рассмотрение одного из возможных способов построения псевдослучайного помехоустойчивого кода. Данные коды при реализации в недвоичных каналах со сложными сигналами позволяют повысить помехозащищенность, практически, без снижения эффективной скорости передачи [3].

Случайные коды объемно-кубической укладки в поле Пуассона

Способ решения задачи построения оптимального двоичного кода, который имеет при заданном объеме кодовой книги – $M = 2^k$ (k –количество информационных двоичных символов в блоке сообщения) и длине блока – n наибольшее значение минимального кодового расстояния – $d_{\min}$ (при достаточно большом n), может быть основан на случайном выборе M разрешенных кодовых комбинаций из 2^n возможных.

При случайном формировании канальных символов кодовая книга представляется в виде совокупности сигналов объемной укладки. Для каждого из M блоков символов (сообщений) двоичного источника длиной k бит формируется сообщение из n символов канала из R^n. Каждый символ является действительным числом, равномерно распределенным в диапазоне $\Delta = 2\sqrt{3 \cdot R \cdot E_b}$, расположенном симметрично относительно нуля. В этом случае средняя энергия, расходуемая для передачи кодового слова, составит $E_c = (n \cdot \Delta^2)/12 = n \cdot R \cdot E_b = k \cdot E_b$.

Общий случай канального кодирования, при котором символы сообщений источника S_c мощностью алфавита q_s, объединенные в блоки длиной k, отображаются в символы сообщений канала C_c аналогичными параметрами $q_c \neq q_s, n \neq k$: $S(q_s, k) \Leftrightarrow C(q_c, n)$. Для обеспечения однозначного отображения необходимо, чтобы выполнялось равенство ${q_s}^k = {q_c}^n$. Для придания канальному коду свойства помехоустойчивости необходимо выполнение неравенства: ${q_s}^k < {q_c}^n$.

В соответствии с законом больших чисел точки кодовой книги n-символьного кода при достаточно большой длине блока располагаются вблизи поверхности n-мерной сферы, обладающей радиусом $r = \sqrt{E_c}$. При посимвольном построении кода равномерно случайным образом в фиксированном объеме, кодовая

книга образует в n-мерном пространстве случайное (Пуассоново) поле точек.

Действие аддитивного белого гауссова шума (АБГШ), при оценивании числовых значений канальных символов (для, практически, любого из известных методов модуляции), приводит к тому, что наблюдаемые на выходе канала кодовые точки распределяются вокруг истинных точек внутри сферы неопределенности с радиусом $r_{N_0} = \sqrt{n \cdot N_0 / 2}$, где N_0 – спектральная плотность мощности АБГШ. Их распределение для гауссова канала подчинено многомерному нормальному закону с независимыми координатами. На основании свойств поля Пуассона можно определить вероятность декодирования с ошибкой, которая произойдет, если внутри сферы с радиусом r_{N_0} окажется хотя бы одна кодовая точка, кроме истинной: $P_{ОШ} = 1 - \exp(-\lambda \cdot V(r_{N_0}))$, где λ – плотность кодовых точек в n-мерном поле; $V(r_{N_0})$ – объем гиперсферы с радиусом r_{N_0}. Результаты аналитического расчета требуемых значений длин блоков псевдослучайного кода в поле Пуассона для достижения остаточной вероятности декодирования с ошибкой в приведении одному биту (BER) $P_{ОШ} = 10^{-5}$ представлены в виде соответствующих чисел на рис.1.

Результаты аналитического моделирования псевдослучайных кодов.

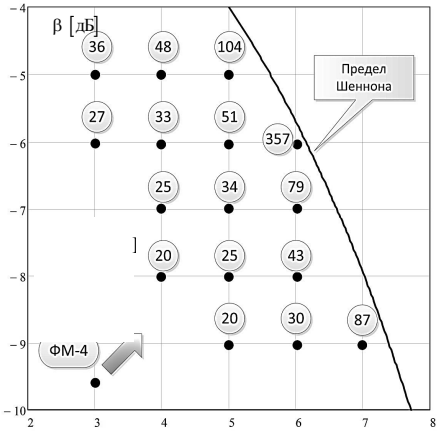


Рис. 1 – Отображающие точки равномерно случайных кодов

На рис.1 показано расположение отображающих точек случайных кодов на плоскости удельной эффективности систем передачи информации [2].

Результаты расчетов (рис.1) показывают, что выбор, например, длины блока порядка 30-40 канальных символов позволяет получить случайный код, повышающий одновременно показатели удельной энергетической (β) и удельной частотной (γ) эффективности на 3 дБ по сравнению с некодированной модуляцией.

Выводы

Существенной преградой для практического использования методов случайного и псевдослучайного кодирования является то, что декодирование таких кодов основывается на реализации правила максимального правдоподобия и возможно лишь с использованием переборных алгоритмов. Вычислительная сложность таких алгоритмов возрастает экспоненциально с длиной блока кода и при практически требуемых значениях длины блока является неприемлемой. Поэтому актуальными для дальнейших исследований являются методы построения и декодирования псевдослучайных кодов на основе правил, мало уступающих по объективности правилу максимального правдоподобия и обладающих вычислительной сложностью, не выше полиномиальной.

Список использованной литературы:

1. Широкополосные беспроводные сети передачи информации / В. М. Вишневский, А. И. Ляхов, С. Л. Портной, И. В. Шахнович – М.: Техносфера, 2005. – 592 с.
2. Помехоустойчивость и эффективность систем передачи информации / Под ред. А. Г. Зюко. − М.: Радио и связь, 1985.− 272 с.
3. Рассомахин С. Г. Технология псевдослучайного кодирования в сетевых коммуникационных протоколах канального уровня / С.Г. Рассомахин // Системи обробки інформації. – 2012. – Вип. 3(101), т.2. – С. 206 – 211.

КОНЦЕПЦІЯ ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ ТЕХНОЛОГІЙ ПРОЕКТУВАННЯ, МЕТОДОЛОГІЙ ТА СЕРЕДОВИЩ РОЗРОБЛЕННЯ ДЛЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ РІЗНИХ ТИПІВ

Р.А.Малярчук, Т.О.Говорущенко
Хмельницький національний університет, Україна
sayhello2fly@gmail.com, tat_yana@ukr.net

Вступ

З аналізу галузі створення програмного забезпечення (ПЗ) випливає, що наразі для створення повнофункціональних програмних додатків програмістам доводиться використовувати комплексні засоби розроблення - технологію проектування ПЗ, методологію розроблення ПЗ та середовище розроблення ПЗ.

Питанням оцінювання ефективності технології проектування, методології та середовища розроблення для ПЗ певного типу слід приділяти підвищену увагу: як показує досвід, без ефективної технології/методології/середовища навіть невеликі програмні проекти навряд чи можуть бути успішними.

Статистика успішності програмних проектів за даними The Standish Group International [1] представлена на рис. 1.

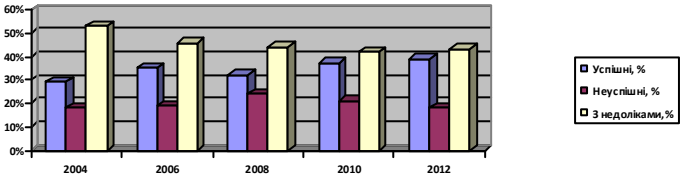


Рис.1. Статистика успішності впровадження програмних проектів за даними The Standish Group International

Аналіз даних з рис. 1 дав можливість побачити, що частка проектів з недоліками є досить сталою величиною і складає мінімум 42%. При цьому лише невелика кількість (максимум 39% за 1994-2012 роки) проектів має необхідні функціональні можливості при існуючих обмеженнях за вартістю та терміном [1].

З проведеного аналізу відомих технологій проектування, методологій та середовищ розроблення [2] очевидним є існування значної кількості технологій/методологій/середовищ за відсутності універсальної технології/методології/ середовища, яка підходила б для будь-якого програмного проекту, а також за відсутності чітких стандартизованих критеріїв оцінювання та вибору технології проектування, методології та середовища розроблення ПЗ.

Тому при всій множині існуючих технологій/методологій/середовищ кількість програмних проектів з недоліками та неуспішних програмних проектів залишається досить високою (61% у 2012 році), а процес розроблення ПЗ залишається недетермінованим, і результат його завжди невідомий. Такий стан справ можна пояснити наступною причиною в контексті технологій/методологій/середовищ - їх неефективністю для ПЗ певного типу.

Отже, задача оцінювання ефективності технологій проектування, методологій та середовищ розроблення для ПЗ різних типів є актуальною на етапі проектування, оскільки від ефективності технології/методології/середовища залежить майбутня успішність програмного проекту.

Сучасний стан галузі вибору технології проектування, методології та середовища розроблення ПЗ

Галузеві публікації вказують на часте використання тієї чи іншої *технології проектування* для ПЗ певного типу:

1) технологія RUP часто використовується при створенні систем автоматизованого проектування, CASE-систем, систем з елементами штучного інтелекту, а також систем підтримки структурно-параметричного синтезу об'єктів [3];

2) технологія Borland застосовується як софтверними компаніями, які реалізують великі дорогі програмні проекти, так і невеликими софтверними компаніями завдяки демократичним цінам багатьох редакцій цієї технології [4];

3) технологія Oracle призначена для швидкого створення та розгортання Інтернет-додатків, динамічних Web-порталів та розгортання Web-сервісів [5].

Дані відомості можна використовувати як рекомендації щодо використання технології проектування для ПЗ різних типів, але в галузевих публікаціях відсутня інформація про те, які (тип та кількість) з цих програмних проектів є успішними, тобто невідомо, чи є ефективною та чи інша технологія для ПЗ певного типу.

Оцінювання ефективності *методології розроблення* взагалі вимагає врахування багатьох різноманітних умов та факторів [6]: 1) масштаб проекту; 2) критичність проекту; 3) кількість та розподіл повноважень учасників проекту; 4) ступінь новизни проекту; 5) очікувана тривалість проекту; 6) вимоги замовника; 7) специфіка та складність проекту.

Знов-таки галузеві публікації дають можливість зробити наступні висновки щодо використання методологій проектування:

1) для великих, розрахованих на тривалий термін, проектів, як правило, використовуються методології з більшим ступенем

формалізації – традиційні та ітеративні методології, в той час як для менш масштабних проектів, розрахованих на оперативну реалізацію, підходять менш формалізовані гнучкі методології [6];

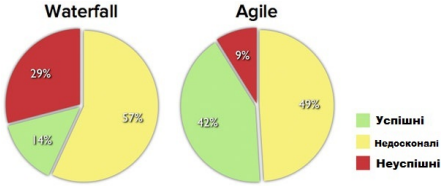
2) методологія RUP використовується для масштабних та тривалих проектів [7];

3) методологія MSF є оптимальною *для* великих проектів, які вимагають дотримання балансу між ресурсами, часом розроблення та можливостями [7];

4) методологія SCRUM ідеальна для невеликих та середніх проектів, особливо, якщо під час процесу розроблення очікується внесення численних змін [8];

5) методологія XP розрахована на невеликі проекти, в яких не виникає необхідності у створенні детальної документації та регламентації всіх кроків розроблення [9].

Отже, галузеві публікації знову не відображають кількості успішних проектів, розроблених з використанням тієї чи іншої методології. Єдине джерело, яке вносить хоч якусь ясність у це питання, це діаграма порівняння успішності програмних проектів, реалізованих за каскадними та AGILE-методологіями (рис.2) [10], яка показує, що AGILE-проекти втричі успішніші за каскадні проекти, але з неї незрозуміло, яку саме з гнучких методологій було використано для успішних програмних проектів, а також проекти якого типу розглядались при цьому дослідженні.



Джерело: The CHAOS Manifesto, The Standish Group, 2012.

Рис.2. Порівняння успішності програмних проектів, реалізованих за каскадними та AGILE- методологіями

Оцінювання ефективності *середовища розроблення ПЗ* теж зводиться до багатокритеріальної задачі і далеко не очевидне. Галузеві публікації показують наступне використання середовищ розроблення для ПЗ різних типів:

1) середовище C++ Builder дозволяє створювати як консольні додатки, так і додатки з графічним інтерфейсом користувача [11];

2) Code::Blocks використовується для багатоцільових програмних проектів [12];

3) MonoDevelop дає можливість розробникам швидко створювати додатки для кишенькових пристроїв та web-додатки ASP.NET для Linux, Windows, MacOSX; дозволяє створення web-проектів на XSP [13];

4) Microsoft Visual Studio дозволяє розробляти як консольні додатки, так і додатки з графічним інтерфейсом, а також web-сторінки, web-додатки та web-служби [14];

5) NetBeans підтримує розроблення для платформ J2SE і J2EE [15].

Галузеві публікації також не показують, скільки програмних проектів стали успішними в результаті розроблення в тому чи іншому середовищі.

При оцінюванні технології проектування, методології та середовища розроблення ПЗ розробники виділяють наступні проблеми [16]: 1) відсутність єдиної бази знань про технології/методології/середовища; 2) відсутність єдиного стандарту для оцінювання технологій/методологій/середовищ; 3) відсутність єдиних критеріїв, за якими оцінюються технології/методології/середовища для ПЗ різних типів. Оцінювання ефективності технології/методології/середовища може підвищити якість ПЗ, а також знизити витрати на його розроблення.

Концепція оцінювання ефективності технології проектування, методології та середовища розроблення для ПЗ різних типів

Зрозуміло, що кожна софтверна компанія використовує вже придбані та впроваджені технологію проектування, методологію та середовище(а) розроблення, для роботи з якими є достатня кількість фахівців. Навряд чи якась софтверна компанія погодиться купувати та впроваджувати іншу технологію/методологію/середовище, перенавчати своїх або наймати інших фахівців, яку система підтримки прийняття рішень визначила як більш переважну для ПЗ саме такого типу. Отже, розроблення системи підтримки прийняття рішень щодо вибору технології/методології/середовища не є нагальною задачею, тому що результати її роботи навряд чи використовуватимуть компанії-розробники.

Дослідницька задача полягає у іншому – визначити, на скільки вдало вдасться спроектувати та розробити ПЗ певного типу, використовуючи певні технологію проектування, методологію та середовище розроблення, тобто на скільки успішним буде програмний проект із використанням тієї чи іншої технології/методології/середовища. Отже, іншими словами, *актуальною науковою задачею* є визначення ефективності конкретної технології проектування, методології та середовища розроблення для ПЗ певного типу.

Якщо оцінювати ефективність за адитивним критерієм, то матимемо формулу вигляду (1):

$$E = \sum_{i=1}^{n} a_i \cdot x_i \,, \qquad (1)$$

де x_i - характеристики ПЗ (наприклад, якість, надійність, зручність використання, вартість, тривалість, швидкодія): $x_i \in [0;1]$, де 0 – показує найменшу можливість реалізації характеристики з використанням певної технології/ методології/середовища, а 1 – найбільшу можливість реалізації характеристики з використанням певної технології/ методології/середовища; a_i - вагові коефіцієнти характеристик, які відображають роль та пріоритетність конкретної характеристики для даного ПЗ, тобто відображають роль характеристики для ПЗ даного типу з врахуванням цільового призначення та типу програмного проекту: нехай $a_i \in [-10;10]$, але кожний ваговий коефіцієнт a_i повинен обмежуватись своїм діапазоном в загальному діапазоні – саме цей діапазон вказуватиме важливість характеристики x_i для конкретного ПЗ (наприклад, для програмної системи електронного навчання характеристика «зручність використання» повинна мати ваговий коефіцієнт $a_1 \in [-10;10]$, а характеристика «швидкодія» може мати ваговий коефіцієнт $a_2 \in [-1;1]$).

Висновки

Вирішення задачі оцінювання ефективності конкретної технології проектування, методології та середовища розроблення для ПЗ певного типу допоможе розробнику оцінити, чи варто йому братись за той чи інший програмний проект за наявної технології/методології/ середовища (і не витрачати часу та коштів на неуспішні програмні проекти), а замовнику – оцінити, на скільки вдалим буде програмний проект, якщо його реалізовуватиме конкретна софтверна фірма, тобто дозволить зробити обґрунтований висновок про підходящість або непідходящість технології/методології/середовища, а відтак і софтверної компанії, яка їх використовує, для розроблення конкретного ПЗ або ПЗ певного типу на основі отриманого значення ефективності.

Для оцінювання ефективності технологій проектування, методологій та середовищ розроблення для ПЗ певного типу слід вирішити наступні задачі:

1) провести аналіз софтверних компаній України та світу за наступними критеріями: які вони використовують технології/методології/середовища; яке ПЗ розробляють; які значення (ступені виконання) основних характеристик ПЗ забезпечують; типи та кількість успішних і неуспішних (через непідходящість технології/методології/середовища) проектів;

2) визначити значення основних характеристик x_i ПЗ з точки зору можливості їх реалізації з використанням певної технології/ методології/середовища на основі аналізу п.1;

3) визначити роль та пріоритетність характеристик для ПЗ кожного типу з врахуванням цільового призначення та типу програмного проекту, а також визначення діапазонів значень вагових коефіцієнтів a_i для ПЗ різних типів та різного призначення. Це є дуже важливою задачею, тому що саме вагові коефіцієнти a_i повинні не давати можливості компенсувати низьке значення зреалізованості важливої характеристики високим значенням зреалізованості іншої, менш важливої, характеристики (наприклад, для програмної системи електронного навчання високе значення можливості реалізації характеристики «швидкодія» не повинне в жодному разі компенсувати низького значення можливості реалізації характеристики «зручність використання», оскільки значущість цих характеристик абсолютно нерівнозначна для такої системи). Отже, значення вагових коефіцієнтів a_i повинні бути підібрані таким чином, щоб при неможливості (або низькій можливості) реалізації важливої характеристики конкретного ПЗ певною технологією/методологією/середовищем ефективність технології/методології/середовища для розроблення цього ПЗ дуже стрімко зменшувалась;

4) обґрунтовано обрати компоненти (статистичні або інтелектуальні) для визначення значень основних характеристик x_i і, особливо, для визначення діапазонів та значень вагових коефіцієнтів a_i - в залежності від обсягу отриманої інформації;

5) на основі обраних компонентів розробити моделі, методи та програмні засоби оцінювання ефективності конкретної технології проектування, методології та середовища розроблення для ПЗ певного типу.

Список використаних джерел

1. CHAOS Manifesto: Think Big, Act Small, 2013 // [Electronic resource] - Access mode: http://www.versionone.com/assets/img/files/CHAOSManifesto2013.pdf
2. Т.О.Говорущенко. Аналіз процесу вибору технології проектування, методології та середовища розроблення програмного забезпечення / Т.О.Говорущенко, Р.А.Малярчук // Вісник Хмельницького національного університету – Хмельницький: ХНУ, 2014. - №6
3. RUP – рациональный унифицированный процесс разработки программного обеспечения // [Електронний ресурс] – Режим доступу: http://www.structuralist.narod.ru/dictionary/rup.htm
4. М.Фаулер. Рефакторинг: улучшение существующего кода / М.Фаулер. – СПб.: Символ-Плюс, 2012. – 489 с.
5. Oracle Developer Suite // [Electronic resource] – Access mode: http://www.interface.ru/oracle/OracleDS10g.htm
6. Разработка программного обеспечения: Методологии разработки программного обеспечения // [Електронний ресурс] – Режим доступу: http://bms-soft.com.ua/ru/services/razrabotka-programmnogo-obespecheniya/metodologii-razrabotki-po
7. Р. К. Мартин. Быстрая разработка программ: принципы, примеры, практика / Р. К. Мартин, Д. В. Ньюкирк, Р. С. Косс – М.: Вильямс, 2004. – 752 с.
8. Обзор методологии разработки программного обеспечения SCRUM // [Електронний ресурс] – Режим доступу: http://www.dpgrup.ru/methodology-scrum.htm
9. Разработка программного обеспечения: Методологии разработки программного обеспечения // [Електронний ресурс] – Режим доступу: http://bms-soft.com.ua/ru/services/razrabotka-programmnogo-obespecheniya/metodologii-razrabotki-po
10. Agile Succeeds Three Times More Often Than Waterfall // [Electronic resource] – Access mode: http://www.mountaingoatsoftware.com/blog/agile-succeeds-three-times-more-often-than-waterfall
11. Д.Холингворт. Borland C++ Builder 6. Руководство разработчика / Д.Холингворт, Б.Сворт, М.Кэшмэн, П.Густавсон.— М.: «Вильямс», 2004. — 976 с.
12. Code::Blocks features // [Electronic resource] – Access mode: http://www.codeblocks.org/features
13. MonoDevelop // [Electronic resource] – Access mode: http://monodevelop.com/

14. Н.Рендольф. Visual Studio 2010 для профессионалов / Н.Рендольф, Д.Гарднер, М.Минутилло, К.Андерсон — М.: «Диалектика», 2011. —1184 с.
15. Монахов В.В. Язык программирования Java и среда NetBeans / Монахов В.В. – СПб.: БХВ-Петербург, 2011. – 704 с.
16. Т.О.Говорущенко. Підтримка вибору середовища програмування для системного програмного забезпечення / Т.О.Говорущенко, Р.А.Малярчук, В.В.Лаврінчук // ІТСП-2014. ІІІ Всеукраїнська НПК молодих учених та студентів. Збірник наукових праць - Хмельницький: Гонта А.С., 2014 - с.372-381

DYNAMIC LOGIC

A.N.Maslov
Limited Liability Research and Publishing Centrum "Ray",
Moscow, Russia
anmaslov@rambler.ru

Introduction

The traditional logic and the theory of sets are based on the law of identity. According to the law of identity all classes under consideration are invariable.

However the majority of classes vary, as well as the real world.

Only some abstract classes are really invariable.

A class is called a set of elements or classes. Here an element is something invariable and indivisible in ours consciousness.

Suppose A and B are some classes. Let $A + B$ denote the sum (union) of classes, $A \times B$ denote the product (intersection) of classes and $A - B$ denote the difference of classes.

Any sequence of modifications of a class in time is called notion.

The notion is called a growing class or a growing notion if this sequence does not decrease.

Some notions can gain new elements and lose them. For example, "students at the lecture".

A notion is called static if the notion does not vary in time otherwise it is called dynamic.

Each logical operation corresponds to some operation over notions. Operations of traditional logic correspond to some operation over static concepts.

Some of these operations are applicable to growing classes, together with any notions. In particular the sum and the product are applicable to any notions.

We can introduce a difference of growing classes in this logic but not negation. In such minimum logic there will be a False, but there will be no absolute true.

We can partially compensate lack of operation of negation by means of the following modification of rules of De Morgan:

$$C - (A + B) = (C - A) \times (C - B)$$

$$C - (A \times B) = (C - A) \times (C - B)$$

Further we will consider differences of dynamic logic and the traditional logic and we will define a negation in the dynamic logic.

Difference of classes

A difference of any growing classes is called a monomial.

For example, it is possible to represent the notion "now living" as a difference of the growing classes "all ever bore" and "all dead."

A notion is called a polynomial if it is the sum of several monomials.

The theorem. A class of all monomials is closed w.r.t. sum, product and difference.

Comprehensive dynamic class

A growing dynamic class U is defined as:

1) U(1) = Ø where Ø is empty class.

2) U(t+1) contains class U(t) and all its subclasses.

Dynamic class **U** we will name a comprehensive class.

U is the proper growing dynamic class.

This class should be distinguished from "set of all sets", which is a source of paradoxes of the set theory.

Any class enters into comprehensive class **U**, but it enters from some instant.

Operation of Negation

Now we shall give the following definition. Suppose U is the comprehensive class. If A is a subclass of a class **U**(t) then a class **U**(t)-A is a subclass of the class **U**(t+1). This class is called a negation of the class A and it is denoted by -A.

Note that **U** (t)-A is not necessarily an element of **U**(t).

It is obvious that - - A contains class A, but not necessarily coincides with it. It will agree with Old Indian logic and the intuitionism.

The temporal logic of von Wright

Von Wright introduced new logical operation *T*. Expression "*pTq*" is read thus: "Now there is an event *p*, and then, i.e. during the following moment, there is an event *q*".

In our designations-:

$$pTq(t) \equiv p(t) \times q(t+1)$$

The temporary logic of von Wright is the axioms of propositional logic plus four axioms of von Wright (with the operation T) plus the extensionality rule.

In this logic negation and implication have classical sense. We claim that these operations should depend on time.

Propositional logic and natural languages

In our common speech the conjunction "and" can denote "and then" (as in the logic of von Wright) or "and consequently" and others. The sense "and simultaneously" is the closest one to the classical logic. For example the pair of phrases: "the Man has asked where the train

terminated, and has boarded a coach" and "the Man has boarded a coach and has asked where the train terminates" are identical in a formal logic, but not in practical life.

In ordinary speech the conjunction "or" can correspond to logical "or" or logical "either". Sometimes it is meant that the second operand is fulfilled after not realizations of the first one. For example, a wife can tell: "Buy me this brilliant or divorce."

The conjunctions "and not" usually implies simultaneous fulfillment of operands, but the outcome can appear in the future. For example "I take a red and not big apple".

There are also the conjunctions explicitly pointing at the fulfillment of reasoning or operations at various times. For example conjunctions: “then”, “before”, “after” and others.

Dynamic propositional logic

A dynamic logic deals with functions.

Operations of dynamic logic +, × and - are irrespective of time.

Other operations depend on time:

1. Operation of negation "-" (is introduced above).
2. Operation T (von Wright).
3. Operation of implication "if x then y follows"

$$(x \to y)(t) \equiv x(t) \Rightarrow y(t+1)$$

where the symbol $\Rightarrow$ is used in sense of classical logic.

Note that $x \to x$ is tautologous argument only for the static class.

4. Operation of inverse implication "x if then y follows"

$$(x \leftarrow y)(t) \equiv x(t) \Leftarrow y(t+1)$$

where the symbol $\Leftarrow$ is used in sense of classical logic.

Note that $x \leftarrow x$ is tautologous argument.

5. Operation of equality

$$(x + y)(t) \equiv x(t) = y(t+1)$$

Note that $x=x$ is tautologous argument only for the static classes.

References

1. Vopěnka Petr. Introduction to mathematics in the alternative set theory/ Bratislava: Alfa, 1989. (translation into Russian 2004).
2. Vilkelyonok Diana. Aspect of a relativity in a grammatical category of time / KALBŲ STUDIJOS. 2012, 20 NR., STUDIES ABOUT LANGUAGES. 2012. NO. 20 with. 45-56.
3. von Wright G.K./ Explanation and Understanding. Cornell University, Press, Ithaca. N. Y., 1971. xvii + 230 p.

АЛГЕБРИ КВАЗІАРНИХ ВІДНОШЕНЬ

М.С. Нікітченко, С.С. Шкільняк
Київський національний університет імені Тараса Шевченка,
Україна
nikitchenko@unicyb.kiev.ua, sssh@unicyb.kiev.ua

Вступ

Поняття відношення належить до найважливіших понять математики. Під n-арним відношенням зазвичай розуміють [1] множину кортежів довжини n. Водночас низка задач інформатики та програмування вимагають узагальнення цього поняття. Метою даної роботи є побудова та вивчення алгебр квазіарних відношень та дослідження їх зв'язків із алгебрами квазіарних предикатів.

Скінченне n-арне відношення можна розглядати як таблицю, що має n стовпчиків. Рядок таблиці (n-ка базових значень даних) – це елемент відношення. При цьому в деяких випадках заповненими можуть бути не всі клітинки таблиці. Наприклад, якщо розглядати екзаменаційну відомість як таблицю, то не всі її клітинки заповнюються під час іспиту (зокрема, через неявку студента на іспит).

Формально таку частково заповнену таблицю можна задати наступним чином. Нехай V – множина атрибутів (предметних імен), A – множина предметних значень. Часткову функцію із V в A назвемо номінативною (іменною) множиною. Клас всіх таких множин позначаємо ${}^{V}A$. *Квазіарне відношення* – це довільна підмножина $R \subseteq {}^{V}A$. Тепер номінативна множина, що входить у відношення R, може розглядатися як частково заповнений рядок таблиці.

Іменні множини та квазіарні предикати

Нагадаємо необхідні для подальшого викладу визначення.

Поняття, які тут не визначаються, тлумачимо в сенсі [2, 3].

V-іменна множина (V-ІМ) над A – це часткова однозначна функція $d : V \to A$. V-ІМ подаємо у вигляді $[v_1 \mapsto a_1,...,v_n \mapsto a_n,...]$, де $v_i \in V$, $a_i \in A$, $v_i \neq v_j$ при $i \neq j$.

Для V-ІМ введемо функцію $asn : {}^{V}A \to 2^{V}$:

$asn(d) = \{v \in V \mid v \mapsto a \in d$ для деякого $a \in A\}$.

Операцію $\|_{-x}$ видалення компоненти з іменем x задамо так:

$d\|_{-x} = [v \mapsto a \in d \mid v \neq x]$.

Операцію ∇ накладки V-ІМ η на V-ІМ δ задамо таким чином:

$\delta \nabla \eta = \eta \cup [v \mapsto a \in \delta \mid v \notin asn(\eta)]$.

Задамо операцію реномінації $\mathrm{r}^{v_1,\dots,v_n}_{x_1,\dots,x_n} : {}^{V}A \to {}^{V}A$ *:*

$$\mathrm{r}^{v_1,\ldots,v_n}_{x_1,\ldots,x_n}(d) = d\nabla[v_1 \mapsto d(x_1),\ldots,v_n \mapsto d(x_n)]\,.$$

Далі зазвичай скорочено пишемо $\overline{y}$ замість $y_1,\ldots,y_n$.

Операцію реномінації ІМ продовжуємо на множини ІМ:

$$\mathrm{r}^{\overline{v}}_{\overline{x}}(L) = \{\mathrm{r}^{\overline{v}}_{\overline{x}}(d) \mid d \in L\},\ \text{де}\ L \subseteq {}^{V}A\,.$$

Введемо відношення $=_{-x}$ рівності зточністю до компоненти з іменем x: $d_1 =_{-x} d_2$, якщо $d_1\|_{-x} = d_2\|_{-x}$.

Послідовне застосування двох операцій $\mathrm{r}^{\overline{v}}_{\overline{x}}$ (зовнішня) та $\mathrm{r}^{\overline{u}}_{\overline{y}}$ (внутрішня) можна подати у вигляді однієї операції реномінації, яку назвемо згорткою операцій $\mathrm{r}^{\overline{v}}_{\overline{x}}$ та $\mathrm{r}^{\overline{u}}_{\overline{y}}$ і позначатимемо $\mathrm{r}^{\overline{v}}_{\overline{x}} \square \mathrm{r}^{\overline{u}}_{\overline{y}}$.

Нехай маємо послідовне застосування операцій реномінації $\mathrm{r}^{v_1,\ldots v_i,u_1,\ldots,u_k}_{s_1,\ldots,s_i,z_1,\ldots,z_k}$ та $\mathrm{r}^{v_1,\ldots,v_n,w_1,\ldots,w_m}_{x_1,\ldots,x_n,y_1,\ldots,y_m}$, де $\{w_1,\ldots,w_m\} \cap \{u_1,\ldots,u_k\} = \varnothing$. Тоді для кожного $d \in {}^{V}A$ маємо: $\mathrm{r}^{v_1,\ldots,v_n,w_1,\ldots,w_m}_{x_1,\ldots,x_n,y_1,\ldots,y_m} \square \mathrm{r}^{v_1,\ldots v_n,u_1,\ldots,u_k}_{s_1,\ldots,s_n,z_1,\ldots,z_k}(d) =$

$$= \mathrm{r}^{v_1,\ldots,v_n,w_1,\ldots,w_m}_{x_1,\ldots,x_n,y_1,\ldots,y_m}(\mathrm{r}^{v_1,\ldots v_i,u_1,\ldots,u_k}_{s_1,\ldots,s_i,z_1,\ldots,z_k}(d)) = \mathrm{r}^{v_1,\ldots,v_n,w_1,\ldots,w_m,u_1,\ldots,u_k}_{a_1,\ldots,a_n,b_1,\ldots,b_m,z_1,\ldots,z_k}(d)\,,$$

де кожні a_i та b_j задаються так:

$$a_i = \begin{cases} x_i, \text{ якщо } x_i \notin \{v_1,\ldots v_n,u_1,\ldots,u_k\},\\ s_l, \text{ якщо } x_i = v_l \text{ для деякого } v_l,\\ z_l, \text{ якщо } x_i = u_l \text{ для деякого } u_l; \end{cases}$$

$$b_j = \begin{cases} y_j, \text{ якщо } y_j \notin \{v_1,\ldots v_n,u_1,\ldots,u_k\},\\ s_l, \text{ якщо } y_j = v_l \text{ для деякого } v_l,\\ z_l, \text{ якщо } y_j = u_l \text{ для деякого } u_l. \end{cases}$$

Під V-квазіарним предикатом на множині A розуміємо довільну (часткову неоднозначну, взагалі кажучи) функцію вигляду $P : {}^{V}A \to \{T, F\}$. Клас V-квазіарних предикатів на A позначимо Pr^A.

Область істинності та *область хибності* предиката $P : {}^{V}A \to \{T, F\}$ – це множини

$$T(P) = \{d \in {}^{V}A \mid T \in P(d)\}\ \text{ та }\ F(P) = \{d \in {}^{V}A \mid F \in P(d)\}.$$

Ми трактуємо часткові неоднозначні квазіарні предикати як відношення між ${}^{V}A$ та множиною істиннісних значень $\{T, F\}$. Назвемо їх предикатами реляційного типу. Вони формалізують найпростіше уточнення поняття часткового неоднозначного предиката.

Предикат P однозначний, якщо $T(P) \cap F(P) = \varnothing$.

Предикат P тотальний, якщо $T(P) \cup F(P) = {}^{V}A$.

Ім'я $z \in V$ (строго) неістотне для предиката P, якщо

для всіх $d_1, d_2 \in {}^{V}A$ таких, що $d_1 =_{-z} d_2$, маємо $P(d_1) = P(d_2)$.

Композиції квазіарних предикатів

На пропозиційному рівні композиції фактично працюють

лише з виробленими предикатами істиннісними значеннями. Такі композиції називають логічними зв'язками.

Основними логічними зв'язками є $\neg$, $\vee$, &, а також $\to$, $\leftrightarrow$.

Дамо визначення $\neg$, $\vee$, & через області істинності та хибності відповідних предикатів. Предикати $\neg(P)$, $\vee(P, Q)$, $\&(P, Q)$ позначаємо $\neg P$, $P \vee Q$, $P \& Q$. Вони задаються так.

$T(\neg P) = F(P)$;

$F(\neg P) = T(P)$;

$T(P \vee Q) = T(P) \cup T(Q)$; $F(P \vee Q) = F(P) \cap F(Q)$;

$T(P \& Q) = T(P) \cap T(Q)$; $F(P \& Q) = F(P) \cup F(Q)$.

$\neg$ та $\vee$ назвемо базовими пропозиційними композиціями. Композиції &, $\to$, $\leftrightarrow$ є похідними, вони виражаються через $\neg$ та $\vee$:

$P \& Q = \neg(\neg P \vee \neg Q)$; $P \to Q = \neg P \vee Q$; $P \leftrightarrow Q = (P \to Q) \& (Q \to P)$.

На рівні чистих першопорядкових логік (на кванторному рівні) до логічних зв'язок і реномінацій додаються 1-арні параметричні композиції *реномінації* $\mathrm{R}_{\bar{x}}^{\bar{v}}$ та *квантифікації* $\exists x$ і $\forall x$.

Композиція $\mathrm{R}_{\bar{x}}^{\bar{v}}$ задається так:

для кожного $d \in {}^{V}A$ маємо $\mathrm{R}_{\bar{x}}^{\bar{v}}(P)(d) = P(\mathrm{r}_{\bar{x}}^{\bar{v}}(d))$.

Композицію $\mathrm{R}_{\bar{x}}^{\bar{v}}$ можна визначити через області істинності та хибності відповідного предиката:

$$T(\mathrm{R}_{\bar{x}}^{\bar{v}}(P)) = \{d \in {}^{V}A \mid \mathrm{r}_{\bar{x}}^{\bar{v}}(d) \in T(P)\} = (\mathrm{r}_{\bar{x}}^{\bar{v}})^{-1}(T(P)) ;$$

$$F(\mathrm{R}_{\bar{x}}^{\bar{v}}(P)) = \{d \in {}^{V}A \mid \mathrm{r}_{\bar{x}}^{\bar{v}}(d) \in F(P)\} = (\mathrm{r}_{\bar{x}}^{\bar{v}})^{-1}(F(P)) .$$

Композиції $\exists x$ і $\forall x$ задамо через області істинності та хибності предикатів $\exists xP$ і $\forall xP$;

$T(\exists xP) = \{d \in {}^{V}A \mid T \in P(d\nabla x \mapsto a)$ для деякого $a \in A\}$;

$F(\exists xP) = \{d \in {}^{V}A \mid F \in P(d\nabla x \mapsto a)$ для всіх $a \in A\}$;

$T(\forall xP) = \{d \in {}^{V}A \mid T \in P(d\nabla x \mapsto a)$ для всіх $a \in A\}$;

$F(\forall xP) = \{d \in {}^{V}A \mid F \in P(d\nabla x \mapsto a)$ для деякого $a \in A\}$.

Композиції $\neg$, $\vee$, $\mathrm{R}_{\bar{x}}^{\bar{v}}$, $\exists x$ – це базові композиції логік кванторного рівня. Композиція $\forall x$ є похідною: маємо $\forall xP = \neg\exists x\neg P$.

Результатом послідовного виконання двох композицій $\mathrm{R}_{\bar{y}}^{\bar{w}}$ (застосовується першою) та $\mathrm{R}_{\bar{x}}^{\bar{v}}$ (застосовується другою) є їх згортка – композиція реномінації $\mathrm{R}_{\bar{x}}^{\bar{v}} \circ \mathrm{R}_{\bar{y}}^{\bar{w}}$, яка визначається так:

$\mathrm{R}^{\overline{v}}_{\overline{x}} \circ^{\overline{w}}_{\overline{y}} (P)(d) = P(\mathrm{r}^{\overline{w}}_{\overline{y}} \square^{\overline{v}}_{\overline{x}}(d))$ для кожного $d \in {}^{V}A$.

Основні властивості композицій реномінації:

– $\mathrm{R}^{z,\overline{v}}_{z,\overline{x}}(P) = \mathrm{R}^{\overline{v}}_{\overline{x}}(P)$ – згортка тотожної пари імен;

– $\mathrm{R}^{\overline{v}}_{\overline{x}}(\neg P) = \neg \mathrm{R}^{\overline{v}}_{\overline{x}}(P)$ – R¬-дистрибутивність;

– $\mathrm{R}^{\overline{v}}_{\overline{x}}(P \vee Q) = \mathrm{R}^{\overline{v}}_{\overline{x}}(P) \vee \mathrm{R}^{\overline{v}}_{\overline{x}}(Q)$ – R∨-дистрибутивність;

– $\mathrm{R}^{\overline{v}}_{\overline{x}}(P \,\&\, Q) = \mathrm{R}^{\overline{v}}_{\overline{x}}(P) \,\&\, \mathrm{R}^{\overline{v}}_{\overline{x}}(Q)$ – R&-дистрибутивність;

– $\mathrm{R}^{z,\overline{v}}_{y,\overline{x}}(P) = \mathrm{R}^{\overline{v}}_{\overline{x}}(P)$ за умови: $z \in V$ неістотне для P.

Традиційні властивості композицій $\exists x$ та $\forall x$:

– $\exists x \exists y P = \exists y \exists x P$ та $\forall x \forall y P = \forall y \forall x P$;

– $\neg \exists x P = \forall x \neg P$ та $\neg \forall x P = \exists x \neg P$;

– $\exists x \exists x P = \exists x P$; $\exists x \forall x P = \forall x P$; $\forall x \exists x P = \exists x P$; $\forall x \forall x P = \forall x P$;

– $\exists x P \vee \exists x Q = \exists x (P \vee Q)$ та $\forall x P \,\&\, \forall x Q = \forall x (P \& Q)$.

Залучаючи до розгляду реномінації, маємо властивості:

– $\exists y P = \exists z \mathrm{R}^{y}_{z}(P)$, якщо z неістотне для P;

– $\mathrm{R}^{\overline{v}}_{\overline{x}}(\exists y P) = \exists y \mathrm{R}^{\overline{v}}_{\overline{x}}(P)$, якщо $y \notin \{\overline{v}, \overline{x}\}$;

– $\mathrm{R}^{\overline{v}}_{\overline{x}}(\exists y P) = \exists z \mathrm{R}^{\overline{v}}_{\overline{x}} \circ^{y}_{z}(P)$, якщо z неістотне для P, $z \notin \{\overline{v}, \overline{x}\}$;

– $\mathrm{R}^{\overline{u},x}_{\overline{v},y}(\exists x P) = \mathrm{R}^{\overline{u}}_{\overline{v}}(\exists x P)$; зокрема: $\mathrm{R}^{x}_{y}(\exists x P) = \exists x P$.

Ці властивості можна переформулювати для $\forall x$.

Квазіарні реляції

Квазіарна реляція (відношення) – це довільна $L \subseteq {}^{V}A$.

Для квазіарних реляцій як множин ІМ вводимо теоретико-множинні операції: об'єднання $\cup$, перетин $\cap$, доповнення $\sim$. Вводимо також номінативні операції реномінації та квантифікації.

Операція реномінації $\rho^{\overline{v}}_{\overline{x}}$ індукована відповідною операцією реномінації ІМ $\mathrm{r}^{\overline{v}}_{\overline{x}}$:

$$\rho^{\overline{v}}_{\overline{x}}(L) = \{d \in {}^{V}A \mid \mathrm{r}^{\overline{v}}_{\overline{x}}(d) \in L\} = (\mathrm{r}^{\overline{v}}_{\overline{x}})^{-1}(L).$$

Згортка композицій реномінації $\rho^{\overline{w}}_{\overline{y}}$ (застосовується першою) та $\rho^{\overline{v}}_{\overline{x}}$ (застосовується другою) – це композиція реномінації $\rho^{\overline{w}}_{\overline{y}} \circ^{\overline{v}}_{\overline{x}}$, вона визначається так:

$$\rho^{\overline{v}}_{\overline{x}}(\rho^{\overline{w}}_{\overline{y}}(L)) = \rho^{\overline{w}}_{\overline{y}} \circ^{\overline{v}}_{\overline{x}}(L) = (\mathrm{r}^{\overline{w}}_{\overline{y}} \square^{\overline{v}}_{\overline{x}})^{-1}(L).$$

Операції квантифікації $\exists x$ та $\forall x$ індуковані відповідними композиціями квазіарних предикатів $\exists x$ та $\forall x$. Задаємо їх так:

$$\exists x(L) = \{d \in {}^{V}A \mid d \nabla x \mapsto a) \in L \text{ для деякого } a \in A\};$$

$\forall x(L) = \{d \in {}^{V}A \mid d\nabla x \mapsto a) \in L$ для всіх $a \in A\}$.

Ім'я $z \in V$ *неістотне* для квазіарної реляції L, якщо

для всіх $d_1, d_2 \in {}^{V}A$ таких, що $d_1 =_{-z} d_2$, маємо: $d_1 \in L \Leftrightarrow d_2 \in L$.

Властивості квазіарних реляцій індуковані відповідними властивостями квазіарних предикатів.

Композиціям $\neg$, $\vee$, & для предикатів відповідають операції $\sim$, $\cup$, $\cap$ для квазіарних реляцій – *областей істинності* цих предикатів. Звідси отримуємо традиційні властивості операцій $\sim$, $\cup$, $\cap$ булевої алгебри множин.

1) Комутативність $\cup$ та $\cap$:

$L \cup M = M \cup L$;

$L \cap M = M \cap L$.

2) Асоціативність $\cup$ та $\cap$:

$(L \cup M) \cup R = L \cup (M \cup R)$;

$(L \cap M) \cap R = L \cap (M \cap R)$.

3) Дистрибутивність $\cup$ відносно $\cap$ та $\cap$ відносно $\cup$:

$(L \cup M) \cap R = (L \cap R) \cup (M \cap R)$;

$(L \cap M) \cup R = (L \cup R) \cap (M \cup R)$.

4) Зняття подвійного заперечення:

$\sim\sim L = L$.

5) Ідемпотентність $\cup$ та $\cap$:

$L = L \cup L$;

$L = L \cap L$.

6) Закони де Моргана:

$\sim(L \cup M) = (\sim M) \cap (\sim L)$;

$\sim(L \cap M) = (\sim M) \cup (\sim L)$.

Для операції реномінації маємо:

– $\rho^{z,\bar{v}}_{z,\bar{x}}(L) = \rho^{\bar{v}}_{\bar{x}}(L)$;

– $\rho^{\bar{v}}_{\bar{x}}(\sim L) = \sim \rho^{\bar{v}}_{\bar{x}}(L)$;

– $\rho^{\bar{v}}_{\bar{x}}(L \cup M) = \rho^{\bar{v}}_{\bar{x}}(L) \cup \rho^{\bar{v}}_{\bar{x}}(M)$;

– $\rho^{\bar{v}}_{\bar{x}}(L \cap M) = \rho^{\bar{v}}_{\bar{x}}(L) \cap \rho^{\bar{v}}_{\bar{x}}(M)$;

– $\rho^{z,\bar{v}}_{y,\bar{x}}(L) = \rho^{\bar{v}}_{\bar{x}}(L)$ за умови: $z \in V$ неістотне для L.

Для операцій квантифікації маємо:

– $\exists x(\exists y(L)) = \exists y(\exists x(L))$ та $\forall x(\forall y(L)) = \forall y(\forall x(L))$;

– $\sim(\exists x(L)) = \forall x(\sim(L))$ та $\sim(\forall x(L)) = \exists x(\sim(L))$;

– $\exists x(\exists x(L)) = \exists x(L)$; $\exists x(\forall x(L)) = \forall x(L)$;

– $\forall x(\exists x(L)) = \exists x(L)$; $\forall x(\forall x(L)) = \forall x(L)$;

– $\exists x(L) \cup \exists x(M) = \exists x(L \cup M)$ та $\forall x(L) \cap \forall x(M) = \forall x(L \cap M)$.

Залучаючи до розгляду реномінації, маємо:

– $\exists y(L) = \exists y(\rho_z^y(L))$, якщо z неістотне для L;

– $\rho_{\bar{x}}^{\bar{v}}(\exists y(L)) = \exists y(\rho_{\bar{x}}^{\bar{v}}(L)$, якщо $y \notin \{\bar{v}, \bar{x}\}$;

– $\rho_{\bar{x}}^{\bar{v}}(\exists y(L)) = \exists z(\rho_{\bar{x}}^{\bar{v}} \circ_z^y (L))$, якщо z неістотне для P, $z \notin \{\bar{v}, \bar{x}\}$;

– $\rho_{\bar{v},y}^{\bar{u},x}(\exists x(L)) = \rho_{\bar{v}}^{\bar{u}}(\exists x(L))$; зокрема: $\rho_y^x(\exists x(L)) = \exists x(L)$.

Наведені властивості можна переформулювати для $\forall x$.

Алгебри квазіарних та бі-квазіарних реляцій

Носієм алгебри квазіарних реляцій є множина квазіарних реляцій, а сигнатура (множина символів базових операцій) – це $\{\neg, \vee, \&, R_{\bar{x}}^{\bar{v}}, \exists x, \forall x\}$.

Стандартна інтерпретація сигнатурних символів $\neg, \vee, \&$, $R_{\bar{x}}^{\bar{v}}, \exists x, \forall x$ на множині квазіарних реляцій – це відповідно операції $\sim, \cup, \cap, \rho_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$. Квазіарні реляції – елементи носія алгебри – трактуємо як області *істинності* квазіарних предикатів.

Отимуємо *стандартну* алгебру квазіарних реляцій $A_{QR} = (2^{V_A}; I_{QR})$, де I_{QR} – відображення стандартної інтерпретації.

Дуальна інтерпретація сигнатурних символів $\neg, \vee, \&, R_{\bar{x}}^{\bar{v}}, \exists x$, $\forall x$ на множині квазіарних реляцій – це відповідно $\sim, \cap, \cup, \rho_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$. У цьому випадку елементи носія алгебри трактуємо як області *хибності* квазіарних предикатів.

Тепер отримуємо *дуальну* алгебру квазіарних реляцій $A_{QRD} = (2^{V_A}; I_{QRD})$, де – відображення дуальної інтерпретації.

Теорема 1. Алгебри A_{QR} та A_{QRD} ізоморфні.

Кожний тотальний однозначний квазіарний предикат цілком задається областю істинності, область хибності є доповненням до області істинності. Дія композицій $\neg$, $\vee$, $\&$, $R_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$ на предикати відповідає дії операцій $\sim, \cup, \cap, \rho_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$ на квазіарні реляції – області їх істинності.

Теорема 2. Алгебра A_{QR} ізоморфна алгебрі тотальних однозначних квазіарних предикатів $A_{STP} = (Pr_{ST}^A; I_{CP})$.

Тут I_{CP} – відображення інтерпретації сигнатурних символів $\neg, \vee, \&, R_{\bar{x}}^{\bar{v}}, \exists x, \forall x$ на множину композицій $\neg$, $\vee$, $\&$, $R_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$ квазіарних предикатів.

Для задання нетотального чи неоднозначного квазіарного

предиката необхідно вказувати як область його істинності, так і область хибності. Дія композицій $\neg$, $\vee$, $\&$, $\mathrm{R}_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$ на квазіарні предикати відповідає дії операцій $\sim, \cup, \cap, \rho_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$ на області їх істинності та дії операцій $\sim, \cap, \cup, \rho_{\bar{x}}^{\bar{v}}$, $\forall x$, $\exists x$ на області їх хибності.

Таким чином, дія композиції на квазіарний предикат рівносильна дії відповідної операції та дуальної до неї до двох квазіарних реляцій – областей істинності та хибності.

Побудуємо тепер алгебри, визначені на множинах пар квазіарних реляцій – алгебри бі-квазіарних реляцій. Такі алгебри мають вигляд $A_{BQR} = (2^{{}^{V}A} \times 2^{{}^{V}A};\ I_{BQR})$, де I_{BQR} – відображення інтерпретації сигнатурних символів. При цій інтерпретації сигнатурні операції $\neg_B, \vee_B, \&_B, R_{\bar{x}\,B}^{\bar{v}}, \exists x_B, \forall x_B$, відповідні символам $\neg, \vee, \&, R_{\bar{x}}^{\bar{v}}, \exists x, \forall x$, діють на парах квазіарних реляцій так:

– як операції $\sim, \cup, \cap, \rho_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$ на першій компоненті пари (як стандартні);

– як операції $\sim, \cap, \cup, \rho_{\bar{x}}^{\bar{v}}$, $\forall x$, $\exists x$ на другій компоненті пари (як дуальні).

Тому визначаємо:

$\neg_B(L, M) = (M, L)$;

$(L, M) \vee_B (R, S) = (L \cup R, M \cap S)$;

$(L, M) \&_B (R, S) = (L \cap R, M \cup S)$;

$R_{\bar{x}\,B}^{\bar{v}}(L, M) = (\rho_{\bar{x}}^{\bar{v}}(L), \rho_{\bar{x}}^{\bar{v}}(M))$;

$\exists_B x\,(L, M) = (\exists x(L),\ \forall x(M))$;

$\forall_B x\,(L, M) = (\forall x(L),\ \exists x(M))$.

Бі-квазіарна реляція $(L, M) \subseteq {}^{V}A \times {}^{V}A$:

– однозначна, якщо $L \cap M = \varnothing$;

– тотальна, якщо $L \cup M = {}^{V}A$.

Класи тотальних квазіарних предикатів та однозначних квазіарних предикатів замкнені щодо композицій $\neg$, $\vee$, $\&$, $\mathrm{R}_{\bar{x}}^{\bar{v}}$, $\exists x$, $\forall x$. Звідси отримуємо:

Теорема 4. Класи однозначних бі-квазіарних реляцій та тотальних бі-квазіарних реляцій замкнені щодо сигнатурних операцій $\neg_B, \vee_B, \&_B, R_{\bar{x}\,B}^{\bar{v}}, \exists x_B, \forall x_B$.

Таким чином, виділяємо наступні підалгебри алгебри A_{BQR}:

$A_{SBQR} = (SR; I_{BQR})$; $A_{TBQR} = (TR; I_{BQR})$; $A_{STQR} = (STR; I_{BQR})$.

Тут SR – клас однозначних бі-квазіарних реляцій,

TR – клас тотальних бі-квазіарних реляцій,

STR – клас однозначних тотальних бі-квазіарних реляцій.

Зауважимо, що елементи *STR* мають вигляд $(L, \sim L)$.

Задамо відображення дуалізації $\delta : 2^{V_A} \times 2^{V_A} \to 2^{V_A} \times 2^{V_A}$ так: $\delta(L, M) = (\sim M, \sim L)$.

Теорема 4. Відображення дуалізації δ:

1) є автоморфізмом алгебри A_{BQR};

2) є ізоморфізмом алгебри A_{SBQR} на алгебру A_{TBQR};

3) є ізоморфізмом алгебри A_{TBQR} на алгебру A_{SBQR};

4) є тотожним автоморфізмом алгебри A_{STQR}.

Теорема 5. 1) Алгебра A_{BQR} ізоморфна алгебрі часткових неоднозначних квазіарних предикатів;

2) алгебра A_{SBQR} ізоморфна алгебрі однозначних квазіарних предикатів;

3) алгебра A_{TBQR} ізоморфна алгебрі тотальних квазіарних предикатів;

4) алгебра A_{STQR} ізоморфна алгебрі однозначних тотальних квазіарних предикатів.

Висновки

В роботі побудовано та досліджено низку алгебр квазіарних реляцій (відношень), розглянуто їх зв'язки із алгебрами квазіарних предикатів. На множині всіх квазіарних реляцій природним чином задаються булеві операції об'єднання, перетину, доповнення, а також спеціальні номінативні операції реномінації та квантифікації. Встановлено ізоморфізм алгебри квазіарних реляцій та першопорядкової алгебри тотальних однозначних квазіарних предикатів.

Побудовано алгебри бі-квазіарних реляцій, задані на множинах пар квазіарних реляцій. Визначено різні підкласи таких алгебр та досліджено їх зв'язки з алгебрами часткових однозначних, тотальних неоднозначних та часткових неоднозначних предикатів.

Список використаних джерел

1. Глушков В. М. Алгебра, языки, программирование / В. М. Глушков, Г. Е. Цейтлин, Е. Л. Ющенко. – Київ: Наукова думка, 1974. – 328 с.
2. Нікітченко М.С. Математична логіка та теорія алгоритмів / М.С. Нікітченко, С.С. Шкільняк. – Київ: ВПЦ Київський університет, 2008. – 528 с.
3. Нікітченко М.С. Прикладна логіка / М.С. Нікітченко, С.С. Шкільняк. – Київ: ВПЦ Київський університет, 2013. – 278 с.

ЧИСТІ ПЕРШОПОРЯДКОВІ ЛОГІКИ ЧАСТКОВИХ ТА НЕОДНОЗНАЧНИХ ПРЕДИКАТІВ

М.С. Нікітченко, О.С. Шкільняк, С.С. Шкільняк
Київський національний університет імені Тараса Шевченка,
Україна
nikitchenko@unicyb.kiev.ua, me.oksana@gmail.com,
sssh@unicyb.kiev.ua

Вступ

Успішне вирішення задачі створення ефективних і надійних програмних систем неможливе без широкого використання понять і методів математичної логіки. Різноманітні логічні системи, які успішно використовуються в інформатиці й програмуванні, зазвичай базуються на класичній логіці предикатів. Проте поява нових застосувань висвітила принципові обмеження класичної логіки, які ускладнюють її використання. Тому на перший план висувається проблема побудови нових, програмно-орієнтованих логік. Природною основою такої побудови є спільний для логіки й програмування композиційно-номінативний підхід. На його базі розроблено (див., напр., [1–6]). низку різноманітних логічних систем на різних рівнях абстрактності й загальності. Логіки, збудовані на основі цього підходу, названо композиційно-номінативними.

В даній роботі досліджено низку класів чистих першопорядкових композиційно-номінативних логік (ЧКНЛ) часткових однозначних, тотальних неоднозначних і часткових неоднозначних предикатів. Розглянуто ЧКНЛ базового рівня та їх розширення:

– ЧКНЛРР – це ЧКНЛ із розширеними реномінаціями;

– ε-ЧКНЛ – це ЧКНЛ із предикатами-індикаторами наявності значення для змінних;

– ε-ЧКНЛРР, які поєднують можливості ε-ЧКНЛ і ЧКНЛРР.

Описано мови та семантичні моделі таких логік, розглянуто різні формалізації відношення логічного наслідку для пар та для множин формул. Для різних класів ЧКНЛ запропоновано спектр числень секвенційного типу.

Поняття, які тут не визначаються, тлумачимо в сенсі [1, 2, 5].

Композиції чистих першопорядкових квазіарних предикатів

V-іменна множина (V-ІМ) над A – це часткова однозначна функція вигляду $V\to A$. V-ІМ подаємо як $[v_1\mapsto a_1,...,v_n\mapsto a_n,...]$, де $v_i\in V$, $a_i\in A$. Операція реномінації $\mathrm{r}^{v_1,...,v_n}_{x_1,...,x_n}: {}^{V}A\to {}^{V}A$ визначається [2]

так: $\mathrm{r}^{v_1,\ldots,v_n}_{x_1,\ldots,x_n}(d) = d\,\|_{-\{v_1,\ldots,v_n\}} \cup [v_1 \mapsto d(x_1),\ldots,v_n \mapsto d(x_n)]$. Тут $\|_{-X}$ – операція вилучення компонент з іменами $X \subseteq V$: $d\|_{-X} = [v \mapsto a \in d \mid v \notin X]$.

Узагальненням операції реномінації є операція розширеної реномінації, яка дає змогу явно задавати відсутність значення для предметних імен, тобто явно вилучати компоненти з певними іменами. Відсутність значення для імені x задаємо парою, де верхнє ім'я – x, а відповідне нижнє ім'я – спеціальний символ $\perp$. Операція розширеної реномінації $\mathrm{r}^{v_1,\ldots,v_n,u_1,\ldots,u_m}_{x_1,\ldots,x_n,\perp,\ldots,\perp} : {}^{V}A \to {}^{V}A$ визначається [5] так:

$$\mathrm{r}^{v_1,\ldots,v_n,u_1,\ldots,u_m}_{x_1,\ldots,x_n,\perp,\ldots,\perp}(d) = d\,\|_{-\{v_1,\ldots,v_n,u_1,\ldots,u_m\}} \cup [\mathrm{v_1} \mapsto \mathrm{d(x_1)},\ldots,\mathrm{v_n} \mapsto \mathrm{d(x_n)}].$$

Замість $\mathrm{r}^{v_1,\ldots,v_n}_{x_1,\ldots,x_n}$ та $\mathrm{r}^{v_1,\ldots,v_n,u_1,\ldots,u_m}_{x_1,\ldots,x_n,\perp,\ldots,\perp}$ скорочено пишемо $\mathrm{r}^{\bar{v}}_{\bar{x}}$ та $\mathrm{r}^{\bar{v},\bar{u}}_{\bar{x},\perp}$.

Послідовне застосування двох операцій реномінації можна подати у вигляді однієї, яку назвемо згорткою початкових (див. [1, 5]). Згортку операцій $\mathrm{r}^{\bar{v}}_{\bar{x}}$ (застосовується першою) та $\mathrm{r}^{\bar{z}}_{\bar{y}}$ (застосовується другою) позначаємо $\mathrm{r}^{\bar{v}}_{\bar{x}} \square \mathrm{r}^{\bar{z}}_{\bar{y}}$; тоді $r^{\bar{z}}_{\bar{y}}(r^{\bar{v}}_{\bar{x}}(d)) = r^{\bar{z}}_{\bar{y}} \square {}^{\bar{v}}_{\bar{x}}(d)$. Для розширених реномінацій згортку $r^{\bar{v},\bar{t}}_{\bar{x},\perp}$ та $r^{\bar{z},\bar{u}}_{\bar{y},\perp}$ позначимо $r^{\bar{z},\bar{u}}_{\bar{y},\perp} \square {}^{\bar{v},\bar{t}}_{\bar{x},\perp}$ (визначення див. [5]). Тоді $r^{\bar{z},\bar{u}}_{\bar{y},\perp}(r^{\bar{v},\bar{t}}_{\bar{x},\perp}(d)) = r^{\bar{z},\bar{u}}_{\bar{y},\perp} \square {}^{\bar{v},\bar{t}}_{\bar{x},\perp}(d)$.

Часткові неоднозначні предикати на множині ${}^{V}A$ ми трактуємо як відношення між ${}^{V}A$ та $\{T, F\}$. Такі предикати назвемо предикатами реляційного типу. У цьому випадку $P(d)$ позначає множину тих значень, які предикат $P : {}^{V}A \to \{T, F\}$ може прийняти на $d \in {}^{V}A$.

Кожний квазіарний предикат однозначно задається своїми областями істинності та хибності. Це множини

$T(P) = \{d \in {}^{V}A \mid T \in P(d)\}$;
$F(P) = \{d \in {}^{V}A \mid F \in P(d)\}$.

V-квазіарний предикат $P : {}^{V}A \to \{T, F\}$ назвемо:
– неспростовним, якщо $F(P) = \varnothing$;
– виконуваним, якщо $T(P) \neq \varnothing$;
– тотально істинним, якщо $T(P) = {}^{V}A$;
– тотально хибним, якщо $F(P) = {}^{V}A$;
– тотожно істинним, якщо $T(P) = {}^{V}A$ та $F(P) = \varnothing$;
– тотожно хибним, якщо $T(P) = \varnothing$ та $F(P) = {}^{V}A$);
– всюди невизначеним, якщо $T(P) = F(P) = \varnothing$;
– тотально насиченим, якщо $T(P) = {}^{V}A$ та $F(P) = {}^{V}A$.

Квазіарний предикат P:
– монотонний, якщо $d \subseteq d' \Rightarrow P(d) \subseteq P(d')$.
– антитонний, якщо $d \subseteq d' \Rightarrow P(d) \supseteq P(d')$.

Базовими композиціями ЧКНЛ є пропозиційні композиції (логічні зв'язки) $\neg$ і $\vee$, реномінації $\mathrm{R}^{\bar{v}}_{\bar{x}}$, квантори $\exists x$. Задамо їх через області істинності та хибності відповідних предикатів:

$$T(\neg P) = F(P);\ F(\neg P) = T(P);$$
$$T(P \vee Q) = T(P) \cup T(Q);\ F(P \vee Q) = F(P) \cap F(Q);$$
$$T(\mathrm{R}^{\bar{v}}_{\bar{x}}(P)) = \{d \in {}^{V}A \mid r^{\bar{v}}_{\bar{x}}(d) \in T(P)\};$$
$$F(\mathrm{R}^{\bar{v}}_{\bar{x}}(P)) = \{d \in {}^{V}A \mid r^{\bar{v}}_{\bar{x}}(d) \in F(P)\};$$
$$T(\exists xP) = \{d \in {}^{V}A \mid T \in P(d\nabla x \mapsto a) \text{ для деякого } a \in A\};$$
$$F(\exists xP) = \{d \in {}^{V}A \mid F \in P(d\nabla x \mapsto a) \text{ для всіх } a \in A\}.$$

Подібним чином задаємо [2] логічні операції $\rightarrow$, $\&$, $\leftrightarrow$, $\forall x$.

Логічні зв'язки $\rightarrow$, $\&$, $\leftrightarrow$ є похідними, вони виражаються через $\neg$ та $\vee$. Композиція $\forall x$ теж є похідною: маємо $\forall xP = \neg\exists x\neg P$.

Основні властивості наведених композицій $\neg$, $\vee$, $\rightarrow$, $\&$, $\leftrightarrow$ та $\exists x$, $\forall x$ в цілому аналогічні властивостям відповідних класичних логічних зв'язок та кванторів.

ЧКНЛ із розширеними реномінаціями запропоновано в [3], вони названі ЧКНЛРР. Базовими композиціями ЧКНЛРР є $\neg$, $\vee$, $\exists x$ та розширені реномінації $\mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}$. Композиції $\mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}$ визначаються так:

$$T(\mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}(P)) = \{d \in {}^{V}A \mid \mathrm{r}^{\bar{v},\bar{u}}_{\bar{x},\perp}(d) \in T(P)\};$$
$$F(\mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}(P)) = \{d \in {}^{V}A \mid \mathrm{r}^{\bar{v},\bar{u}}_{\bar{x},\perp}(d) \in F(P)\}.$$

За допомогою розширених реномінацій визначаємо похідні композиції розширеної квантифікації (розширені квантори):

$$\exists_{\perp} xP = \exists xP \vee \mathrm{R}^{x}_{\perp}(P);\quad \forall_{\perp} xP = \forall xP \,\&\, \mathrm{R}^{x}_{\perp}(P).$$

Основні властивості розширених кванторів наведено в [3, 5].

Характерною особливістю квазіарних предикатів є те, що значення $P(d)$ може бути різним залежно від того, входить чи не входить до d компонента з певним іменем. Тому для логік квазіарних предикатів вже невірні деякі важливі закони класичної логіки (див [1, 2]). Наприклад, предикати вигляду $\forall xP \rightarrow P$ та $P \rightarrow \exists xP$ не завжди неспростовні. Тому доцільно явно вказувати означені й неозначені предметні імена. Для цього використовуємо спеціальні 0-арні композиції – предикати-індикатори εz, які визначають наявність в даних компоненти з відповідним іменем (змінною) z, тобто наявність значення для z. Предикати-індикатори εz визначаємо так:

$$T(\varepsilon z) = \{d \mid d(z)\uparrow\};\ F(\varepsilon z) = \{d \mid d(z)\downarrow\}.$$

Предикати-індикатори εz немонотонні та неантитонні.

ЧКНЛ з виділеними предикатами-індикаторами названо [4]

ε-ЧКНЛ. Базовими композиціями ε-ЧКНЛ є $\neg$, $\vee$, $\mathrm{R}^{\bar{v}}_{\bar{x}}$, $\exists x$ та εz.

Можливості ЧКНЛРР та ε-ЧКНЛ поєднують ε-ЧКНЛРР, які розглянуто в [5]. Базові композиції ε-ЧКНЛРР: $\neg, \vee, \mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}, \exists x, \varepsilon z$.

Теорема 1. 1) Композиції $\neg$, $\vee$, $\mathrm{R}^{\bar{v}}_{\bar{x}}$, $\mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}$, $\exists x$ зберігають монотонність і антитонність квазіарних предикатів.

2) $\neg$, $\vee$, $\mathrm{R}^{\bar{v}}_{\bar{x}}$, $\mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}$, $\exists x$, εz зберігають тотальність предикатів.

Звідси отримуємо, що всюди невизначений предикат $\perp$ не може бути виражений за допомогою εy, $\neg$, $\vee$, $\mathrm{R}^{\bar{v}}_{\bar{x}}$, $\mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}$, $\exists x$.

Таким чином, можна отримати нетривіальні розширення ЧКНЛ, ЧКНЛРР, ε-ЧКНЛ та ε-ЧКНЛРР шляхом додавання $\perp$ як спеціальної 0-арної композиції. Такі розширені логіки назвемо відповідно $\mathrm{ЧКНЛ_U}$, $\mathrm{ЧКНЛРР_U}$, $\mathrm{ε\text{-}ЧКНЛ_U}$ та $\mathrm{ε\text{-}ЧКНЛРР_U}$. Для зазначених логік можна розглядати лише неокласичну семантику та загальну семантику, адже всюди невизначений предикат $\perp$ нетотальний.

Дослідження $\mathrm{ЧКНЛ_U}$, $\mathrm{ЧКНЛРР_U}$, $\mathrm{ε\text{-}ЧКНЛ_U}$ та $\mathrm{ε\text{-}ЧКНЛРР_U}$ буде проведене в наступних роботах.

Семантичні моделі та мови ЧКНЛ

Семантичними моделями ЧКНЛ є [1, 2] композиційні системи квазіарних предикатів $({}^{V}A, Pr^{A}, C)$, де Pr^{A} – клас V-квазіарних предикатів на A, C визначається базовими композиціями. Терми композиційної алгебри (Pr^{A}, C) трактуємо як формули мови. Алфавіт мови: символи базових композицій, множини V предметних імен та Ps предикатних символів (сигнатура мови). Символи базових композицій: $\neg$, $\vee, R^{\bar{v}}_{\bar{x}}, \exists x$ для ЧКНЛ; $\neg, \vee, R^{\bar{v},\bar{u}}_{\bar{x},\perp}, \exists x$ для ЧКНЛРР; $\neg$, $\vee, R^{\bar{v}}_{\bar{x}}$, $\exists x$, εz для ε-ЧКНЛ; $\neg$, $\vee, R^{\bar{v},\bar{u}}_{\bar{x},\perp}, \exists x$, εz для ε-ЧКНЛРР.

Множина Fr формул мови ε-ЧКНЛРР визначається так:

– $Ps \subseteq Fr$ та $\{\varepsilon z \mid z \in V\} \subseteq Fr$; формули $p \in Ps$ та εz – атомарні;

– $\Phi, \Psi \in Fr \Rightarrow \neg\Phi, \vee\Phi\Psi, R^{\bar{v},\bar{u}}_{\bar{x},\perp}\Phi, \exists x\Phi \in Fr$.

Множини формул мови ЧКНЛ, ε-ЧКНЛ, ЧКНЛРР визначаємо аналогічно.

Відображення інтерпретації формул $I : Fr \to Pr^{A}$ визначаємо за допомогою тотального однозначного $I : Ps \to Pr^{A}$, яке позначає символами Ps базові предикати. Для складних формул відображення $I : Fr \to Pr^{A}$ задається індуктивно:

– $I(\neg\Phi) = \neg(I(\Phi))$; $I(\vee\Phi\Psi) = \vee(I(\Phi), I(\Psi))$;

– $I(R^{\bar{v},\bar{u}}_{\bar{x},\perp}\Phi) = \mathrm{R}^{\bar{v},\bar{u}}_{\bar{x},\perp}(I(\Phi))$; $I(\exists x\Phi) = \exists x(I(\Phi))$.

Відображення I пов'язує алгебру даних (A, Pr) із мовою. Отримуємо об'єкт $((A, Pr^A), I)$ – алгебраїчну систему (АС) з доданою сигнатурою, яка визначає композиційну систему $({}^V\!A, Pr^A, C)$. АС з доданою сигнатурою є інтегрованими семантичними моделями, які пов'язують мови КНЛ із алгебрами даних.

Назвемо такі АС моделями мови та позначатимемо $\boldsymbol{A} = (A, I)$.

Предикат $I(\Phi)$ – значення формули Ф при інтерпретації на $\boldsymbol{A}$, – позначаємо Φ_A. Формули εz завжди інтерпретуються однаково, тому предикати-індикатори, які є їх значенням, теж позначаємо εz.

Формула Ф частково істинна при інтерпретації на $\boldsymbol{A} = (A, I)$, або $\boldsymbol{A}$-неспростовна, якщо Φ_A – неспростовний предикат. Формула Ф неспростовна, якщо Ф $\boldsymbol{A}$-неспростовна на кожній моделі мови $\boldsymbol{A}$.

Подібним чином даємо визначення формули: тотожно істинної на $\boldsymbol{A}$ та тотожно істинної, виконуваної на $\boldsymbol{A}$ та виконуваної.

Характерним для програмування і моделювання є використання часткових та необов'язково однозначних відображень над складними даними. Тому постає проблема дослідження КНЛ із нетрадиційними семантиками. Класична логіка – це логіка тотальних однозначних предикатів. КНЛ однозначних часткових предикатів – це логіки з неокласичною семантикою, КНЛ тотальних неоднозначних предикатів –логіки з пересиченою семантикою, КНЛ часткових неоднозначних предикатів – логіки із загальною семантикою.

Модель мови $\boldsymbol{B} = (A, I_B)$ дуальна до моделі мови $\boldsymbol{A} = (A, I_A)$, якщо для кожного $p \in Ps$ маємо $F(p_B) = \overline{T(p_A)}$.

Якщо $\boldsymbol{B}$ дуальна до $\boldsymbol{A}$, то $T(p_A) = \overline{F(p_B)}$ та $F(p_A) = \overline{T(p_B)}$, тобто $\boldsymbol{A}$ дуальна до $\boldsymbol{B}$. Якщо $\boldsymbol{A} = (A, I_A)$ – АС з частковими однозначними предикатами, то дуальна $\boldsymbol{B} = (A, I_B)$ – АС з тотальними неоднозначними предикатами, та навпаки.

Теорема 2. Якщо $\boldsymbol{B} = (A, I_B)$ дуальна до $\boldsymbol{A} = (A, I_A)$, то для кожної $\Phi \in Fr$ маємо $T(\Phi_B) = \overline{F(\Phi_A)}$ та $F(\Phi_B) = \overline{T(\Phi_A)}$. При цьому: Φ_A монотонний $\Rightarrow \Phi_B$ антитонний; Φ_A антитонний $\Rightarrow \Phi_B$ монотонний.

Наслідок 1. Φ_A неспростовний на $\boldsymbol{A}$ із частковими однозначними предикатами $\Leftrightarrow \Phi_B$ тотально істинний на дуальній $\boldsymbol{B}$ із тотальними неоднозначними предикатами. Цей означає: неокласична семантика та пересичена семантика дуальні.

Відношення логічного наслідку

Введемо відношення наслідку для двох формул при інтерпретації на фіксованій моделі мови $\boldsymbol{A}$. Це робиться однаково для ЧКНЛ базового рівня (див. [2]) та для їх розширень:

– неспростовнісний наслідок $_A|\!=_{Cl}$: $\Phi_A|\!=_{Cl}\Psi \Leftrightarrow T(\Phi_A)\cap F(\Psi_A)=\varnothing$;

– насичений наслідок $_A|\!=_{Cm}$: $\Phi_A|\!=_{Cm}\Psi \Leftrightarrow F(\Phi_A)\cup T(\Psi_A)={}^{V}A$;

– істиннісний наслідок $_A|\!=_{T}$: $\Phi_A|\!=_{T}\Psi \Leftrightarrow T(\Phi_A)\subseteq T(\Psi_A)$;

– хибнісний наслідок $_A|\!=_{F}$: $\Phi_A|\!=_{F}\Psi \Leftrightarrow F(\Psi_A)\subseteq F(\Phi_A)$;

– сильний наслідок $_A|\!=_{TF}$: $\Phi_A|\!=_{TF}\Psi \Leftrightarrow \Phi_A|\!=_{T}\Psi$ та $\Phi_A|\!=_{F}\Psi$.

Відношення логічного наслідку $|\!=_{Cl}$, $|\!=_{Cm}$, $|\!=_{T}$, $|\!=_{F}$, $|\!=_{TF}$ визначаємо за схемою: $\Phi|\!=_{*}\Psi \Leftrightarrow \Phi_A|\!=_{*}\Psi$ для кожної моделі мови $\boldsymbol{A}$.

Відношення логічного наслідку індукують на множині формул відношення логічної еквівалентності. Відношення еквівалентності $_A\sim_{Cl}$, $_A\sim_{Cm}$, $_A\sim_{T}$, $_A\sim_{F}$, $_A\sim_{TF}$ в моделі мови $\boldsymbol{A}$ визначаємо за такою схемою: $\Phi_A\sim_{*}\Psi$, якщо $\Phi_A|\!=_{*}\Psi$ та $\Psi_A|\!=_{*}\Phi$.

Звідси: $\Phi_A\sim_{TF}\Psi \Leftrightarrow T(\Phi_A)=T(\Psi_A)$ та $F(\Phi_A)=F(\Psi_A) \Leftrightarrow \Phi_A=\Psi_A$.

Відношення логічної еквівалентності $\sim_{Cl}$, $\sim_{Cm}$, $\sim_{T}$, $\sim_{F}$, $\sim_{TF}$ визначаємо за такою схемою: $\Phi\sim_{*}\Psi$, якщо $\Phi|\!=_{*}\Psi$ та $\Psi|\!=_{*}\Phi$.

Для $\sim_{Cl}$, $\sim_{Cm}$ та $\sim_{TF}$ маємо (тут $*$ – це одне з Cl, Cm, TF):

Теорема 3 (семантичної еквівалентності). Нехай Φ' отримана з Φ заміною деяких входжень $\Phi_1,...,\Phi_n$ на $\Psi_1,...,\Psi_n$. Якщо $\Phi_1\sim_{*}\Psi_1$, ..., $\Phi_n\sim_{*}\Psi_n$, то $\Phi\sim_{*}\Phi'$.

Для відношень $\sim_T$ та $\sim_F$ теорема 3 невірна (див. [2]).

Семантичні властивості формул мови ЧКНЛ в різних семантиках досліджено в [2]. Для розширень ЧКНЛ додаються властивості, пов'язані з розширеними реномінаціями та взаємодією предикатів-індикаторів із кванторами та реномінаціями (див. [3–6]).

Задамо відношення логічного наслідку для множин формул.

Нехай $\Gamma,\Delta\subseteq Fr$. У випадку $\Gamma_A|\!=\Delta$ позначаємо: $\bigcap\limits_{\Phi\in\Gamma}T(\Phi_A)$ як $T(\Gamma_A)$, $\bigcup\limits_{\Psi\in\Delta}T(\Psi_A)$ як $T(\Delta_A)$, $\bigcup\limits_{\Phi\in\Gamma}F(\Phi_A)$ як $F(\Gamma_A)$, $\bigcap\limits_{\Psi\in\Delta}F(\Psi_A)$ як $F(\Delta_A)$.

Відношення $_A|\!=_{Cl}$, $_A|\!=_{Cm}$, $_A|\!=_{T}$, $_A|\!=_{F}$, $_A|\!=_{TF}$ наслідку для пари множин формул в моделі мови $\boldsymbol{A}$ задаємо так:

$\Gamma_A|\!=_{Cl}\Delta$, якщо $T(\Gamma_A)\cap F(\Delta_A)=\varnothing$;

$\Gamma_A|\!=_{Cm}\Delta$, якщо $F(\Gamma_A)\cup T(\Delta_A)={}^{V}A$;

$\Gamma_A|\!=_{T}\Delta$, якщо $T(\Gamma_A)\subseteq T(\Delta_A)$;

$\Gamma_A|\!=_{F}\Delta$, якщо $F(\Delta_A)\subseteq F(\Gamma_A)$;

$\Gamma_A|\!=_{TF}\Delta$, якщо $\Gamma_A|\!=_{T}\Delta$ і $\Gamma_A|\!=_{F}\Delta$.

Відповідні відношення $|\!=_{Cl}$, $|\!=_{Cm}$, $|\!=_{T}$, $|\!=_{F}$, $|\!=_{TF}$ логічного наслідку для пари множин формул вводимо за наступною схемою: $\Gamma|\!=_{*}\Delta \Leftrightarrow \Gamma_A|\!=_{*}\Delta$ для кожної моделі мови $\boldsymbol{A}$.

Теорема 4. Нехай $\boldsymbol{A}=(A, I_A)$ та $\boldsymbol{B}=(A, I_B)$ дуальні. Тоді:

1) $\Gamma_A|\!=_{T}\Delta \Leftrightarrow \Gamma_B|\!=_{F}\Delta$ та $\Gamma_A|\!=_{F}\Delta \Leftrightarrow \Gamma_B|\!=_{T}\Delta$;

2) $\Gamma_A|\!=_{Cl}\Delta \Leftrightarrow \Gamma_B|\!=_{Cm}\Delta$ та $\Gamma_A|\!=_{Cm}\Delta \Leftrightarrow \Gamma_B|\!=_{Cl}\Delta$.

Наслідок 2. 1) Для загальної семантики маємо:

$\Gamma \models_T \Delta \Leftrightarrow \Gamma \models_F \Delta \Leftrightarrow \Gamma \models_{TF} \Delta$;

2) $\Gamma \models_{Cl} \Delta$ в неокласичній семантиці $\Leftrightarrow \Gamma \models_{Cm} \Delta$ в пересиченій;

3) $\Gamma \models_T \Delta$ в неокласичній семантиці $\Leftrightarrow \Gamma \models_F \Delta$ в пересиченій;

4) $\Gamma \models_F \Delta$ в неокласичній семантиці $\Leftrightarrow \Gamma \models_T \Delta$ в пересиченій.

Теорема 5 (заміни еквівалентних). Нехай $\Phi \sim_{TF} \Psi$, тоді $\Phi, \Gamma \models_* \Delta \Leftrightarrow \Psi, \Gamma \models_* \Delta$ та $\Gamma \models_* \Delta, \Phi \Leftrightarrow \Gamma \models_* \Delta, \Psi$ (∗ –одне з *Cl*, *Cm*, *T*, *F*, *TF* у відповідних семантиках).

Властивості відношень логічного наслідку для множин формул в різних семантиках для випадку ЧКНЛ наведено в [2], для випадків ЧКНЛР, ε-ЧКНЛ, ε-ЧКНЛРР – в [3–5, 8]. Властивості, пов'язані з реномінаціями, кванторами та предикатами-індикаторами, індукуються відповідними властивостями для формул.

Спектр секвенційних числень ЧКНЛ

Властивості відношень логічного наслідку для множин формул є семантичною основою побудови для розглянутих класів ЧКНЛ низки числень секвенційного типу. Ми пропонуємо спектр секвенційних числень для різних відношень логічного наслідку в різних семантиках для ЧКНЛ та їх розширень (табл. 1). Тут E означає "еквітонних предикатів", A означає "антитонних предикатів".

Таблиця 1. Спектр секвенційних числень ЧКНЛ

Семантика	$\models_{Cl}$	$\models_{Cm}$	$\models_T$	$\models_F$	$\models_{TF}$
Неокласична ЧКНЛ	QC	–	QL	QR	QLR
Неокласична ЧКНЛ E	QMC	–	QML	QMR	$QMLR$
Неокласична ЧКНЛРР	$Q_\perp C$	–	$Q_\perp L$	$Q_\perp R$	$Q_\perp LR$
Неокласична ЧКНЛРР E	$Q_\perp MC$	–	$Q_\perp ML$	$Q_\perp MR$	$Q_\perp MLR$
Неокласична ε-ЧКНЛ	$Q\varepsilon C$	–	$Q\varepsilon L$	$Q\varepsilon R$	$Q\varepsilon LR$
Неокласична ε-ЧКНЛРР	$Q_\perp\varepsilon C$	–	$Q_\perp\varepsilon L$	$Q_\perp\varepsilon R$	$Q_\perp\varepsilon LR$
Пересичена ЧКНЛ	–	QC	QR	QL	QLR
Пересичена ЧКНЛ A		QMC	QMR	QML	$QMLR$
Пересичена ЧКНЛРР	–	$Q_\perp C$	$Q_\perp R$	$Q_\perp L$	$Q_\perp LR$
Пересичена ЧКНЛРР A		$Q_\perp MC$	$Q_\perp MR$	$Q_\perp ML$	$Q_\perp MLR$
Пересичена ε-ЧКНЛ	–	$Q\varepsilon C$	$Q\varepsilon R$	$Q\varepsilon L$	$Q\varepsilon LR$
Пересичена ε-ЧКНЛРР	–	$Q_\perp\varepsilon C$	$Q_\perp\varepsilon R$	$Q_\perp\varepsilon L$	$Q_\perp\varepsilon LR$
Загальна ЧКНЛ	–	–	QG	QG	QG
Загальна ЧКНЛРР	–	–	$Q_\perp G$	$Q_\perp G$	$Q_\perp G$
Загальна ε-ЧКНЛ	–	–	$Q\varepsilon G$	$Q\varepsilon G$	$Q\varepsilon G$
Загальна ε-ЧКНЛРР	–	–	$Q_\perp\varepsilon G$	$Q_\perp\varepsilon G$	$Q_\perp\varepsilon G$

Для пропонованих секвенційних чисел справджуються теореми коректності та повноти.

Секвенційні числення ЧКНЛ, зокрема, ЧКНЛ монотонних предикатів та ЧКНЛ антитонних предикатів, описано в [7]. Секвенційні числення ε-ЧКНЛ та ЧКНЛРР розглянуто в [4] та [8].

Опис секвенційних числень ЧКНЛРР монотонних предикатів і антитонних предикатів та числень ε-ЧКНЛРР, $ЧКНЛ_U$, $ЧКНЛРР_U$, $ε\text{-}ЧКНЛ_U$, $ε\text{-}ЧКНЛРР_U$ буде зроблено в наступних роботах.

Висновки

Досліджено чисті першопорядкові логіки часткових однозначних, тотальних неоднозначних та часткових неоднозначних квазіарних предикатів. Розглянуто розширення цих логік за допомогою узагальнених реномінацій і предикатів-індикаторів наявності значення для змінних. Описано мови та семантичні моделі пропонованих логік, розглянуто різні формалізації відношення логічного наслідку для пар та для множин формул. Для різних класів досліджених логік запропоновано спектр числень секвенційного типу.

Список використаних джерел

1. Нікітченко М.С. Математична логіка та теорія алгоритмів / М.С. Нікітченко, С.С. Шкільняк. – Київ: ВПЦ Київський університет, 2008. – 528 с.
2. Нікітченко М.С. Прикладна логіка / М.С. Нікітченко, С.С. Шкільняк. – Київ: ВПЦ Київський університет, 2013. – 278 с.
3. Нікітченко М.С. Логіки часткових предикатів з розширеними реномінаціями та кванторами / М.С. Нікітченко, О.С. Шкільняк, С.С. Шкільняк // Вісник Київського національного університету імені Тараса Шевченка. Сер.: фіз.-мат. науки. – 2013. – Вип. 2. – С. 210–215.
4. Нікітченко М.С. Композиційно-номінативні логіки із спеціальними предикатами наявності значення для змінних / М.С. Нікітченко, С.С. Шкільняк // Вісник Київського національного університету імені Тараса Шевченка. Сер.: фіз.-мат. науки. – 2013. – Спецвипуск. – С. 128–133.
5. Нікітченко М.С. Першопорядкові композиційно-номінативні логіки із узагальненими реномінаціями / М.С. Нікітченко, О.С. Шкільняк, С.С. Шкільняк // Проблеми програмування. – 2014. – № 2–3 – С. 17–28.
6. Шкільняк С.С. Композиційно-номінативні логіки часткових та неоднозначних предикатів / С.С. Шкільняк // Компьютерная математика. – 2014. – Вып. 1. – С. 93–102.

7. Шкільняк С.С. Спектр секвенційних числень першопорядкових композиційно-номінативних логік / С.С. Шкільняк // Проблеми програмування. – 2013. – № 3 – С. 22–37.
8. Шкільняк О.С. Секвенційні числення логік часткових предикатів з розширеними реномінаціями / О.С. Шкільняк // Вісник Київського національного університету імені Тараса Шевченка. Сер.: фіз.-мат. науки. – 2013. – Спецвипуск. – С. 199–204.

ПРО ДЕЯКІ ЗАДАЧІ ІНФОРМАЦІЙНОГО КЕРУВАННЯ ТА МАНІПУЛЮВАННЯ ПРОЦЕСОМ ПРИЙНЯТТЯ РІШЕНЬ В РАМКАХ БАГАТОРІВНЕВОГО АГЕНТНО-ОРІЄНТОВАНОГО МОДЕЛЮВАННЯ

О.В.Олецький
Національний Університет "Києво-Могилянська Академія"
oletsky@ukr.net

Вступ

Ряд наукових та прикладних задач зручно характеризувати в термінах послідовного прийняття рішень, спрямованих на досягнення певної мети. Прийняття рішень передбачає аналіз поточної ситуації на основі наявної інформації та вибір однієї з можливих дій, що призводить до зміни ситуації.

Основна частина

Як основу для побудови моделей і методів послідовного прийняття рішень часто розглядається граф станів [1, 2, 3 та ін.]: вузли такого графа відповідають можливим ситуаціям, а дуги-діям.

Більш формально, нехай S – деяка множина основних станів. Крім основних станів, введемо деяку множину неявних станів (гіперстанів) Q. На відміну від основних станів, гіперстани – це приховані стани. Переходи між ними явно не постулюються, але агент в процесі розв'язання задачі може потрапляти в той чи інший гіперстан.

Нехай $A(S_k)$ - множина дій, можливих у стані S_k. В результаті дії $A_{jk} \in A(S_k)$ відбудеться перехід до деякого стану S_l з імовірністю $p(S_k, A_{jk}, S_l)$. Таким чином, стан S_l безпосередньо досяжний зі стану S_k, якщо існує дія A_{jk} така, що $p(S_k, A_{jk}, S_l) > 0$.

Переходи між станами можуть характеризуватися вагами. Позначимо через c_{ik} вагу переходу між станом S_i та станом S_k. Змістовно ці ваги можуть мати різні значення: вартість переходу, довжина з'єднуючої дуги тощо.

Деякі задачі в рамках цієї базової моделі є досить простими, і їх можна розв'язувати на основі суто математичних методів. Але

для більш складних задач часто буває доцільним застосовувати методики імітаційного моделювання, зокрема підхід на основі побудови інтелектуальних агентів. В рамках цього будується загальна модель середовища, в якому діють окремі агенти. На основі певних правил і моделей поведінки агент має приймати рішення в кожній ситуації.

Різні постановки задачі послідовного прийняття рішень призводять до різних моделей та формалізацій, і не всі вони є достатньо дослідженими. Зокрема, можна розрізняти такі ситуації:

1. Стратегії прийняття рішень є фіксованими і наперед заданими. У такому випадку власне задача прийняття рішень є фактично вирішеною, але може виникати задача аналізу повторюваності або важливості окремих ситуацій, тобто фактично розподілу певного ресурсу.

2. Стратегії прийняття рішень формуються на основі тих чи інших породжуючих моделей. На найбільш загальному рівні ці моделі можна класифікувати наступним чином:

- чисто випадковий вибір;
- врахування евристичних правил прийняття рішень;
- глибинний аналіз наявних знань про предметну область.

3. Стратегії прийняття рішень формуються в результаті навчання. Типовим є навчання на основі проб та помилок, тобто навчання з підкріпленням. Відомі різні підходи до навчання з підкріпленням, на сучасному етапі як один з магістральних напрямків прийнято розглядати марковські процеси прийняття рішень [1].

Але агенти взаємодіють з оточуючим середовищем, і, можливо, між собою; важливо дослідити можливі моделі такої взаємодії і те, як вони впливають на процес прийняття рішень. Зокрема, агенти мають різні цілі, тобто зразу ж виникає питання про доцільність залучення тих чи інших ігрових підходів. Середовище може також мати власні цілі, які не обов'язково співпадають з цілями агента. Крім того, в ряді задач можна виділити проміжні рівні керування, пов'язані з окремими вузлами графа станів.

Важливим є і те, що агенти приймають рішення на основі певної інформації. Ця інформація може накопичуватися в процесі навчання, але вона може надаватися середовищем. Таким чином, виникає задача інформаційного керування з боку центрального вузла керування, а також окремих керуючих рівнів.

Ми розглядаємо задачі, в яких керуючі впливи носять опосередкований характер, при якому вузли керування не можуть безпосередньо впливати на прийняття рішень. Вони лише надають

агентам певну інформацію, і агенти мають самі вирішувати, як використовувати цю інформацію. Таке керування можна охарактеризувати як інформаційне керування.

Якщо мета вузла керування співпадає з метою агента, вузол керування має сприяти агентові та надавати інформацію, яка допоможе йому приймати оптимальні рішення. Тоді можна говорити про підтримку прийняття рішень. Але, якщо цілі різні, вузол керування буде схильний надавати таку інформацію, яка б спонукала агента прийняти рішення, вигідне для вузла керування, а не для самого агента. Тоді можна казати про маніпулювання прийняттям рішень. Приклади такого маніпулювання можна навести навіть для найпростіших і найбільш досліджених задач.

Але вузли керування теж можуть не мати повної інформації про корисність того чи іншого рішення, і вони теж мають навчатися. Тому мова має йти про паралельне навчання різних рівнів керування та про побудову відповідних моделей і методик.

За останні роки з'явилося багато нових прикладних задач, які можна моделювати та досліджувати на основі описаної методики. Це задачі, пов'язані з інформаційним пошуком в середовищі WWW або на окремому тематичному порталі. Зокрема, в [5, 6] розглядалися наступні напрямки:

1. Оптимізація структури веб-порталу та його навігаційного графа.
2. Автоматизований підбір контекстної реклами
3. Використання в навчальному процесі.

Базовий підхід до організації такого експерименту може полягати в побудові породжуючих моделей, на основі яких можна отримати імовірності (точніше кажучи – коефіцієнтів упевненості) переходів між станами. Таким чином, можна сформувати матрицю перехідних імовірностей $P=(p_{ij}),\ i,j=1,\ldots,n$, де p_{ij} – імовірність того, що агент, який перебуває в стані s_i, перейде до стану s_j. За певних умов такі переходи утворюють марковський процес, аналіз якого дозволяє отримати стаціонарний вектор розподілу ймовірностей $p=(p_1,\ldots,p_n)$, де p_j – ймовірність того, що в деякий момент часу агент буде перебувати в стані s_j. Ці ймовірності є стаціонарними, тобто не залежать від вибору моменту часу, і вектор p можна отримати як головний лівий власний вектор матриці P, тобто, як розв'язок рівняння

$$p = p\,P$$

Далі розподіл мір важливостей між станами певним чином пов'язується з відповідними стаціонарними ймовірностями.

Подібний аналіз лежить в основі відомого алгоритму PageRank [4], Але класичний PageRank орієнтований на деяку

фіксовану породжуючу модель для задачі інформаційного пошуку, і можна розглядати різні породжуючи моделі для різних задач.

Найпростішим методом отримання стаціонарних імовірностей є метод простої ітерації, або степеневий метод. Робота методу починається з вибору початкового наближення x_0; кожне наступне наближення отримується з попереднього за формулою

$$p_{i+1} = p_i P$$

Послідовність наближень p_i збігається до вектора p^*, який і є головним лівим власним вектором.

Але цю базову ітеративну схему легко адаптувати до застосування різних типів породжуючих моделей. Зокрема, можна ввести деяку функцію *WP(·)* – процедура оцінки міри важливості станів за заданими стаціонарними ймовірностями.

Якщо вже знайдені міри важливостей станів, стани можна упорядкувати за цими мірами. При цьому імовірність переходу до більш важливого стану, очевидно, збільшується.

Можна розглядати деяку функцію залежності ймовірностей переходу від мір важливості *PW(·),* яка вочевидь має бути монотонно неспадною, якщо аргументом є власне міра важливості і монотонно не зростаюча, якщо аргументом є порядковий номер.

Тоді модифікований ітераційний процес можна записати у вигляді наступної схеми:

```
Дано: матриця перехідних імовірностей P.
do {
- отримати вектор стаціонарних імовірностей
p як головний лівий власний вектор матриці P,
тобто як розв'язок системи

pP = p

- оновити міри важливості на основі функції
WP(·)  :

w:=WP(p)

– оновити перехідні імовірності на основі
функції PW(·)
P:–PW(w);
}
while (не досягнуто критерію завершення)
```

Висновки

В описаній схемі ключову роль мають відігравати породжуючі моделі обчислення перехідних імовірностей на основі

інформації, отриманої на основі навчання (зокрема, на основі марковських моделей прийняття рішень), наявних знань про предметну область, а також на основі повідомлень, отриманих від центрів керування різних рівнів. Відповідні формалізації – як загального характеру, так і в застосуванні до окремих конкретних задач, мають стати предметом подальшого дослідження.

Список використаних джерел

1. Рассел С. Искусственный интеллект: современный подход./ С. Рассел, П. Норвиг – М.: Изд. дом «Вильямс», 2006. – 1408 с.
2. Люгер Дж. Искусственный интеллект: стратегии и методы решения сложных проблем. /Дж. Люгер – М.: Изд. дом «Вильямс», 2003. – 864 с.
3. Глибовець М.М. Штучний інтелект. / М.М. Глибовець, О.В. Олецький – К.: Вид.дім "Академія", 2002. – 366 с.
4. Маннинг К.Д. Введение в информационный поиск. / К.Д. Маннинг, П. Рагхаван, Х. Шютце – М.: ООО «И.Д. Вильямс», 2011. – 528 с.
5. Олецький О.В. Про оптимізацію структури веб-порталу на основі марковських процесів прийняття рішень. / О.В. Олецький // Вісник Київського національного університету імені Тараса Шевченка**.** Серія фізико-математичні науки. 2013 р., спецвипуск. – С.134-137.
6. Олецький О.В. Про застосування марковських процесів прийняття рішень для автоматизованого добору навчальних матеріалів у системах blended learning. / О.В. Олецький // Наукові записки НаУКМА. Комп'ютерні науки. – К., 2013. – С.115-118.

A METHOD FOR PARALLEL SOFTWARE CORRECTNESS PROOF IN IPCL

T. V. Panchenko
National Taras Shevchenko University of Kyiv, Ukraine
pantaras@ukr.net

Introduction

The methodology for software correctness proof was developed in [1] and presented there in details. Using this methodology one can formulate and prove properties of programs developed in Interleaving Parallel Composition Language (IPCL) [1]. The syntax and operational semantics of IPCL presented ibid. Here we concentrate on simplified method for program properties (including correctness) proof for some subclass of all IPCL programs. The notion of simplified state for this kind of reasoning (introduced in [1]) as well as clarification on specific IPCL subclass mentioned above are also discussed here.

IPCL Subclass and Method Description

Following [2], the composition languages class [3] IPCL and methodology proposed by the author for programs properties (including correctness) proof in composition languages IPCL [2] are well suited for the specification and verification of the server-side software in client-server environments.

But it should be noted that in the absence of local data in programs A, B, …, C (which are subroutines of program P), it is advisable to consider the concept of a **Simplified State** (here we use the notation and follow the terms according to [2], in particular, the program $P = A^n \parallel B^m \parallel \ldots \parallel C^k \in SeqILProgs$, where *SeqILProgs* is the sequential programs subclass of all IPCL programs). Simplified state is an aggregated state of the following form $N^{Pmq} \times D$, where $Pmq = \|A_{marks}\| + \|B_{marks}\| + \ldots + \|C_{marks}\|$, i.e. the sum of all tags amounts (A_{marks}, B_{marks}, …, C_{marks}) for programs A, B, …, C (which are subroutines of P), where A_{marks} is the set of labels for labeled program A – i.e. when we labeled every particular atomic operator (or operation, function call etc.) in the text notation of program A, $\|A\| = card(\mathrm{A})$ is the power of the set A, and D contains the global (shared, common) data for all routines in the form of *nominative data* [3], which means a set of pairs $name \mapsto value$.

First $\|A_{marks}\|$ components of such a state include the number of programs that are executing now operators at appropriate labels of program A in this state, and the sum of these components in each state for the program P is equal to n – i.e. the number of instances (copies) of A program in P (in terms of program P). The following $\|\mathrm{B}_{\mathrm{marks}}\|$

components contain the number of programs that are executing now operators at appropriate labels of *B* program in this state, and the sum of these components in each state for the program *P* is equal to *m* (the number of instances of the program *B* in *P*) in terms of *P* programs and so on. The last component *D* contains global shared data for all routines of *P*.

Simplified state operates with the number of routines which are executing now at appropriate label instead of label identification for each instance of every individual routine (*A*, *B*, ..., *C*). In fact, routines are not distinguished from each other accurate to the label of current execution if they do not change local data (actually – do not have them at all), but only operate with shared global data.

One can obtain a simplified state for the state $S \in A_{marks}{}^{n} \times B_{marks}{}^{m} \times \ldots \times C_{marks}{}^{k} \times D \times D^{numprocs}$, $numprocs=n+m+\ldots+k$, in the following way. As sets $A_{marks}, B_{marks}, \ldots, C_{marks}$ are finite, let $A_{marks}=\{A_1, \ldots, A_a\}$, …, $C_{marks}=\{C_1, \ldots, C_c\}$. Let $Pr_i(S)$ be the i-th component of a tuple *S*. Then for any regular state *S* the corresponding simplified state will be $SS=(a_1, \ldots, a_a, \ldots, c_1, \ldots, c_c, d)$, where $a_j=\|\{ i \mid Pr_i(S)=A_j, i \in N_{Pmq} \}\|= P_{S[A_j]}$, $\forall j \in N_a$, …, $c_j=\|\{ i \mid Pr_i(S)=C_j, i \in N_{Pmq} \}\|= P_{S[C_j]}$, $\forall j \in N_c$, $d=Pr_{numprocs+1}(S)$.

The set of simplified states *SStates* introduced accordingly. The same goes with simplified initial states *SStartStates* (they all have the structure (n, 0, ..., 0, ..., k, 0, ..., 0, d), which means all the routines are at their appropriate start labels), simplified final states *SStopStates* (their structure is (0 ..., 0, n, ..., 0, ..., 0, k, d), which means all the routines have stopped execution at their exit ("after-program") label) and the transition function (one step execution of *P* program) over the simplified states *SStep*: $SStates \rightarrow SStates$. For every particular possible transition of execution control for each routine this new transition function *SStep* decreases the value of some components of the *SStates* vector by 1 (eg. first component) and at the same time increases the value of some other component by 1 (eg. second component), which means the transfer of execution control from one label (in this case A_1) to another label (in this case A_2) for one of the routines (in this case *A*), and also changes the value of the last components (global data) if the current function is in subclass *Oper* of class *F*. All these objects could be easily obtained by transferring states into their simplified states appropriately or by direct construction.

Sample Correctness Task: Parallel Increment To Shared Variable. Discussion

To prove the properties of both types (defined in [1,2]) one could apply the methodology given in [1,2]. To demonstrate the simplicity of the method [1,2] and the convenience of simplified state model usage, let us discuss a sample that – de facto – (see. [4], [5] and others) become "standard" to check the "efficiency" of parallel programs and methods concerning modeling, execution, model checking and correctness proof. Let the program twice (separately, independently) increments some shared (global) variable concurrently. In the IPCL notation program P will look like $x:=x+1 \parallel x:=x+1$, where x is a shared (global) variable and the set $F=\{f_1\}$, where $f_1\ (d) \equiv d \nabla [\ x \mapsto (x \Rightarrow (d)+1)\ ]$ (here the semantic function f_1 has syntactic form of $x:=x+1$). It is clear that every action is atomic (read, add +1, write back the variable value). The problem is formulated as follows: to prove that when the initial value of shared (global) variable x is 0, the final value of x (after the program stops) will be equal to 2.

Without going into details, "pure" Owicki-Gries method [4] requires program properties to be formulated and tested for each operator taking into account their potential interference and "cross"-interference effects. Recall that inspections quantity is quadratic with respect to the program operator count.

The extended version of the rely-guarantee Owicki-Gries method modification [5] requires two additional variables (for such a trivial program!) introduction and formulating of nontrivial (!) rely- and guarantee- conditions for application of the method to this task. The proof itself takes more than one page space [5].

TLA [6] offers (for the very similar task) to build a model that is not much easier than the first two, and the formulation of the model and, in fact, the proof itself (with explanations) is again at least the page of formulas in space.

Sample Correctness Task: Parallel Increment To Shared Variable. Proof

Let us now consider a *detailed* proof of *generalization* of this property in IPCL. Explore generalized version of the task first. Thus, let us have a program $P = Inc^n$ (for some fixed $n \in N$), where $Inc = x:=x+1$. The semantic model described above with $F=\{f_1\}$. We consider a simplified model, as we have no local variables for routines (just shared global x). Labelling algorithm [1] (obviously) will result to the next for Inc: $Inc =$ **[M1]** $x:=x+1$ **[M2]**, $SStates=\{(s_1, s_2, d) \mid s_1 \in N,\ s_2 \in N,\ d \in D\}$, where s_1 is the number of programs (all performed in parallel) executing currently at the label **[M1]**, s_2 is the number of programs executing at

the label **[M2]** (in fact, finished currently), and d is the shared (global) data (containing variable x and its value). It is clear that $SStep=\{((s_1, s_2, d), (s_1-1, s_2+1, f_1(d))) \mid s_1>0 \ \& \ s_1 \in N \ \& \ s_2 \in N\}$, $SStartStates=\{(n, 0, [\, x \mapsto 0\,])\}$, $SStopStates=\{ s \mid s=(0, n, d') \ \& \ \exists s_1, s_2, \ldots, s_l \bullet (s_1 \in SStartStates \ \& \ s_l=s \ \& \ (\forall i \in N_{l-1} \bullet (s_i, s_{i+1}) \in SStep)\ )\ \}$ – by definition.

Let $PreCond(SS) \equiv (x \Rightarrow (d)=0)$ and $PostCond(SS) \equiv (x \Rightarrow (d)=n)$, where $SS=(s_1, s_2, d) \in SStates$ – (simplified) state. Consider the *invariant* $Inv(SS) \equiv (s_2=x \Rightarrow (d))$. Let us verify $InvCond(Inv, PreCond, PostCond)$: $\forall S \in SStartStates \bullet (\ PreCond(S) \rightarrow Inv(S)\ ) = PreCond((n, 0, [\, x \mapsto 0\,])) \rightarrow Inv((n, 0, [\, x \mapsto 0\,])) = \text{True} \rightarrow \text{True} = \text{True}$, $\forall S \in SStopStates \bullet (\ Inv(S) \rightarrow PostCond(S)\ ) = \forall S=(s_1, s_2, d)=(0, n, d') \in SStopStates \bullet (\ (s_2=x \Rightarrow (d)) \rightarrow (x \Rightarrow (d)=n)\ )$, since $s_2=n$, then the implication is True, and the entire predicate is True, $\forall (S, S') \in SStep \bullet (\ Inv(S) \rightarrow Inv(S')\ ) = \forall (S, S')=((s_1, s_2, d), (s_1-1, s_2+1, f_1(d))) \in SStep \bullet (\ (s_2=x \Rightarrow (d)) \rightarrow (s_2+1=x \Rightarrow (f_1(d)))\ ) = \forall (S, S') \in SStep \bullet$ True, since it is obvious that $x \Rightarrow (f_1(d)) = x \Rightarrow (d)+1$.

Hence, *Inv* indeed is invariant for the program P, besides at the starting states it is a logical consequence of the pre-condition and it implies post-condition at the final states. Q.E.D.

Sample Correctness Task: Parallel Increment To Shared Variable. Conclusions

Thus, we have proved a *generalized* property. Originally formulated (correctness) property can be derived from a more general just letting n=2.

Note that some of sources mentioned here are considered a variant of the example $P' = x:=x+1 \parallel x:=x+2$ with pre-condition x=0. To solve this problem with the method proposed one should use another (but obvious) invariant $Inv(SS) \equiv (s_2+2*s_4 = x \Rightarrow (d))$, where $SS=(s_1, s_2, s_3, s_4, d) \in SStates$ is simplified state, for generalized program $P'' = (x:=x+1)^n \parallel (x:=x+2)^m$ apart with labels (labelled program P): **[M1]** $x:=x+1$ **[M2]**, **[M3]** $x:=x+2$ **[M4]**, and initial states $SStartStates=\{(n, 0, m, 0, [\, x \mapsto 0\,])\}$ (with $n=m=1$).

Note also that the number of inspections to prove is linear regarding the number of program operators in both examples (in general, for every program). Details on this and other examples as well as detailed theory can be found in [1].

Conclusions

The simplified method for program properties (including correctness) proof is appropriate for parallel program without local variables which use only shared ones. This case is appropriate model of

server software (with SMP architecture) and such a kind of computing. The method and the model as well as Interleaving Parallel Compositional Languages class (IPCL) are described in details in [1] and were first presented in [2] and then in [7].

References

1. Panchenko T.V. Compositional Methods for Software Systems Specification and Verification. PhD Thesis. – Kyiv, 2006. – 177 p.
2. Panchenko T.V. The Methodology for Program Properties Proof in Compositional Languages IPCL // Proceedings of the International Conference "Theoretical and Applied Aspects of Program Systems Develpment" (TAAPSD'2004). – Kyiv, 2004. – P. 62–67.
3. Nikitchenko N. A Composition Nominative Approach to Program Semantics. – Technical Report IT-TR: 1998-020. – Technical University of Denmark. – 1998. – 103 p.
4. Owicki S., Gries D. An Axiomatic Proof Technique for Parallel Programs // Acta Informatica. – 1976. – Vol. 6, № 4. – P. 319–340.
5. Xu Q., de Roever W.-P., He J. The Rely-Guarantee Method for Verifying Shared Variable Concurrent Programs // Formal Aspects of Computing. – 1997. – Vol. 9, № 2. – P. 149–174/
6. Lamport L. Verification and Specification of Concurrent Programs // deBakker J., deRoever W., Rozenberg G. (eds.) A Decade of Concurrency, Vol. 803. – Berlin: Springer-Verlag, 1993. – P. 347–374.
7. Panchenko T.V. The Simplified State Model for Properties Proof Method in IPCL Languages and its use and advantages // Proceedings of the International Conference "Theoretical and Applied Aspects of Program Systems Develpment" (TAAPSD'2007). – Berdyansk, 2007. – P. 319–322.

CYBERSECURITY IN UKRAINE: PROBLEM AND PERSPECTIVE

O. V. Potii[1], I.D. Gorbenko[1], O.V. Korneyko[2], Y.I. Gorbenko[3]
[1]V. N. Karazin Kharkiv National University, Ukraine
[2]State Service for Special Communication and Information Protection of Ukraine, Kiev, Ukraine
[3]JSC Institute of Information Technology, Kharkov, Ukraine
potav@ua.fm, gorbenkoi@iit.kharkov.ua, GorbenkoU@iit.kharkov.ua

Introduction

One of the key issues in the context of globalization of information exchange and the widespread introduction of information technology is the issue of protection of information processed in the information and telecommunication systems, challenges and threats in cyberspace.

The possibilities of cyberspace, the development and implementation of new information and communication technologies provide great circumtances for the accumulation and use of information and create a fundamental dependence on their functioning in all spheres of society and the state: the economy, politics, spheres of national and international security and so on. This dependence is a vulnerable point in the operation of facilities and critical national infrastructure.

On the other hand it enables criminals to realize unlawful actions in cyberspace by destroying the integrity, availability and confidentiality of information and damage information resources and telecommunication systems.

Of particular concern is the possibility of using information technology in cyberspace in the interests of political and military power and confrontation of terrorism and hacker attacks.

In such circumstances, the main task of the state is to take measures that will enable resist unlawful actions in cyberspace, to avoid or reduce the negative consequences of the implementation of cyber threats. Because cyber security in the development of the global information society is a necessary component of the national security of any country. The sources of cyber threats are international criminal groups of hackers, some trained in information technology criminals, foreign government agencies, terrorist and extremist groups, multinational corporations and financial-industrial groups and others.

Challenges and basic cyber threats for Ukrainian cyberspace

The problem of ensuring cybersecurity for Ukraine is very urgent. Particularly strong this problem has worsened in the background of Russian aggression. Thus, Russia is conducting cyberwarfare operations against Ukraine as part of its military incursion into Crimea,

the Navy admiral designated to be the next commander of U.S. Cyber Command told Congress in 12 of march 2014. Vice Adm. **Michael Rogers**, the nominee to head Cybercom and the National Security Agency, also said his biggest challenge if he is confirmed for the posts will be dealing with the threat of cyber attacks and penetrations of U.S. computer networks."We clearly see that there's an ongoing cyber element to the challenges in the Ukraine at the moment," the three-star admiral said. (Read more: http://www.washingtontimes.com/news/2014)

A lot of computers in the Ukrainian government, and at least 10 Ukrainian embassies abroad have been victims of dangerous cyber attacks related to the Russian newspaper claims Financial Times (http://www.ft.com/intl/cms/s/0/2352681e-1e55-11e4-9513-00144feabdc0.html)

From cyberattacks also the embassies at of least nine countries in Eastern Europe suffered. Among them are representative of Germany, China, Poland and Belgium. As a result, attackers gained access to secret diplomatic information.

From cyberattacks not only State agency suffers, but also the private sector in Ukraine. About a third of all cyberattacks conducted in Ukraine, it is necessary for business websites (30%), according to "Kaspersky Lab". Often online media (20%) and online shopping (19%) become victims of DDoS. Attacks on media portals have a convenient way to get more targeted traffic, which can then be converted in to money. Cause of DDoS-attacks on e-commerce site scan become extortion, cover other attacks or unfair competition.

The average duration of attacks in Ukraine in the first half of 2012 amounted 11 hours. Record in the domain zone "ua" has become a cyberattack, which lasted for13 days 4 hours, 23 minutes and was directed to a web site marketing agency.

So, problems in the field of cyber security of Ukraine for anybody not a secret. What challenges in cyberspace is Ukraine faced? What issues are the most important?

Experts of the Institute for Strategic Research under the President of Ukraine in the analytical report "Cybersecurity: global trends and challenges for Ukraine" identified several major problems faced by the government policy regarding the protection of the country cyberspace. [1]

1. In Ukraine a normative document describing threats for Ukraine in a cyberspace, determining the state policy in cybersecurity doesn't exist. In Ukraine a consistent terminological system in domain of cybersecurity doesn't exist either. As a result the function of the special police and service units is not supported by the normative

document. Only recently the process of the problem regulation and the ways of its solution finding out has been started.

2. In Ukraine the National interdepartmental coordination structures are not available. These might coordinate and adjust the activity of different departments during inquiry of cybercrime in cyberspace. Unfortunately in fact this cooperation is more on interpersonal level, but not on system level. This cooperation is exposed. Unfortunately, in Ukraine a complex training on cybersecurity with drawing all special departments is not realized (as, for example, "Cyberstorm" in USA).

3. Staffing agencies have unsatisfactory specialists. It is noticed by many heads of state departments. The reasons for this are: unsatisfactory level of training, the lack of tangible and intangible incentives to remain in the public service. In addition, there are no multi-research institutes that are engaged in a comprehensive study of the issues of information security.

4. Liability distribution on cybersecurity among government departments is not clear.

5. Information and telecommunication network in Ukraine is vulnerable because it is widely used by foreign software products and technical basis of foreign manufacture. Search for possible vulnerabilities and special holes in these programs and product scan not be performed. Ukraine's dependence on foreign products is dangerous.

The draft strategy cybersecurity of Ukraine identified the main threats for Ukraine in cyberspace. [2,3]

Cybercrime. Crimes using modern information and communication technologies are becoming more commonplace in the lives of Ukrainian citizens. The new technologies are used not only for committing traditional crimes, but also to commit new crimes, especially characteristic for advanced information society. Most attention is focused on criminals attempted violation or unauthorized use of the information and telecommunication systems of government, credit and banking, utilities, defense, industrial sectors. Classified information circulating in the information and telecommunication systems, is a stable object of interest from other countries, organizations and individuals.

Cyberterrorism. Domestic enterprises, institutions and organizations, the violation of which constitutes a threat to life and health of citizens, can be a potential target for terrorist acts, including the use of modern information and communication technologies. Not less is a threat of to committing illegal acts to the detriment of third countries carried out using the information infrastructure of Ukraine.

Cyberwar. Military sector is undergoing major changes in the result of the development of cyberspace. Most countries in the world are actively transforming their potential in the defense and strengthening capabilities of warfare in cyberspace and protection against similar actions of the enemy, as it becomes more relevant to the new types of threats. Because of the broad information security and defense sector, defense capabilities of Ukraine become more susceptible to cyber threats. Implementation of modern information technology transforms a separate cyberspace, along with the traditional "Earth", "Air", "Sea", space sphere of warfare. Appropriate level of defense capability means there are units that are able to withstand cyberthreads defense.

The *vulnerability of information infrastructure of the state*. Recently, cyberattacks and cybercrime objects are information resources of financial institutions, transport and utilities, state agencies that provide security, defense, emergencies, their official website and email servers. The sharp increase in the number of recorded cases of cyberattacks on governmental information resource effects strengthening of hacker movement to violate the information systems of state agencies.

In addition, spreads of politically motivated activity of cyberspace groups that carry out attacks on government and private websites. This leads to violations of information resources, as well as the reputation and financial losses.

The unsatisfactory state of information security, which is recorded in the event of state control. This may affect the sustain able functioning of critical information infrastructure, lower the defense of the state, its economic, financial and political instability, weaken the image and attractiveness of investment and so on.

The results of the risk analysis is fixed in the regulations of the government. For example, in [4] presented a classification of the negative impact of cyberattacks on critical information infrastructure object:

- technogenic emergencies and negative impact on the environmental safety of the country;
- negative impact on energy security;
- negative impact on the economic security;
- negative impact on national defense, national security and law and order in the country;
- negative impact on the system of governance;
- negative impact on the socio-political situation in the country;
- negative impact on the image of the state;
- violation was functioning financial system;
- violation sustainable operation of transport infrastructure;

- violation sustainable operation of information and communications infrastructure of the state, including its interaction with the infrastructure of other countries.

References

1. Dubov D.V. Ozgevan M.A. Cybersecurity: the World Tendencies and Challenge for Ukraine. - K.: NISR, 2011. - 30 p.
2. Cybersecurity strategy of Ukraine. [Electronic resource]. - Access mode http://www.niss.gov.ua/public/File/2013_nauk_an_ rozrobku/ kiberstrateg.pdf.
3. Cybersecurity strategy of Ukraine. [Electronic resource]. - Access mode http://cst.org.ua/docs/lipen-OUT/strategiya_kiberbezpeku.pdf.
4. On the Action Plan for the protection of state information resources Cabinet Of Ministers Of Ukraine ORDER on November 5, 2014 p. № 1135-r Kyiv.

СУТНІСТЬ ПРОЕКТУ ЕЛЕКТРОННОЇ ІДЕНТИФІКАЦІЇ ТА АВТЕНТИФІКАЦІЇ STORK 2.0 ТА МОЖЛИВОСТІ ВИКОРИСТАННЯ ЙОГО РЕЗУЛЬТАТІВ В УКРАЇНІ

О.О. Продан, К.В. Ісірова
Харківський Національний Університет імені В.Н. Каразіна
Olincess1@gmail.com

Вступ

Широке застосування електронного цифрового підпису в ЄС суттєво поліпшило виконання електронних операцій на цифровому ринку. Але виявилося, що в ЄС не існує всебічних транскордонних та міжрегіональних основ для безпечних, надійних і простих електронних операцій, безпечних електронних послуг з електронної ідентифікації, автентифікації, підписів, печаток, міток часу, електронних документів, послуг електронної доставки і автентифікації веб - сайту і т.д.

Для рішення зазначених протиріч прийнято в ЄС: Регламент ЄС у 2012 році [1], а 28 лютого 2014 Європейська комісія схвалила правила електронної ідентифікації і довірчих послуг для електронних угод на внутрішньому ринку - «Регламент Європейського Парламенту і Ради з електронної ідентифікації і трастових сервісів для електронних операцій на внутрішньому ринку» [2].

Основні поняття та зміст проекту

Слід зазначити,що у проекті STORK 2.0 та у Регламенті ЕС надані наступні визначення:

Довірча послуга - це будь-яка електронна послуга, включаючи електронну ідентифікацію та електронну автентифікацію, що входить до складу створення, перевірки, підтвердження правильності, обробки та зберігання електронних підписів, електронних печаток, електронних міток часу, електронних документів, послуг електронної доставки, підтвердження автентичності сайту та електронних сертифікатів, включаючи сертифікати електронного підпису та електронних печаток [2].

Електронна ідентифікація - процес використання унікальних даних предмету (в тому числі персональних даних особи) в електронній формі з метою встановлення однозначної ідентичності людини/особи. Ідентифікація забезпечує виконання встановлення автентичності та визначення повноважень суб'єкта при допуску його в систему, контролювання встановлених повноважень в процесі сеансу роботи; реєстрація дій та ін.

Електронна автентифікація - Процедура встановленні належності користувачеві інформації в системі пред'явленого ним ідентифікатора [6]. Для впровадження вищеназваних механізмів та їх реалізації було започатковано низку проектів, одним з яких є проект реалізації електронної ідентифікації STORK (Безпечна транскордонна ідентифікація та автентифікація). На сьогоднішній день діє версія STORK 2.0 - це трирічний проект (реалізація по 2015 рік), метою якого є створення і впровадження єдиної загальноєвропейської системи електронної ідентифікації та автентифікації для фізичних та юридичних осіб. Проект анонсований 9 партнерами з 11 країн, проте згодом число партнерів збільшилося до 58 [3].

Завдання проекту STORK 2.0.

Безпечна транскордонна ідентифікація 2.0 сприятиме реалізації єдиної європейської системи електронної ідентифікації та автентифікації. Проект STORK 2.0 спирається на результати проекту STORK, який дав поштовх до створення системи взаємодії для різних підходів на національному та європейському рівнях, електронної ідентифікації фізичних та юридичних осіб, а також використання мандатів [5].

STORK 2.0 буде кроком вперед на шляху до створення повністю інтероперабельної основи та інфраструктури для електронної ідентифікації та автентифікації в ЄС. Це можливо завдяки:

Реалізація проекту STORK 2.0:

- Дозволить Єдиному Ринку охопити юридичних осіб за допомогою використання атрибутів
- Полегшить транскордонне електронне управління
- Знизить адміністративний тягар для компаній та фізичних осіб, які хочуть надавати транскордонні послуги [5].

Проект STORK 2.0 включає в себе 4 пілотних проекти: електронне навчання та Академічна кваліфікація (eLearning & Academic Qualifications); Електронний банкінг (eBanking); Державні послуги для бізнесу (Public Services for Business); Електронна система охорони здоров'я (eHealth).

Пілотні проекти – це інтенсивне використання майже кінцевої версії проектів, з метою виявлення максимального числа помилок в роботі для вилучення їх до фінального виходу проектів. Вони покликані випробувати та протестувати в реальних умовах перспективні проекти, підтвердити загальні специфікації, стандарти та складові частини, а також сприяти мобільності цифрового життя в ЄС для приватного та комерційного секторів.

Проект «Електронне навчання та Академічна акредитація» буде сприяти розширенню кордонів для навчання незалежно від територіального місця знаходження студента. В результаті буде утворена Європейська зона вищої освіти. Важливою особливістю є те, що він зорієнтований не лише на студентів, а також охоплює інтереси випускників та претендентів на робоче місце. Зокрема передбачені механізми для полегшення працевлаштування за кордонами країни проживання. Академічні атрибути для працевлаштування дозволять використовувати:

- сервіси для перевірки кваліфікації для полегшення перевірки академічної кваліфікації;
- сервіси для вибору місця роботи для полегшення відбору працівників компаніями, які наймають іноземних громадян з відповідною кваліфікацією;
- Транскордонні простори для електронного навчання зроблять можливим:
- електронне навчання завдяки загальному простору для електронного навчання студентів;
- електронне опитування використовується для опитування конкретних груп осіб (студенти, викладачі) задля поліпшення умов навчання та роботи в даній конкретній галузі [4].

Метою проекту «Електронний банкінг» є забезпечення взаємодії банківської та користувацької сфер без додаткових правових та бізнес меж. Фізичні та юридичні особи можуть транскордонно користуватися послугами іноземних банків за допомогою їхній web-сервісів.

Послуги відкриття банківського рахунку доповняться наступними перевагами:

- online-запити з використанням засобів транскордонної електронної ідентифікації;
- більш надійне зберігання інформації;
- впевненість у проведених перевірках;
- негайне підтвердження транзакцій.
- Реєстрація у системі e-banking забезпечить:
- можливість використання національних ідентифікаторів для транскордонної системи;
- використання єдиного PIN-коду для входу до систем різних послуг.
- Електронні рахунки дозволять:
- авторизуватися за допомогою єдиного групового ідентифікатора (eID);

- використовувати стандартизовані та нормалізовані рішення, що підвищить рівень довіри серед користувачів відповідних послуг [4].

У сфері державних послуг для бізнесу планується реалізація двох напрямків: реєстрація у державних реєстрах; зручні портали бізнес-сервісів. Також в рамках проекту у юридичних осіб з'явиться можливість використовувати online державні послуги в іншій Державі - члені [4].

Визначено завдання електронної системи охорони здоров'я:

- дозволити пацієнтам отримати доступ до їх медичних даних за кордоном використовуючи їх національні механізми електронної ідентифікації;
- дати змогу іноземним медичним працівникам аварійно-рятувальних служб отримати резюме пацієнта із його рідної країни;
- забезпечити надійні засоби електронної ідентифікації та автентифікації для доступу до інфраструктури epSOS [4].

Результати проекту STORK 2.0. дозволили:

- використовувати досвід від екс-пілотних проектів STORK для впровадження чотирьох транскордонних проектів, а також продовження розвитку попередній напрямків;
- сприяти досягненню цілей Цифрового порядку Денного та підтримувати інновації в Європі за допомогою інформаційно-комуникаційних технологій;
- активізацію Єдиного Ринку для державних та комерційних послуг, сприяння мобільності всередині ЄС;
- розширення можливостей підприємців та громадян за допомогою безкоштовних транскордонних та ефективних моделей електронного управління;
- забезпечення розробки загальноєвропейської інфраструктури для електронної ідентифікації громадян та бізнес сфери.

Висновки. Аналіз можливості участі України в проекті

На сьогоднішній день можна говорити про добре розвиненій інфраструктурі електронного цифрового підпису в Україні: на базі PKI (publickeyinterface) розгорнута широкомасштабна організаційно-технічна структура, призначена для надання послуг електронного цифрового підпису (ЕЦП), що об'єднує центри сертифікації ключів, контролюючий орган, користувачів в єдину систему. Враховуючи переважне використання саме PKI в ЄС, автор вважає, що в нашій країні створено всі технічні умови для впровадження системи єдиної ідентифікації фізичних та

юридичних осіб за допомогою посиленого сертифіката відкритого ключа.

З точки зору правового регулювання, на сьогоднішній день активно розробляється нормативно-правова база, що сприяє розвитку електронних довірчих послуг в Україні згідно з проектом Регламенту Європейського Парламенту і Ради з електронної ідентифікації і довірчих (трастових) сервісах для електронних операцій на внутрішньому ринку. Поточний проект Закону України «Про електронний цифровий підпис» встановлює таку вимогу: «ЕЦП за статусом прирівнюється до власноручного підпису у разі, якщо ЕЦП наложена безпосередньо підписантом - фізичною особою або уповноваженим представником юридичної особи» та «Електронний цифровий підпис за правовим статусом прирівнюється до штампу юридичної особи або фізичної особи - підприємця». Саме це необхідна умова правового статусу ЕЦП забезпечує коректну правову взаємодію бізнес-партнерів (юридичних осіб) з України в рамках ЄС через їх представництво фізичними особами. Використання власноручного підпису або печатки на паперовому документі передбачає їх проставлення безпосередньо підписантом або особою, уповноваженою на використання печатки в діловодстві. Тому, для забезпечення визнання ЕЦП аналогом власноручного підпису або печатки за всіма нормами, в тому числі міждународними, передбачено безпосереднє володіння особистим ключем підписувача та зберігання його в таємниці. Виходячи з вищевикладеного, логічним здається висновок, що в момент прийняття оновленої нормативно-правової бази України стане гідним претендентів на участь у проекті. Подальша євро-інтеграція обіцяє Україні безболісне спадкування принципів і механізмів, прийнятих в ЄС, щодо електронної ідентифікації.

Список використаних джерел.

1. Пропозиція Регламенту Європейського Парламенту та Ради щодо питань ідентифікації та довірчих (трастових) послуг для електронних операцій на внутрішньому ринку (2012).
2. Регламент Європейського Парламенту та Ради щодо питань ідентифікації та довірчих (трастових) послуг для електронних операцій на внутрішньому ринку (2014).
3. STORK 2.0. D1.3.1 Publishable Summary Report.
4. STORK 2.0. D8.4.1 MS Pilot marketing plans.
5. STORK 2.0. D2. 1 Extending eID authentication across Europe.

6. Правила забезпечення захисту інформації в інформаційних, телекомунікаційних та ІТК. Затверджено постановою Кабінету Міністрів України від 29 березня 2006р. №373

АНАЛИЗ ПОДХОДОВ К ПОСТРОЕНИЮ СХЕМ РАЗВОРАЧИВАНИЯ КЛЮЧЕЙ В АЛГОРИТМАХ БЛОЧНОГО СИММЕТРИЧНОГО ШИФРОВАНИЯ

М.Ю. Родинко[1], К.Е. Лисицкий[2]
[1]Харьковский национальный университет радиоэлектроники, Украина
[2]Харьковский национальный университет им. В.Н. Каразина, Украина
m.rodinko@gmail.com, konstantin.lisickiy@mail.ru

Введение

В работе [1] была рассмотрена задача оценки криптографической значимости схем разворачивания ключей в обеспечении стойкости блочных симметричных шифров к атакам линейного и дифференциального криптоанализа. В противоположность существующей точке зрения показано, что схемы разворачивания ключей итеративных (Марковских) шифров не играют существенной роли в обеспечении их стойкости к вышеупомянутым криптоаналитическим атакам. И с нулевыми подключами шифры асимптотически приходят к показателям случайных подстановок. Ключи выполняют лишь функцию осуществления ключезависимого преобразования.

Усложнение схем разворачивания ключей оправдывается стремлением противостоять другим методам криптоанализа, таким, как атаки на связанных ключах, слайд атаки и им подобные. В работе показано, что данная цель может быть достигнута существенно более простыми методами, чем это сделано во многих известных схемах.

Требования к схемам разворачивания ключей

Напомним взгляды разработчиков на требования к схемам разворачивания ключей в современных шифрах.

Ларс Кнудсен [2] полагает, что идея получения сильного ключевого графика состоит в создании подключей таким образом, чтобы задача нахождения связи между любыми битами любых цикловых подключей была практически неразрешимой. Отмечается также побочный эффект – создание такого ключевого графика занимает больше времени, чем традиционные ключевые настройки.

В работе [3] говорится, о том, что весьма желательно, чтобы процедура расширения ключа могла вычислять ключи «на лету» (англ. *on-the-fly*), т.е. параллельно с шифрующими преобразованиями. Кроме того, во многих применениях алгоритмов симметричного шифрования приходится часто менять ключи в шифраторе. Соответственно, весьма сложная процедура расширения ключа не позволит использовать алгоритм шифрования в таких случаях.

Мы считаем, что требования к построению ключевого графика для многих современных шифров можно существенно ослабить. Важно лишь обеспечить отсутствие самоподобия при построении цикловых подключей (они не должны повторяться).

Предложения по построению схем разворачивания ключей

Приведенные в [1] лавинные свойства шифров свидетельствуют о том, что и одного измененного бита вполне достаточно, чтобы сработал механизм ортогонализации зашифрованных текстов. Более сложные схемы разворачивания ключей, на наш взгляд, не будут сколько-нибудь эффективнее простых.

Если полагать, что найти подключ также сложно, как и сам ключ, то можно предложить, по крайней мере, две простых процедуры разворачивания ключей, которые смогут защитить шифры от атак на связанных ключах и других вновь открытых типов атак, в том числе и от отмеченной в [4] сдвиговой атаки.

Далее будем рассматривать шифр со 128-битным размером битового входа и 128-битным мастер-ключом. 128-битный мастер-ключ состоит из 16-ти байтов. Первое предложение состоит в том, чтобы каждые семь последовательных битов (или каждый последовательный усечённый байт) использовать для того, чтобы задать номер бита в мастер-ключе, который будет изменяться в цикловом подключе, формируемом из мастер-ключа для каждого цикла. 128-битного мастер-ключа достаточно, чтобы сформировать подключи для 18-ти циклов. Если потребуется большее число циклов зашифрования, то предложенную процедуру можно сохранить и далее на циклическом продолжении исходного мастер-ключа (для семибитных сегментов).

При равновероятном выборе битов мастер-ключа вероятность r-кратного повторения подключей в r циклах оценивается значением $\left(\frac{1}{2^7}\right)^r$ (в мастер-ключе повторяются семибитные сегменты). Если же распределение битов мастер-ключа

достаточно случайное (например, при формировании мастер-ключа предусмотрена процедура исключения повторений семибитных сегментов), то получается развёрнутый ключ, не имеющий самоподобных фрагментов. Эта процедура проверки может быть выполнена ещё на этапе формирования мастер-ключей, так что при формировании цикловых подключей останется только считать очередной сегмент и выполнить дополнение соответствующего элемента мастер-ключа (позиция которого задаётся считанным сегментом).

Множество разрешённых мастер ключей при этом уменьшается в худшем случае на $128 \cdot \frac{2^n}{2^{7R}} = 2^{n-7R-1}$ ключей. Для $n = 128$ и $R = 10$ получаем $2^{128-69} = 2^{59}$ ключей (заметим, что в шифре ГОСТ 28147-89 половина ключей являются слабыми). На самом деле можно не исключать подряд все повторения или модифицировать их в процессе формирования.

Другой вариант построения развёрнутого ключа – это использование в качестве цикловых подключей циклических сдвигов мастер-ключа, значения которых, как и в первой схеме, задаются его сегментами. Цикловые подключи будут повторяться при повторении соответствующих сегментов мастер-ключа.

Приведём результаты сравнения предлагаемой схемы разворачивания ключей со штатной схемой, используемой в шифре Rijndael, по коллизионным характеристикам [5].

Методика проведения эксперимента состоит в том, что для каждого зашифрованного входного блока данных осуществляется его расшифрование на всём множестве ключей, формируемых схемой разворачивания ключей, и ищутся ключи, при которых результат расшифрования совпадает с исходным открытым текстом. Процедура выполняется для всего множества входных блоков данных.

В табл.1 приведены результаты оценки коллизионных свойств двух схем разворачивания ключей: для схемы разворачивания ключей шифра Rijndael и предлагаемой простой схемы разворачивания ключей, которая названа схемой со сдвигом. В процессе экспериментов каждый текст из выборки 1000 текстов зашифровывался на всех возможных ключах, и оценивалось число повторений шифртекстов. Например, в строке 3 цифра 4017,21 означает, что примерно 4017 шифртекстов повторяются три раза.

Естественно, что повторения шифртекстов приводят к тому, что реализуется неполное множество возможных вариантов шифртекстов. Получается, что из полного множества, равного 2^{16} шифртекстов, фактически реализуется только 0,368 часть.

Как следует из представленных в табл. 1 результатов обе схемы разворачивания ключей демонстрируют практически одинаковые свойства. Для 65536 возможных ключей зашифрования практически только порядка 24100 (0,368 часть) шифртекстов дают биективное преобразование. Остальные 17318 ключей дают коллизии (от 2-ух до 11-ти совпадений).

Таблица 1. Средне число повторений разных шифртекстов для выборки из 65536 ключей зашифрования при сопоставлении двух схем разворачивания ключей

Количество одинаковых шифртекстов	Средне число повторений разных шифртекстов	
	Rijndael	Схема со сдвигом
0	24111,16	24162,95
1	24106,3	24056,14
2	12056,33	12028,09
3	4017,21	4026,45
4	1005,05	1015,32
5	200,75	206,09
6	33,63	35,05
7	4,92	5,18
8	0,56	0,65
9	0,073	0,086
10	0,011	0,005
11	0,001	0,000

В табл. 2 мы представляем результаты оценки множества слабых ключей (ключей, которые приводят при расшифровании к нулевому тексту на выходе дешифратора).

В этом случае текст 0 зашифровывался на первом ключе. Далее осуществлялось его расшифрование на всех 65536-ти ключах и определялось число ключей расшифрования, при которых получался нулевой открытый текст. Далее осуществлялось зашифрование текста 0 на втором ключе и так до 65536. Далее суммировались результаты, т.е. выяснялось, сколько раз расшифрованный текст 0 появлялся один раз (на одном ключе расшифрования), сколько два раза и т.д. Взятая в этом эксперименте выборка – 1000 ключей зашифрования. Результаты вычислений в табл. 1 и табл. 2 представлены для четырех циклов зашифрования.

И в последнем случае результаты свидетельствуют о полной идентичности двух рассмотренных схем разворачивания ключей.

Предложенные выше решения являются обратимыми преобразованиями, т.е. по любому цикловому подключу можно восстановить мастер-ключ. Представленные схемы также можно преобразовать в необратимые функции, например, путём введения дополнительной операции сложения хотя бы двух сформированных цикловых подключей по модулю два (XOR), правда, в этом случае придётся и ключи расшифрования формировать из мастер-ключа.

Таблица 2. Число совпадающих нулевых открытых текстов при расшифровании на всём множестве ключей для двух схем разворачивания ключей

Число совпадающих открытых текстов	Rijndael	Схема со сдвигом
1	357	341
2	357	376
3	197	188
4	70	68
5	11	23
6	6	4
7	2	0

Выводы

Таким образом, предложены два простых алгоритма построения схем разворачивания ключей, которые обеспечивают отсутствие самоподобия цикловых подключей. Преимуществом предложенных схем является возможность вычисления ключей «на лету», недостатком – уменьшение множества допустимых мастер-ключей на 2^{59} (почти наполовину).

Список использованных источников

1. Лисицкая, И.В. О криптографической значимости схем разворачивания ключей в обеспечении стойкости блочных симметричных шифров к атакам линейного и дифференциального криптоанализа / И.В. Лисицкая, А.А. Настенко, К.Е. Лисицкий // Автоматизовані системи управління та прибори автоматики. – 2012. – Вып. 159. – С. 13-21.
2. Knudsen, L. R. On the Role of Key Schedules in Attacks on Iterated Ciphers [Text] / L. R. Knudsen, John Erik Mathiassen // In Samarati et al. (Eds.): ESORICS 2004. – 2004. – LNCS 3193. – pp. 322–334.

3. Панасенко, С. Криптоаналитические атаки на связанных ключах [Электронный ресурс] / С. Панасенко // Режим доступа: URL: - http://old.cio-world.ru/it-market/e-safety/314100/ - 20.04.2007.
4. Biham, E. New Types of Cryptanalytic Attack Using Related Keys [Text] / E. Biham // J. of Cryptology. - 1994. - vol. 7. - pp. 229-246.
5. Advanced Encryption Standard (AES) (FIPS PUB 197) [Text] / National Institute of Standards and Technology. - November 26, 2001.

ОБ ОДНОМ АЛГОРИТМИЧЕСКОМ ПРОЦЕССЕ УСТАНОВЛЕНИЯ ОБЩЕЗНАЧИМОСТИ НЕКОТОРЫХ ФОРМУЛ БЕЗРАНГОВОЙ ЭГАЛИТАРНОЙ ТЕОРИИ

Х.М. Рухая, Л.Т. Тибуа, Г.О. Чанкветадзе
Тбилисский Государственный Университет им.Иванэ Джавахишвили, Сухумский Государственный Университет, Тбилиси, Грузия
khimuri.rukhaia@viam.sci.tsu.ge , ltibua@gmail.com, gelachan@hotmail.com

Введение

Языки, в которых функциональные или предикатные символы не имеют фиксированной арности (местности), в последние годы стали предметом интенсивного изучения по причине довольно широкой сферы их применимости [1]. Обычно встречаются переменные двух типов: предметные переменные, которые можно заменить одним термом, и последовательные переменные (далее мы назовем их «предметными последовательными переменными»), заменить которые можно конечной последовательностью термов. В отличии от вышеупомянутых языков, в изученном нами языке безранговой эгалитарной теории встречаются два типа последовательностей переменных: а) переменные предметной последовательности, представить которые можно конечной последовательностью термов и б) переменные пропозиционной последовательности, заменить которые можно конечной последовательностью формул. Кроме того, область операторов этой теории – $\vee$, $\wedge$, $\supset$, $\overline{x}$, $\exists x$, $\forall x$ не зафиксирована – они безранговые операторы. Определение этих операторов происходит в рамках рациональных правил введения производных операторов Шалвы Пхакадзе [2]. На их основании в

безранговой эгалитарной теории были доказаны аналоги результатов, полученных в эгалитарной теории Н. Бурбаки [3].

Символы языка безранговой эгалитарной теории

Рассмотрим символы языка безранговой эгалитарной теории.

I. Основные символы:

1. Предметные переменные X_0, X_1,... (1)
2. Функциональные символы f, g, h, f_1, g_1, h_1,... (2)
3. Специальный субстанционный двухместный символ $\supset$ (3)
4. Предикатные символы: P, Q, P_1, Q_1,...
5. Специальный реляционный двух местный символ =
6. Логические символы: $\neg$, $\vee$, $\wedge$, τ, $\square$
7. Вспомогательные символы: [,] (,)

II. Символы, внесенные специальными математическими определениями

III. Метасимволы:

1. Предметные метапеременные: x, y, x_1, y_1,… (4)
2. Предметные метапеременные для последовательности термов: z, z_1, …(5)
3. Пропозиционные метапеременные для последовательности формул: φ, φ_1, … (6)
4. Предметные метапеременные для термов: T, U, T_1, U_1,… (7)
5. Пропозиционные метапеременные для формул: A, B, A_1, B_1,… (8)

К понятиям формулы и терма добавляются следующие пункты:

1. $\vee \varphi A$, $\wedge \varphi A$, $\forall x \varphi A$ и $\exists x \varphi A$ являются формулами.
2. $\tau x \varphi A$ и ∋zT являются термами.

Определения некоторых безранговых операторов

1. $\vee' A - A \vee^n A_1 ... A_n - \vee \vee^{n-1} A_1 ... A_{n-1} A_n$, – читается «дизъюнкция формул», $n = 2,3,...$
2. $\wedge' A - A \wedge^n A_1 ... A_n - \wedge \wedge^{n-1} A_1 ... A_{n-1} A_n$, – читается «коньюнкция формул», $n = 2,3,...$

3. $\tau^{n+1} x A_1 \ldots A_n A - \tau x \wedge^{n+1} A_1 \ldots A_n A$, – читается «имеется такой x, который имеет $A_1 \ldots A_n$ и свойство A», $n = 0,1,2,3\ldots$
4. $\exists^{n+1} x A_1 \ldots A_n A - \left(\tau x \wedge^{n+1} A_1 \ldots A_n A / x\right) \wedge^{n+1} A_1 \ldots A_n A$, – читается «существует такой x, который имеет $A_1 \ldots A_n$ и свойство A».
5. $\forall^{n+1} x A_1 \ldots A_n A - \neg \exists^{n+1} x A_1 \ldots A_n \neg A$, – читается «каждый x, который имеет свойство $A_1 \ldots A_n$, а также свойство A».
6. $\supset' T - T, \supset^n T_1 \ldots T_n - \supset \supset^{n-1} T_1 \ldots T_{n-1} T_n$, – читается «n –ка, составленная термами $T_1 \ldots T_n$», $n = 2,3,\ldots$

Заметим, что если из определенных выше операторов удалить верхние индексы, получаем безранговые операторы $\vee$, $\wedge$, $\supset$, τx, $\exists x$ и $\forall x$, то есть операторы, арность которых может быть любое натуральное число (их арность определяем из контекста).

Ниже описан алгоритмический процесс установления общезначимости некоторых формул безранговой эгалитарной теории, который ориентирован на автоматическое доказательство теорем.

При конструировании форм логических теорий особую роль играют скобки – размещение скобок содержит в себе важную информацию о конструкции формы.

Говорят, что в слове C скобки расположены нормально, если имеется такое множество D парных скобок, составленных из скобок слова C, для которого выполняются следующие условия:

1. Каждая скобка слова C (т.е. каждое вхождение скобки в слове C) входит в одну и только в одну пару скобок, взятых из D.
2. Первый член каждой пары скобок, взятых из D, является левой скобкой, а второй член – правой. При этом первый член в слове C стоит слева второго члена.
3. Для любой из двух пар скобок, взятых из D, или одна из них находится внутри другой пары скобок, или ни одна из них не содержит ни одного члена другой пары скобок.

Назовем нормальным множество пар скобок слова C, если оно удовлетворяет условиям 1 – 3.

Пару скобок, составленных из двух скобок слова C, назовем внутренней парой скобок если она составлена из левой скобки, взятой из слова C, и такой правой скобки, лежащей справа от этой левой скобки, что внутри этих скобок нет других скобок.

Для того, чтобы найти нормальное множество пар скобок слова C, соответственно его части, достаточно пронумеровать скобки натуральными числами так, чтобы одинаковые номера имели только лишь взаимосоответствующие скобки. Ниже мы опишем алгоритм, который выяснит, нормально расположены скобки в слове C или нет. И если скобки расположены нормально, то найдем вышеуказанное нумерование нормального множества пар скобок слова C. Этот алгоритм состоит в следующем.

Найдем первую левую скобку без номера и последующую первую правую скобку без номера, будем считать номером этих скобок число k ($k = 0,1,...$) и припишем этим скобкам снизу индекс k. Этот процесс завершается. Скобки будут расположены нормально в слове C тогда и только тогда, когда упомянутый процесс исчерпает множество скобок слова C. Если это условие выполняется, то пары скобок, составленные из скобок, имеющих один и тот же номер, составят нормальное множество пар скобок слова C. Справедлива следующая теорема.

Теорема 1. Каждая пара скобок нормального множества пар скобок формы Ф однозначно определяет различные от переменных формы Ф части.

Связанные метаформулы формул обсужденной выше безранговой эгалитарной теории определяются следующим алгоритмическим процессом.

Напишем на первой строке первоначальную самую формулу. Найдем в формуле все такие кванторы, операторной переменной которых является переменная, взятая из последовательности (1) и которая не действует на других кванторах этого же типа. Если такие кванторы существуют, тогда все буквенные группы, связанные этими кванторами, заменим такими метапеременными взятыми из последовательности (4), которые не входят в данную формулу. Полученную этим путем метаформулу и осуществленную подстановку запишем во второй строке.

Таким же путем из второй строки получим третью строку, если в ней имеется квантор обозначенного типа. Этот процесс обязательно завершится на некоторой строке k записью формулы и соответствующей подстановки ($k \in \{1,2,..\}$)

С правой стороны в записанной k строке метаформулы найдем свободное вхождение переменной, взятой из последовательности (1). Группу свободных переменных, представителем которой является указанная переменная, заменим такой метапеременной из последовательности (4), которая не входит в данную формулу. Полученную таким образом формулу и осуществленную подстановку запишем в $(k+1)$ строке. Тем же самым образом из $(k+1)$ строки получим $(k+2)$ строку, если в ней опять найдется свободное вхождение переменной, взятой из последовательности (1).

Этот процесс обязательно завершится на какой-нибудь $(k+e)$ строке записью формулы и соответственной подстановки ($e \in \{1,2,..\}$).

С правой стороны в записанной в $(k+e)$ строке метаформуле найдем первую внутреннюю пару скобок. Составленная из этих скобок часть будет иметь вид: $[\tau xA]$ или, $[fx_1...x_n]$, или $[x_1 = x_2]$, или $[Px_1...x_n]$. Если часть, составленная отмеченными скобками, имеет вид $[\tau xA]$ или $[fx_1...x_n]$, то данные и ее графически равные части заменим первой такой метапеременной из последовательности (7), которая не входит в данную формулу. Полученную таким путем метаформулу и осуществленную подстановку запишем на $(k+e+1)$ строке. А если часть, составленная отмеченными скобками имеет вид $[x_1 = x_2]$ или $[Px_1...x_n]$, то данные и ее графически равные части заменим первой такой метапеременной из последовательности (8), которая не входит в данную формулу. Полученную таким путем метаформулу и осуществленную подстановку запишем на $(k+e+1)$ строке. Из $(k+e+1)$ строки аналогичным образом получаем $(k+e+2)$ строку, если в ней опять найдутся части типа $[fx_1...x_n]$ или $[x_1 = x_2]$, или $[Px_1...x_n]$.

Этот процесс обязательно закончится на какой-нибудь $(k+e+n)$ строке записью формулы и соответственной подстановки ($n \in \{1,2,..\}$).

Найдем в записанной на $(k+e+n)$ строке метаформуле первую такую внутреннюю пару скобок, в которой

соответствующая часть формулы имеет вид $\left[\exists x A_1 \ldots A_n A\right]$ или $\left[\forall x A_1 \ldots A_n A\right]$. Указанные и графически равные части формулы заменим первой такой метапеременной из последовательности (8), которая не входит в данную формулу. Полученную таким путем метаформулу и осуществленную подстановку запишем на $\left(k+e+n+1\right)$ строке. Если не существует такой внутренней части, то в записанной строке и метаформуле найдем первую внутреннюю пару скобок. Соответствующая часть этой пары скобок будет иметь вид $\left[\neg A\right]$ или $\left[A \vee B\right]$, или $\left[A \wedge B\right]$, тогда указанные части формулы и ее графически равные части заменим первой такой метапеременной из последовательности (8), которая не входит в данную формулу. Полученную таким путем метаформулу и осуществленную подстановку запишем на $\left(k+e+n+1\right)$ строке.

Легко заметить, что упомянутый алгоритмический процесс завершится, потому что число символов, входящих в записанную на каждой последующей строке метаформулу, меньше или равно числу символов метаформулы, записанной в предыдущей строке. В завершающую строку записываем метапеременную из последовательности (4) и соответствующая подстановка.

Формулы, полученные рассмотренным алгоритмическим процессом, назовем связанными с данной формулой метаформулами. Если связанная с данной формулой метаформула состоит только из метапеременных из последовательности (8) и символов $\neg$, $\vee$, $\wedge$, назовем ее связанной с данной формулой пропозиционной метаформулой. Если пропозиционная метаформула, рассмотренная как формула логики высказываний, является тавтологией, то ее называют пропозиционной тавтологической формой.

Пропозиционную тавтологическую форму, формулу типа $\left[T = T\right]$ и также полученную формулу распространением над ней кванторов $\exists$ и $\forall$, назовем тавтологической формой. Непосредственным следствием выше описанного алгоритма является следующая

Теорема 2. Для общезначимости формулы A достаточно, чтобы существовала связанная с формулой A тавтологическая форма.

Доказательство. Допустим, что существует связанная с формулой A тавтологическая форма и покажем, что A истинна в произвольной интерпретации, определенной на произвольном

пространстве. Выпишем все подстановки, которые были использованы в формуле A для получения связанной с ней формы. Рассмотрим произвольную интерпретацию. Очевидно, что каждая часть формулы A будет иметь определенное значение в данной интерпретации. Если вместо части C формулы A вставлена метапеременная B_1, то подразумевается, что значение переменной B_1 в данной интерпретации является значением части C. Очевидно, что в обсуждаемой интерпретации значение формулы A совпадает с значением, полученным указанной подстановкой формулы A_1. Аналогично, значение формулы A_1 совпадает с значением, полученным тем же путем формулы A_2 и т. д. Исходя из этого, легко убедиться, что в данной интерпретации значение формулы A совпадет со значением «истина», связанной с ней формулы A_n, то есть формула A общезначима.

Если формула A получена из тождественно истинной формулы логики высказываний B путем осуществления конечного количества подстановок, то формула A называется частным случаем подстановочной тавтологии.

Теорема 3. Для того, чтобы формула A являлась частным случаем подстановочной тавтологии, необходимо и достаточно существование связанной с формулой A пропозиционной тавтологической формы.

Выводы

В работе на основе безранговых операторов в безранговой эгалитарной теории были получены аналоги трех результатов из эгалитарной теории Н. Бурбаки.

Работа выполнена при поддержке гранта (D/16/4-120/11) национального научного фонда Грузии им. Шота Руставели

Список используемых источников

1. Kutsia T. Theorem Proving with Sequence Variables and Flexible Arity Symbols / T. Kutsia // 9th International Conference, LPAR 2002 Tbilisi, Georgia, October 14–18, 2002 Proceedings. – 2002. – P. 278-291.
2. Пхакадзе Ш.С. Некоторые вопросы теории обозначений / Ш.С. Пхакадзе. – Тбилиси: Изд-во Тбилисского университета, 1977. – 195 с.
3. Бурбаки Н. Теория Множеств / Н. Бурбаки. – Москва: Мир, 1965. – 456 с.

4. Rukhaia Kh. One Method of Constructing a Formal System / Kh. Rukhaia, L. Tibua, G. Chankvetadze, B. Dundua // Applied Mathematics, Informatics and Mechanics. – 2006. – V. 11, N. 2. – P. 81-89.

КЛЮЧИ И ОПЕРАЦИИ В ТАБЛИЧНЫХ АЛГЕБРАХ

А.С. Сенченко
Киевский национальный университет имени Тараса Шевченко
senchenko@pisem.net

Введение

В настоящее время информационные системы широко используются практически во всех областях деятельности человека. Базы данных являются ядром для подавляющего большинства информационных систем. При всем разнообразии различных типов баз данных (объектно-ориентированные, графовые, иерархические, объектно-реляционные, облачные, сетевые), наиболее распространенными остаются реляционные базы данных, математическая модель которых была впервые предложена Э. Коддом в 1970 году [1]. С математической точки зрения реляционная база данных является конечным множеством конечных отношений различной арности между заранее определёнными множествами элементарных данных, то есть реляционная база данных – конечная модель.

Табличные алгебры, введённые В.Н. Редько и Д.Б. Буем [2], построены на основе реляционных алгебр Э. Кодда и существенно их развивают. Они составляют теоретический фундамент языков запросов современных табличных баз данных. Элементы носителя табличной алгебры уточняют реляционные структуры данных, а сигнатурные операции построены на базе основных табличных манипуляций в реляционных алгебрах и SQL-подобных языках.

В реляционных базах данных важную роль играют ключи таблицы – один или несколько ее атрибутов, на значениях которых записи таблицы однозначно идентифицируются. С помощью ключей (первичных и внешних) устанавливаются бинарные связи типа «один-ко-многим». Эти связи служат для поддержания целостности баз данных. Как правило, ключи определяются таким образом, чтобы они были инвариантными к любым изменениям записей в базе данных. В настоящей работе рассматривается вопрос сохранения ключей таблиц, полученных в результате применения к ним сигнатурных операций табличных алгебр. Полученные

результаты представляют интерес для выбора оптимальных ключей при проектировании реляционных баз данных [3, 4].

Основные определения

Зафиксируем некоторое непустое множество атрибутов $A = \{A_1, \ldots, A_n\}$. Произвольное конечное подмножество множества A назовем схемой, причем схема может быть пустым множеством. Строкой s схемы R называется множество пар $s = \{(A'_1, d_1), \ldots, (A'_k, d_k)\}$, проекция которого по первой компоненте равна R, причем атрибуты $A'_1, \ldots, A'_k$ попарно различны, то есть строка является функциональным бинарным отношением. Таблицей схемы R называется конечное множество строк схемы R, количество строк в таблице T обозначаем $|T|$. Далее в работе рассматриваем таблицы схемы R с количеством атрибутов k. На множестве всех таких таблиц введены следующие операции объединения, пересечения и разности таблиц как ограничение одноименных теоретико-множественных операций на множество таблиц.

Для введения операции насыщения необходимо одно вспомогательное понятие. Активным доменом атрибута A относительно таблицы T называется множество $D_{A,T} = \{d \mid \exists s \in T \wedge (A, d) \in s\}$, состоящее, говоря содержательно, из всевозможных значений атрибута A в таблице T (если же атрибут не входит в схему таблицы, то его активный домен пуст). Атрибуты таблицы, мощность активных доменов которых больше единицы, назовем многозначными, в противном случае атрибуты называем однозначными. Насыщением $C(T)$ называется таблица $\prod_{A \in R} D_{A,T}$, где R – схема таблицы T, а $\prod$ – оператор прямого (декартового) произведения, отвечающий индексированию $A \mapsto D_{A,T}$, $A \in R$ [5]. Активным дополнением таблицы T называется таблица $\widetilde{T} = C(T) \underset{R}{-} T$.

Введем определение операции проекции. Проекцией по множеству атрибутов $X \subseteq R$ называется унарная параметрическая операция π_X, значением которой является таблица, состоящая из ограничений по X всех строк исходной

таблицы: $\pi_X(T) = \{s|X \quad |s \in T\}$. Здесь ограничение понимается стандартно: $s|X = s \cap (X \times pr_2\, s)$, где $pr_2\, s$ – проекция строки s по второй компоненте.

Введем определение операции селекции. Селекцией по предикату $P : S \to \{true, false\}$, где S – множество всех строк, называется унарная параметрическая операция σ_P, которая таблице сопоставляет ее подтаблицу, содержащую строки, на которых предикат P принимает истинное значения.

Для введения операции соединения необходимо одно вспомогательное понятие. Бинарные отношения ρ и τ называются совместными (обозначается $\rho \approx \tau$), если $\rho|X = \tau|X$, где $X = pr_1\, \rho \cap pr_1\, \tau$ [6]. Соединением называется бинарная операция $\otimes$, значением которой является таблица, состоящая из всевозможных объединений совместных строк исходных таблиц, то есть $T_1 \otimes T_2 = \{s_1 \cup s_2 |\ s_1 \in T_1 \wedge s_2 \in T_2 \wedge s_1 \approx s_2\}$. Для таблицы T со всеми однозначными атрибутами из множества $O = \{O_1, \ldots, O_z\}$ и значениями их активных доменов $D_{O_1,T} = \{o_1\}, \ldots, D_{O_z,T} = \{o_z\}$ из определения операции соединения очевидно следует равенство $T = \pi_Y(T) \otimes T'$, где $Y = R - O$ и $T' = \begin{matrix} O_1 & \ldots & O_z \\ o_1 & \ldots & o_z \end{matrix}$, то есть $T' = \{\{\langle O_i, o_i \rangle | i = \overline{1,z}\}\}$.

Введем определение операции деления таблиц. Пусть R_1 – схема таблицы T_1, R_2 – схема таблицы T_2 и $R_2 \subseteq R_1$. Делением таблицы T_1 на таблицу T_2 называется таблица схемы $R_1 - R_2$: $T_1 \div \begin{matrix} R_1 \\ R_2 \end{matrix} T_2 = \{s \in \pi_{R_1 - R_2}(T_1) |\ \{s\} \otimes T_2 \subseteq T_1\}$. Если таких строк s в таблице $\pi_{R_1 - R_2}(T_1)$ не существует, считаем, что в этом случае $T_1 \div \begin{matrix} R_1 \\ R_2 \end{matrix} T_2 = T_\varnothing$.

Введем определение операции переименования атрибутов. Переименованием называется унарная, в общем случае частичная

операция RT_ξ, где ξ – инъективное отображение на множестве атрибутов. Эта операция осуществляет только переименование атрибутов таблиц в соответствии с отображением-параметром ξ. Содержательно говоря, переименование таблицы сводится к переименованию первых компонент пар – элементов строк.

Табличной алгеброй называют частичную алгебру с носителем – множеством всех таблиц произвольной схемы – и приведёнными выше девятью операциями (насыщение рассматривается как вспомогательная операция). В табличной алгебре выделяют две особые таблицы: таблицу $T_\varepsilon = \{\varepsilon\}$, где ε – пустая строка, при этом схема таблицы T_ε является пустым множеством, и таблицу $T_\varnothing = \varnothing$ – пустое множество строк произвольной (в том числе и непустой) схемы. Таблица T, не являющаяся особой, называется ненасыщенной, если выполняется неравенство $\widetilde{T} \neq T_\varnothing$ (очевидно, что в этом случае $T \neq C(T)$).

Множество атрибутов $K \subseteq R$ называется ключом таблицы T, если для любых строк $s_1, s_2 \in T$ выполняется импликация $s_1|K = s_2|K \to s_1 = s_2$; другими словами, ограничения по атрибутам ключа всех строк таблицы T попарно различны. Несложно заметить, что схема таблицы будет являться ее ключом, поэтому наибольший интерес представляют так называемые нетривиальные ключи, которые являются собственным подмножеством схемы таблицы. Из определения ключа следует, что ключом таблицы $T_\varnothing$ будет любое подмножество атрибутов ее схемы. Для таблицы T_ε ключом будет являться пустое множество. На практике в реальных базах данных особые таблицы используются чрезвычайно редко (особенно таблица T_ε), поэтому для удобства изложения результатов в работе рассматриваются исключительно таблицы, не являющиеся особыми, при этом большинство результатов справедливо и для таблицы $T_\varnothing$. Также из определения ключа очевидно следует, что пустое множество может быть ключом только для особых таблиц или для таблиц, состоящих из одной строки.

Основные результаты

Поскольку ключи играют важную роль в реляционных базах данных естественным становится вопрос о сохранении нетривиальных ключей сигнатурными операциями табличных алгебр.

Утверждение 1 (сохранение ключей операцией пересечения). Пусть T_1, T_2 – таблицы схемы R и K – ключ таблиц T_1 и (или) T_2. Тогда K – ключ таблицы $T_1 \bigcap\limits_{R} T_2$.

Утверждение 2 (сохранение ключей операцией разности таблиц). Пусть T_1, T_2 – таблицы схемы R и K – ключ таблицы T_1. Тогда K является ключом таблицы $T_1 \underset{R}{-} T_2$.

Утверждение 3 (сохранение ключей операцией селекции). Пусть K – ключ таблицы T. Тогда K – ключ таблицы $\sigma_P(T)$.

Теорема 1 (сохранение ключей операцией активного дополнения для таблиц без однозначных атрибутов). Пусть $R = \{A_1, \ldots, A_n\}$ – схема ненасыщенной таблицы T с многозначными атрибутами и $K = \{K_1, \ldots, K_q\}$ – нетривиальный ключ для T. K является ключом таблицы $\widetilde{T}$ тогда и только тогда, когда одновременно выполняются три условия:

а) $|R - K| = 1$;

б) для атрибута B, принадлежащего множеству $R - K$, выполняется равенство $|D_{B,T}| = 2$;

в) для всех значений $d_1 \in D_{K_1,T}, \ldots, d_q \in D_{K_q,T}$ строка $s' = \{(K_1, d_1), \ldots, (K_q, d_q)\} \in \pi_K(T)$, причем существует только одна такая строка $s \in T$, что $s' = s|\{K_1, \ldots, K_q\}$.

Следствие 1 (сохранение ключей операцией активного дополнения для таблиц с однозначными атрибутами). Пусть $R = \{A_1, \ldots, A_n\}$ – схема ненасыщенной таблицы T и $K = \{K_1, \ldots, K_q\}$ – нетривиальный ключ для T. K является ключом таблицы $\widetilde{T}$ тогда и только тогда, когда одновременно выполняются три условия:

а) множество $R-K$ содержит в точности один многозначный атрибут;

б) для многозначного атрибута B, принадлежащего множеству $R-K$, выполняется равенство $\left|D_{B,T}\right|=2$;

в) для всех значений $d_1 \in D_{K_1,T},\ldots,d_q \in D_{K_q,T}$ строка $s'=\left\{\left(K_1,d_1\right),\ldots,\left(K_q,d_q\right)\right\}\in \pi_K(T)$, причем существует только одна такая строка $s \in T$, что $s'=s\left|\left\{K_1,\ldots,K_q\right\}\right.$.

Теорема 2 (сохранение ключей операцией проекции). Пусть K – ключ таблицы T и $K \subseteq X$. Тогда K является ключом таблицы $\pi_X(T)$.

Теорема 3 (сохранение ключей операцией соединения). Пусть K – ключ таблиц T_1 и T_2. Тогда K – ключ таблицы $T_1 \otimes T_2$.

Теорема 4 (сохранение объединения ключей операцией соединения). Пусть K_1 – ключ таблицы T_1 и K_2 – ключ таблицы T_2. Тогда $K_1 \bigcup K_2$ является ключом таблицы $T_1 \otimes T_2$.

Теорема 5 (сохранение ключей операцией деления). Пусть R_1 – схема таблицы T_1, R_2 – схема таблицы T_2, $T_1 \div \dfrac{R_1}{R_2} T_2 \neq T_\varnothing$, $K=\left\{K_1,\ldots,K_q\right\}$ – ключ таблицы T_1 и $K \bigcap R_2 = \varnothing$. Тогда K – ключ таблицы $T_1 \div \dfrac{R_1}{R_2} T_2$.

Теорема 6 (сохранение части ключа операцией деления). Пусть R_1 – схема таблицы T_1, R_2 – схема таблицы T_2, $T_1 \div \dfrac{R_1}{R_2} T_2 \neq T_\varnothing$, K – ключ таблицы T_1, $K \bigcap R_2 \neq \varnothing$ и $K'=K-\left(K \bigcap R_2\right)\neq \varnothing$. Тогда K' – ключ таблицы $T_1 \div \dfrac{R_1}{R_2} T_2$.

Утверждение 4 (сохранение ключей операцией переименования). Пусть $K = \{K_1, \ldots K_q\}$ – ключ таблицы T и пусть $\xi[K] = \{\xi(K_1), \ldots, \xi(K_q)\}$. Тогда $\xi(K)$ является ключом таблицы RT_ξ.

Выводы

В работе исследовано сохранение ключей сигнатурными операциями табличных алгебр. Показано, что операции пересечения, разности, селекции и переименования атрибутов сохраняют ключи (утверждения 1 – 4); найдены необходимые и достаточные условия, при которых операция дополнения сохраняет ключи (теорема 1, следствие 1). Также найден критерий, при котором операция проекции сохраняет ключ (теорема 2). Показано, что если две таблицы имеют одинаковые ключи, то их соединение сохраняет ключ (теорема 3), а если две таблицы имеют разные ключи, то объединение ключей будет ключом соединения исходных таблиц (теорема 4). Доказано, что операция деления сохраняет ключи (теоремы 5, 6). Результаты работы представляют теоретический и практический интерес и могут быть использованы для выбора оптимальных ключей при проектировании реляционных баз данных

Список используемых источников

1. Codd E.F. A Relational Model of Data for Large Shared Data Banks / E. F. Codd // Communications of the ACM. – 1970. – V. 13, N. 6. – P. 377–387.
2. Конноли Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика. / Т. Конноли, К. Бегг. – Москва: «Вильямс», 2003. – 1440 с.
3. Редько В.Н. К основаниям теории реляционных моделей баз данных / В.Н. Редько, Д.Б. Буй // Кибернетика и системный анализ. – 1996. – №4. – С. 3 – 12.
4. Дейт К.Дж. Введение в системы баз данных, 8-е издание: [пер. с англ.] / К.Дж. Дейт. – Москва: «Вильямс», 2005. – 1328 с.
5. Куратовский К. Топология: в 2 т. / К. Куратовский. – Т. 1. – Москва: Мир, 1966. – 594 с.
6. Реляційні бази даних: табличні алгебри та SQL-подібні мови / [В.Н. Редько, Ю.Й. Брона, Д.Б. Буй, С.А. Поляков]. – Київ: «Академперіодика», 2001. – 198 с.

АЛГОРИТМ РАСПОЗНАВАНИЯ КОНЕЧНЫХ ГРАФОВ С ПОМОЩЬЮ ТРЕХ АГЕНТОВ

А.В. Стёпкин
Донбасский государственный педагогический университет,
Славянск, Украина
stepkin.andrey@rambler.ru

Введение

Важным и актуальным вопросом кибернетики является распознавание неизвестной среды [1], зачастую представленной в виде конечного графа, в котором агенты обходят все вершины и возвращаются в исходную позицию, построив карту графа [2].

В работе рассматривается алгоритм распознавания графа коллективом агентов с использованием процедуры релокации агента, завершившего распознавание своего подграфа. Процедура релокации позволяет агентам-исследователям более равномерно разделять граф на подграфы для распознавания; не требует дополнительных вычислений и шагов для поиска вершин, подобранных по какому-либо фиксированному критерию; также не требует дополнительной информации о нахождении другого агента вблизи вершин. При выполнении указанной процедуры агенты не мешают работе друг друга и не дублируют проделанную ранее работу. Это, в конечном итоге, дает возможность агентам искать новые подграфы для распознавания не требующую повышения верхних оценок рассматриваемых сложностей. Недостатком предложенного алгоритма является невозможность предварительного определения размера обнаруженного подграфа, требующего распознавания. Это может привести к тому, что новый подграф для распознавания окажется небольшим и процедуру релокации агента придется повторять.

Для работы используются два разных типа агентов. Рассмотрим функции различных агентов. Два агента-исследователя передвигаются по связному конечному неориентированному графу без петель и кратных ребер [3]. Они записывают в память вершин номера в двоичном коде, которые они на каждом шаге получают от агента-экспериментатора, а также считывают и изменяют окраску элементов графа и обмениваются необходимой информацией с агентом-экспериментатором. Агент-экспериментатор передает, принимает и идентифицирует сообщения агентов-исследователей и восстанавливает исследуемый граф по полученным данным.

Принцип работы алгоритма основан на стратегии поиска в глубину [4] и заключается в том, что агенты-исследователи идут «в глубину», пока это возможно, затем возвращаются назад, ищут

другой путь с ещё не посещенными вершинами и не пройденными ребрами.

Для рассмотренного в работе алгоритма доказаны следующие теоремы.

Теорема 1. Выполнив алгоритм распознавания на конечном неориентированном графе, агенты распознают этот граф с точностью до изоморфизма.

Теорема 2. Временная сложность алгоритма распознавания равна $O(n)$, емкостная сложность – $O(n^2)$, а коммуникационная $O(n^2 \cdot \log n)$, где n – число вершин графа.

Выводы

Предложен алгоритм распознавания графа коллективом агентов с использованием процедуры релокации агентов-исследователей, что позволило агентам более равномерно разбивать граф на подграфы для распознавания без увеличения сложности выполнения алгоритма [5]. Алгоритм имеет линейную временную и квадратичную емкостную сложности. Коммуникационная сложность алгоритма составляет $O(n^2 \cdot \log n)$, где n – число вершин графа.

Список используемых источников

1. Albers S. Exploring unknown environments. / S. Albers, M. R. Henzinger // SIAM Journal on Computing. – 2000. – 29(4). – P. 1164 – 1188.
2. Стёпкин А. В. Возможность и сложность распознавания графов тремя агентами / А. В. Стёпкин // Таврический вестник информатики и математики. – 2012. – №1 (20). – с. 88 – 98.
3. Стёпкин А. В. Распознавание конечных неориентированных графов коллективом агентов / А. В. Стёпкин // Журнал обчислювальної та прикладної математики. –2013. – №2(112). – С. 161–168.
4. Кормен Т. Алгоритмы: построение и анализ./ Т. Кормен, Ч. Лейзерсон, Р Ривест. – М.: МЦНМО, 2001. – 960 с.
5. Стёпкин А. В. Использование коллектива агентов для распознавания графов / А. В. Стёпкин // Компьютерные исследования и моделирование. – 2013. – Т.5, №4. – С. 525-532.

РЕАЛІЗАЦІЯ НЕЧІТКОЇ НЕЙРОННОЇ МЕРЕЖІ ДЛЯ РОЗПІЗНАВАННЯ НАДЗВИЧАЙНИХ СИТУАЦІЙ У ПАКЕТІ MATLAB

В.Ю. Тітова
Хмельницький національний університет, Україна
sobaka2032@rambler.ru

Вступ

З точки зору диспетчера служби швидкого реагування надзвичайна ситуація – це порушення нормальних умов життя і діяльності людей на об'єкті або території, спричинене аварією, катастрофою, стихійним лихом, епідемією, епізоотією, епіфітотією, великою пожежею, застосуванням засобів ураження, що призвели або можуть призвести до людських і матеріальних втрат.

Після надходження інформації про надзвичайну ситуацію та її первинної обробки, наступною задачею, що постає перед диспетчером, є розпізнавання надзвичайної ситуації, тобто віднесення її до одного з відомих класів та визначення на основі цього наступних дій для вирішення ситуації.

Задача розпізнавання надзвичайної ситуації характеризується великою кількістю вхідних та вихідних параметрів і зв'язків між ними та відноситься до важкоформалізованих задач, а тому забезпечення диспетчеру служби швидкого реагування інформаційно-аналітичної підтримки прийняття рішень за рахунок використання інтелектуальних методів є актуальною науковою задачею.

Структура нейронної мережі для розпізнавання надзвичайних ситуацій.

Структура нейромережі зображена на рис. 1.

Вона складається з трьох шарів нейронів та має 10 входів: v_1 – людність місця ситуації, v_2 - небезпечність місця ситуації, c_1 – абсолютний час ситуації, c_2 – відносний час ситуації, v_3 - площа, яку охоплює ситуація, $p_1..p_5$ - події, які характеризують ситуацію.

v_1, v_2 приймають значення з діапазону [0..3], де 0 – відсутність людності/небезпеки, 1 – низький рівень людності/небезпеки, 2 – середній рівень людності/небезпеки, 3 – високий рівень людності/небезпеки.

c_1 приймає значення з діапазону [0..3], де 0 – «глухі години», на вулиці люди майже відсутні, 1 – години, коли на вулиці

буває небагато людей, 2 – години, коли на вулиці буває багато людей, 3 – години-пік.

c_2 приймає значення з діапазону [0..3], де 0 – якщо з моменту виникнення ситуації пройшло не більше 2 годин, 1 – якщо з моменту виникнення ситуації пройшло 2-24 години, 2 – якщо з моменту виникнення ситуації пройшло 24-48 годин, 3 – якщо з моменту виникнення ситуації пройшло більше 48 годин.

v_3 приймає значення з діапазону [0..10], де 0 – це мінімальна площа, яка може бути охопленою ситуацією в межах об'єкта, 10 – максимальна площа, яка може бути охоплена ситуацією в межах кількох регіонів.

$p_1..p_5$ приймають значення з діапазону [0..100], де 100 - максимальна кількість можливих подій, що характеризують ситуацію, номер події визначається в залежності від того, які сили для свого вирішення вона потребує.

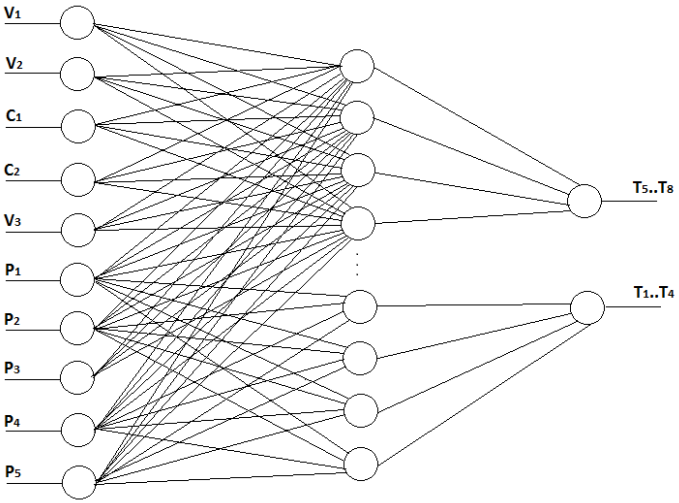


Рис. 1. Структура нейромережі для розпізнавання надзвичайних ситуацій

Кількість виходів мережі два.

Перший вихід визначає тип ситуації за рівнем $t_5..t_8$, може приймати значення в межах [0..1], вихідне значення в межах [0..0,35] відповідає надзвичайній ситуації об'єктового рівня, вихідне значення в межах [0,2..0,55] - надзвичайній ситуації місцевого рівня, вихідне значення в межах [0,4..0,75] -

надзвичайній ситуації регіонального рівня, вихідне значення в межах [0,6..1] відповідає надзвичайній надзвичайній ситуації загальнодержавного рівня.

Другий вихід визначає тип ситуації за характером $t_1..t_4$, може приймати значення в межах [0..1], вихідне значення в межах [0..0,35] відповідає надзвичайній ситуації природного характеру, вихідне значення в межах [0,2..0,55] - надзвичайній ситуації техногенного характеру, вихідне значення в межах [0,4..0,75] - надзвичайній ситуації соціального характеру, вихідне значення в межах [0,6..1] відповідає надзвичайній ситуації воєнного характеру.

Для побудови нечіткої мережі для розпізнавання надзвичайних ситуацій доцільно буде скористатися прикладним пакетом Fuzzy Logic Toolbox програми Matlab, яка на сьогоднішній день є одним з найпопулярніших та найефективніших програмних середовищ розробки штучних нейронних мереж [1].

Проте, використання зазначеного пакету накладає певні обмеження на реалізацію мережі. Зокрема, програмне середовище не дозволяє здійснювати навчання нейронних мереж, що мають більше одного виходу.

Обійти це обмеження можливо, якщо проаналізувати усі взаємозв'язки та залежності між вхідними змінними мережі та можливими типами ситуацій. Як можна побачити зі структури нейромережі на рівень ситуації впливають усі вхідні дані, у той час, як на характер ситуації впливають лише події ситуації.

Отже, отримуємо дві мережі, перша з яких визначає тип ситуації за характером (рис. 2), друга – тип ситуації за рівнем (рис.3).

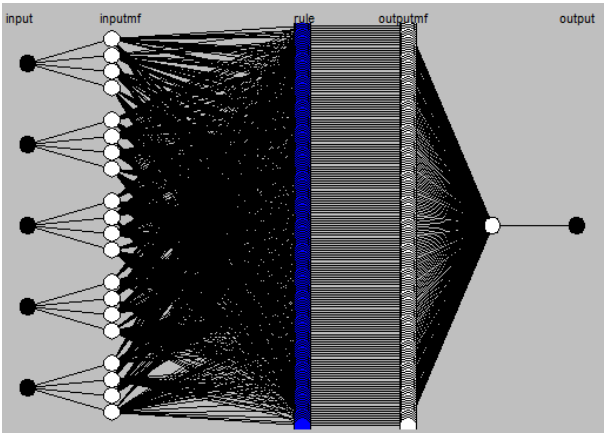


Рис. 2. Структура нейромережі для визначення типу надзвичайної ситуації за характером

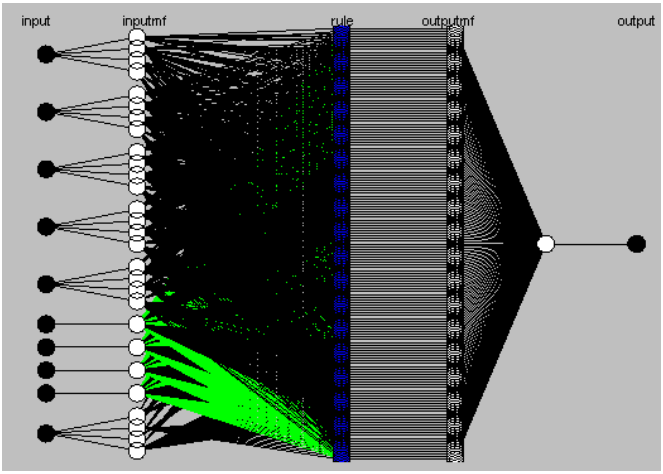


Рис. 3. Структура нейромережі для визначення типу надзвичайної ситуації за рівнем

Обидві мережі є тришаровими. Нейрони першого шару першої мережі визначають ступінь належності вхідних змінних $p_1..p_5$ до нечітких множин характеру ситуації (рис.4).

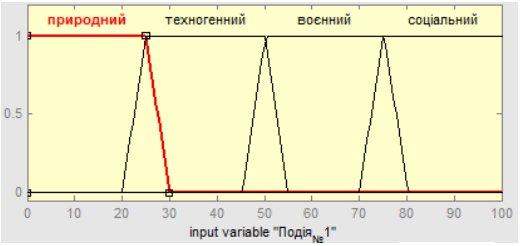


Рис. 4. Функції належності вхідних змінних $p_1..p_5$ до нечітких множин характеру ситуації

Нейрони першого шару другої мережі визначають ступінь належності вхідних змінних $p_1..p_5$ до нечітких множин рівня ситуації (рис.5), вхідної змінної v_3 до нечітких множин рівня ситуації (рис. 6) та вхідних змінних v_1, v_2, c_1, c_2 (рис.7) .

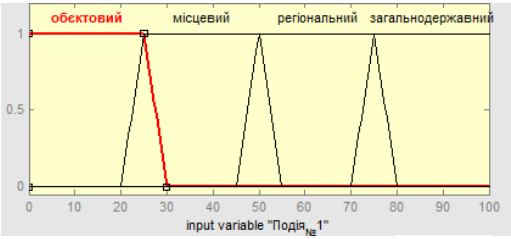


Рис. 5. Функції належності вхідних змінних $p_1..p_5$ до нечітких множин рівня ситуації

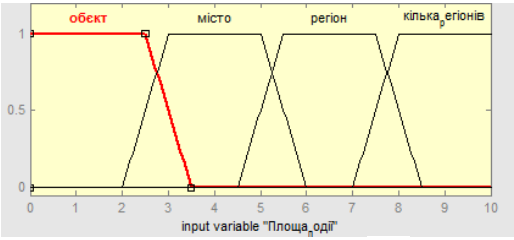


Рис. 6. Функції належності вхідної змінної v_3 до нечітких множин рівня ситуації

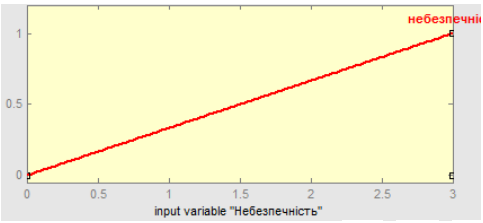


Рис. 7. Функція належності змінних v_1 , v_2 , c_1 , c_2

Виходами нейронів другого шару першої мережі є ступені істинності для кожного з 163-ох правил, які визначають залежності між подіями, що характеризують надзвичайну ситуацію та можливими типами ситуації за характером. Формат правил є наступним:

1. if (*Подія№1* is *природний*) and (*Подія№2* is *природний*) and (*Подія№3* is *природний*) and (*Подія№4* is *природний*) and (*Подія№5* is *природний*) then (*Тип_ситуації* is *природний*).
2. if (*Подія№1* is *техногенний*) and (*Подія№2* is *природний*) and (*Подія№3* is *природний*) and (*Подія№4* is *природний*) and (*Подія№5* is *природний*) then (*Тип_ситуації* is *природний*).

3. if (*Подія№1* is *природний*) and (*Подія№2* is *техногенний*) and (*Подія№3* is *природний*) and (*Подія№4* is *природний*) and (*Подія№5* is *природний*) then (*Тип_ситуації* is *природний*).

.

.

162. if (*Подія№1* is *соціальний*) and (*Подія№2* is *соціальний*) and (*Подія№3* is *соціальний*) and (*Подія№4* is *соціальний*) and (*Подія№5* is *воєнний*) then (*Тип_ситуації* is *соціальний*).

163. if (*Подія№1* is *соціальний*) and (*Подія№2* is *соціальний*) and (*Подія№3* is *соціальний*) and (*Подія№4* is *соціальний*) and (*Подія№5* is *соціальний*) then (*Тип_ситуації* is *соціальний*).

Виходами нейронів другого шару другої мережі є ступені істинності для кожного з 652-ох правил, які визначають залежності між подіями, що характеризують надзвичайну ситуацію, площею, яку охоплює ситуація, показниками людності, небезпечності, абсолютним і відносним часом ситуації та можливими типами ситуації за рівнем. Формат правил є наступним:

1. if (*Подія№1* is *об'єктовий*) and (*Подія№2* is *об'єктовий*) and (*Подія№3* is *об'єктовий*) and (*Подія№4* is *об'єктовий*) and (*Подія№5* is *об'єктовий*) and (*Небезпечність* is *not небезпечність)* and (*Людність* is *not людність)* and (*Відносний_час* is *not відносний_час)* and *(Абсолютний_час* is *not абсолютний_час)* and (*Площа_ситуації* is *об'єкт*) then (*Рівень_ситуації* is *об'єктовий*).

2. if (*Подія№1* is *місцевий*) and (*Подія№2* is *об'єктовий*) and (*Подія№3* is *об'єктовий*) and (*Подія№4* is *об'єктовий*) and (*Подія№5* is *об'єктовий*) and *(Небезпечність* is *not небезпечність)* and (*Людність* is *not людність)* and (*Відносний_час* is *not відносний_час)* and *(Абсолютний_час* is *not абсолютний_час)* and (*Площа_ситуації* is *об'єкт*) then (*Рівень_ситуації* is *об'єктовий*).

3. if (*Подія№1* is *об'єктовий*) and (*Подія№2* is *місцевий*) and (*Подія№3* is *об'єктовий*) and (*Подія№4* is *об'єктовий*) and (*Подія№5* is *об'єктовий*) and *Небезпечність* is *not небезпечність)* and (*Людність* is *not людність)* and (*Відносний_час* is *not відносний_час)* and *(Абсолютний_час* is *not абсолютний_час)* and (*Площа_ситуації* is *об'єкт*) then (*Рівень_ситуації* is *об'єктовий*).

.

.

651. if (*Подія№1* is *загальнодержавний*) and (*Подія№2* is *загальнодержавний*) and (*Подія№3* is *загальнодержавний*) and (*Подія№4* is *загальнодержавний*) and (*Подія№5* is *регіональний*) and *Небезпечність* is *небезпечність)* and (*Людність* is *людність)* and

(*Відносний_час* is *відносний_час)* and *(Абсолютний_час* is *абсолютний_час)* and (*Площа_ситуації* is *кілька_регіонів*) then (*Рівень_ситуації* is *загальнодержавний*).
652. if (*Подія№1* is *загальнодержавний*) and (*Подія№2* is *загальнодержавний*) and (*Подія№3* is *загальнодержавний*) and (*Подія№4* is *загальнодержавний*) and (*Подія№5* is *загальнодержавний*) and *Небезпечність* is *небезпечність)* and (*Людність* is *людність)* and (*Відносний_час* is *відносний_час)* and *(Абсолютний_час* is *абсолютний_час)* and (*Площа_ситуації* is *кілька_регіонів*) then (*Рівень_ситуації* is *загальнодержавний*).

Нейрони третього шару обох мереж є звичайними нейронами, які виконують зважене додавання.

Результати роботи нейронних мереж для розпізнавання надзвичайних ситуацій за типом та рівнем.

Результати роботи представлені у вигляді поверхонь відгуку (рис. 8-11).

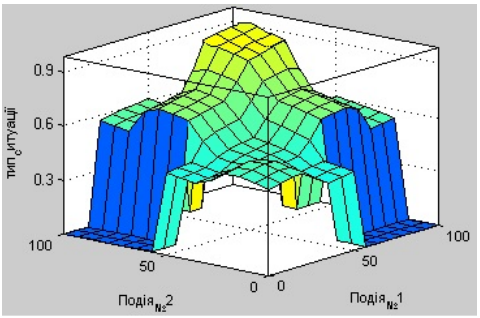


Рис. 8. Поверхня відгуку для виходу «тип ситуації за характером» для вхідних значень «Подія№1» та «Подія№2»

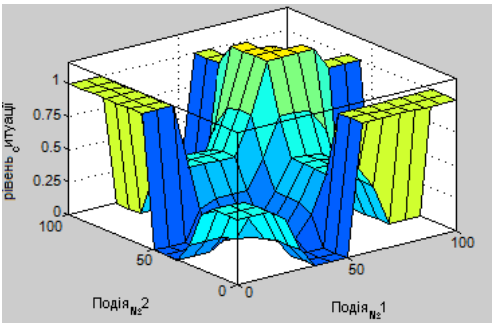


Рис. 9. Поверхня відгуку для виходу «тип ситуації за рівнем» для вхідних значень «Подія№1» та «Подія№2»

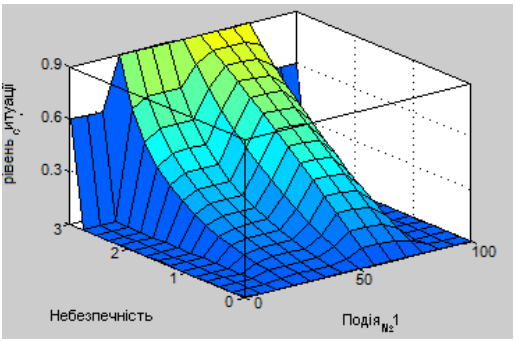


Рис. 10. Поверхня відгуку для виходу «тип ситуації за рівнем» для вхідних значень «Подія№1» та «Небезпечність місця ситуації»

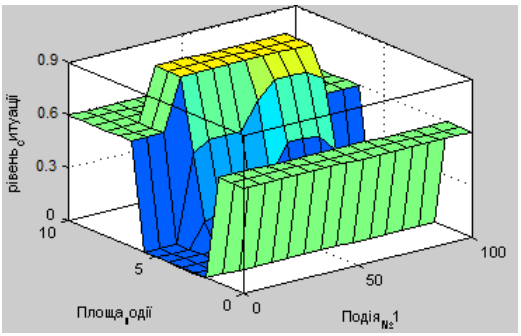


Рис. 11. Поверхня відгуку для виходу «тип ситуації за рівнем» для вхідних значень «Подія№1» та «Площа Події»

Висновки

Використання запропонованої у статті нейронної мережі забезпечує інформаційно-аналітичну підтримку прийняття рішень диспетчеру служби швидкого реагування та дозволяє підвищити якість розпізнавання надзвичайних ситуацій.

Список використаних джерел

1. Круглов В.В. Искусственные нейронные сети. Теория и практика / В.В. Круглов, В.В. Борисов – М.: Горячая линия - Телеком, 2001. – 382 с.

КОНЦЕПТУАЛЬНАЯ МОДЕЛЬ ТЕХНОЛОГИИ РАЗРАБОТКИ ПРИКЛАДНОГО МУЛЬТИПАРАЛЛЕЛЬНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ И УПРАВЛЯЮЩИХ СИСТЕМ

Е.Г. Толстолужская
Харьковский национальный университет имени В.Н. Каразина, Украина
tda_ua@pochtamt.ru

Введение

Современный научно-технический прогресс характеризуется увеличением количества новых перспективных направлений развития науки, техники, производства и управления. Этот процесс сопровождается ростом количества и сложности задач информационных и управляющих систем (ИУС), неотъемлемыми компонентами которых являются данные, техническое и программное обеспечение [1]. Анализ технологий разработки программного обеспечения ИУС выявил следующие принципиальные недостатки [2]:

- концепция выполнения программистом вручную наиболее сложных, неформализованных, творческих этапов разработки параллельных программ;
- недостаточность средств современной дискретной математики для разработки параллельных программ с заданными временными и экономическими характеристиками: временем реализации, интервалом ввода данных, быстродействием/производительностью, достоверностью, сложностью реализации/стоимостью;
- спецификация только статических объектов и отсутствие учета «параметра времени» как одного из существенных факторов повышения эффективности параллельных программ;
- невозможность использования при разработке параллельных программ рациональной совокупности известных методов параллельной обработки данных для различных задач и различных требований и ограничений.

Эти недостатки принципиально ограничивают возможности повышения эффективности известных технологий разработки параллельных программ, снижают их инновационную ценность и обусловливают необходимость обоснования нового класса параллельных программ (времяпараметризованных мультипараллельных программ) и концептуальной модели,

соответствующей информационной технологии параллельного программирования.

Временная параметризация задачи – постановка в соответствие каждому оператору статической модели задачи момента начала его выполнения, длительности реализации и переход от исходной статической модели задачи к времяпараметризованной модели задачи.

Мультипараллельность – использование всех или некоторой части методов параллельной обработки (совмещение операций, конвейерный метод, декомпозиционный метод, метод смеси алгоритмов), определяемого особенностями алгоритмов задач и системой требований и ограничений.

Основные результаты

В основу создания новой технологии положены следующие принципы построения:

- разработка и применение в ИУС нового класса параллельных программ – времяпараметризованных мультипараллельных программ (ВПМП) [3-5];
- применение для разработки ВПМП нового аппарата дискретной математики – пространственно-временных структур семантико-числовой спецификации (СЧС) [3,4];
- мультипараллельность и временная параметризация;
- совместная обработка семантических и числовых характеристик информации;
- поддержка разработки двух основных направлений создания параллельных программ: архитектурно-ориентированных и проблемно-ориентированных ВПМП.

Времяпараметризованная мультипараллельная программа – это конструкция, которая содержит в явном виде спецификации следующих категорий информации:

- множество объектов-данных, над которыми должны выполняться действия;
- множество действий (операций/функций), которые должны быть выполнены над данными для решения задачи;
- множество статических связей, задающих отношения упорядоченности операций/функций по данным и по управлению;
- упорядоченность операций/функций в динамике параллельного вычислительного процесса, задаваемую множеством моментов времени начала выполнения операций/функций;
- разделение множества операций/функций на временные фрагменты (множественные временные операторы, МВО), включающие множество операций/функций, выполнение которых начинается одновременно в конкретный момент дискретного

времени;

- разделение множества данных на фрагменты, поставленные в однозначное соответствие множественным временным операторам и используемые в соответствующие моменты дискретного времени;

- наличие информации о разбиении множества операций/функций различных фрагментов на подмножества (нити), выполняемые соответствующими модулями/процессорами;

- наличие информации о единицах измерения физических величин данных.

Принципиальными отличиями ВПМП от традиционных статических параллельных программ, применяемых в многопроцессорных системах (классов *SMP*, *NUMA*, *MPP, CLUSTER*), являются:

- явное отображение в конструкциях параллельных/ последовательных программ фактора времени как определяющего параметра параллельных процессов;

- мультипараллелизм программ – использование при проектировании ВПМП рационального состава (обеспечивающего повышение эффективности и удовлетворение заданных требований/ограничений) методов параллельной обработки данных;

- учет особенностей конкретных архитектур и конфигураций различных классов параллельных процессоров и ВС (архитектурная ориентация);

- использование принципа «временного управления на каждом такте» в отличие от принципа управления на основе «потока команд» программы и «потока данных» программы, принятых в известных параллельных процессорах и ВС;

- использование в программах единиц измерения физических величин.

Предложена и обоснована концептуальная модель технологии разработки прикладного мультипараллельного программного обеспечения ИУС (рис. 1). В качестве основной группы показателей эффективности мультипараллельных программ выбраны временные показатели (время решения, заданный интервал ввода данных, сокращение временных затрат/ускорение, загрузка оборудования, временные трудозатраты на разработку параллельных программ).

Исходная информация технологии разработки прикладного мультипараллельного программного обеспечения ИУС и поддерживаемые факторы эффективности: последовательная программа частного алгоритма (ЧА) задачи или комплекса

алгоритмов (КА) задач ИУС; классы, архитектуры и конфигурации параллельных вычислительных систем; методы параллельной обработки данных; длительности выполнения операций/функций задач; показатели эффективности и требования/ограничения; проблемно- или архитектурно-ориентированная прикладная направленность разработки ВПМП.

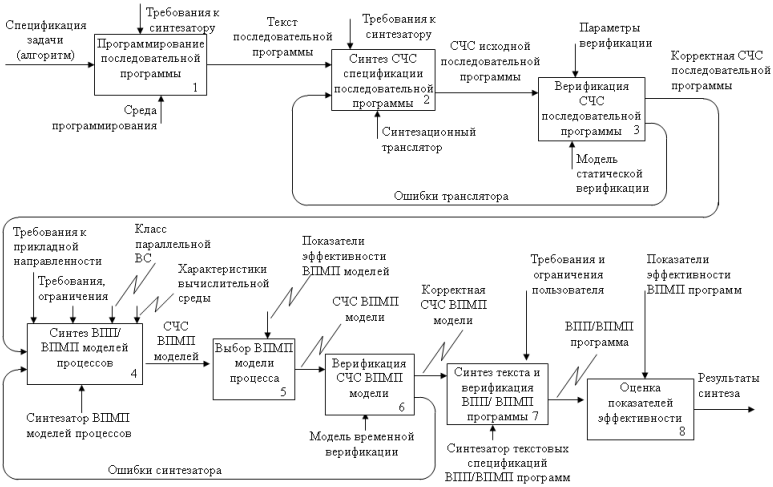


Рис. 1. Концептуальная модель технологии разработки прикладного мультипараллельного программного обеспечения ИУС

Выходная информация: семантико-числовые и текстовые спецификации времяпараметризованных параллельных (ВПП) или мультипараллельных программ; значения показателей эффективности синтезируемых объектов; оценка степени соответствия показателей эффективности системе требований и ограничений оценки достоверности результатов.

Работоспособность и достоверность разработанных моделей и методов технологии показана на примерах реальных задач.

Выводы

1. Существует противоречие между требованиями повышения эффективности параллельного программного обеспечения ИУС и невозможностью их удовлетворения с помощью известных информационных технологий параллельного программирования. Снятие этого противоречия невозможно без обоснования новых классов высокоэффективных параллельных программ и создания новых эффективных технологий их разработки.

2. Разработан новый класс параллельных программ – времяпараметризованные мультипараллельные программы для известных и перспективных параллельных ИУС. Их основными отличиями от существующих параллельных программ являются: учет в явном виде времени как существенного фактора повышения эффективности параллельных программ; обеспечение возможности использования различных методов параллельной обработки данных; учет особенностей принципов построения и архитектур различных параллельных ВС; учет требований и ограничений различных прикладных областей, что позволяет синтезировать параллельные высокоэффективные программы.

3. Создана концептуальная модель технологии разработки прикладного мультипараллельного программного обеспечения ИУС.

Список использованных источников

1. ISO/IEC 2382-1:1993. Information technology – Vocabulary – Part 1: Fundamental terms.
2. Воеводин В. В. Параллельные вычисления / В. В. Воеводин, Вл. В. Воеводин. – СПб. : БХВ-Петербург, 2002. – 608 с.
3. Синтез и анализ параллельных процессов в адаптивных времяпараметризованных вычислительных системах : монография / Г. А. Поляков, С. И. Шматков, Е. Г. Толстолужская, Д. А. Толстолужский ; – Х.: ХНУ им. В.Н. Каразина, 2012. – 670 с.
4. Поляков Г. А. Технология проектирования время-параметризованных мультипараллельных программ как стратегия развития систем параллельного проектирования / Г. А. Поляков, Е. Г. Толстолужская // Радіоелектронні і комп'ютерні системи – Х.: Національний аерокосмічний університет ім. М.Є. Жуковського «Харківський авіаційний інститут», 2009. – Вип. 6(40). – С. 166-171.
5. Tolstoluzka H. G. Synthesis of time parameterized parallel programs for computer systems of class of MPP / H. G. Tolstoluzka // Системи обробки інформації. - 2013. - Вип. 9. - С. 173-180.

THIRD DECADE DEVELOPMENTS OF C++ PROGRAMMING LANGUAGE

O. I. Chentsov
Taras Shevchenko National University of Kyiv, Ukraine
chentsov@ukr.net

Programming language is a crucial tool of software development process. Important aspects that characterize the language are efficiency, ability to create new abstractions, clearness of expression that implies readability and simplicity of modification. The relative importance of these factors changed with the advances of computing resources. C++ is the language that from the start aimed providing abstraction mechanisms for systems programming [1] where efficiency is the first priority. Due to this it has critical constraint that its every construct directly maps to hardware facilities [1,2]. Contributors to the language avoided conservative decisions that favored better structuring over the flexibility. As Stroustrup put it efficiency and flexibility have been maintained without compromise [2]. This makes C++ untidy sometimes and harder to grasp but practitioners worked out specific idioms that addressed this problem. Overall C++ has following features: compilation to machine code, dozens of free and proprietary compilers (rivaled only by C), support for multiple styles of programming, additional upper level that serves compile-time computations and programmer-guided optimization, its evolution and standardization is closely watched and directed by hundreds of C++ practitioners and key figures in software industry. All this combined distinguish C++ from rivals, and makes its evolution a unique and valuable experience in the software history by itself. As of today C++ supports following programming styles: procedural programming, object oriented programming, generic programming[3-6], (template) metaprogramming, functional programming (to some extent). Though far from perfect C++ is the leading tool in research of generic programming[4,5]. This and template metaprogramming makes C++ the implementation language of the first choice for the highly efficient interoperable reusable components, e.g., Boost-libraries.

This work summarizes what C++ language has to offer for software developer based on the refinements of the language in C++11 and C++14 standards. As usual changes are divided into categories: core language features and evolution of standard library. This includes facilities in compile-time computations, automatic type reconstruction, variadic templates, support for multithreading, lambda-expressions, bindings, function objects and other elements of functional programming, concept checking, traits, and special functions on types.

Where appropriate changes are illustrated with code samples. Special stress is made on features facilitating generic programming like concepts that though first rejected as raw gradually find their way to the algorithm specifications[7] and language syntax[8]. Also this work outlines possible further evolution of the language and implications for its community.

References

1. B. Stroustrup: The Design and Evolution of C++ / B. Stroustrup. – Addison-Wesley Longman, 1994. – 480 pp.
2. Stroustrup B. Evolving a Language in and for the Real World: C++ 1991-2006 / Bjarne Stroustrup // Proceedings of the Third ACM SIGPLAN Conference on History of Programming Languages (HOPL III, San Diego, California, 9-10 June 2007). – New York: ACM, 2007. – Pp. 4-1– 4-59.
3. Generic Programming / [Mehdi Jazayeri, R. Loos, David Musser, and Alexander Stepanov] // Report of the Dagstuhl Seminar on Generic Programming. – Dagstuhl Schloss, Germany, 1998. – http://www.cs.rpi.edu/~musser/gp/dagstuhl/gpdag.ps.
4. Siek J. A language for generic programming: Ph.D. Thesis / J. Siek; Indiana University. – 2005. — 260 pp.
5. An extended comparative study of language support for generic programming / R. Garcia, J. Jaarvi, A. Lumsdaine et al. // Journal of Functional Programming. – 2005. – Pp. 115–134.
6. Ченцов А. Параметризованное программирование: современное состояние и перспективы / А. Ченцов // Тези доп. міжн. конф. "Теоретичні та прикладні аспекти побудови програмних систем" (КНУ, 8-10 грудня 2009). – Київ, 2009. – С.159–164/
7. Stepanov A. Elements of programming / Alexander Stepanov, Paul McJones. – Addison-Wesley, 2009 – 288 pp.
8. Stroustrup B. A Concept Design for the STL. / B. Stroustrup and A. Sutton // Technical Report N3351=12-0041, ISO/IEC JTC1/SC22/WG21. – The C++ Standards Committee, Jan. 2012. – 133 pp.

ТРЕБОВАНИЯ И МЕХАНИЗМ ЗАЩИТЫ АУТЕНТИФИКАЦИИ WEB – САЙТОВ

Д.А. Черкасов
Харьковский национальный университет имени В.Н. Каразина, Украина
dima.scorpio.cherkasov@gmail.com

Введение

В настоящее время в Европейском Союзе ведутся интенсивные разработки, которые связаны с предоставление трансграничных и электронно-доверительных услуг. Одной из таких базовых услуг, является услуга электронной аутентификации web-сайта.

Под аутентификацией понимается электронный процесс, который позволяет подтвердить принадлежность идентификационных данных физическому или юридическому лицу, информационной системе, а также происхождение и целостность электронных данных во время доступа к ним в информационной системе (ИС).

На сегодняшний день очень большое количество компаний использует сайты в сети Интернет для своих целей. Владельцы сайтов могут размещать там конфиденциальную информацию, доступ к которой ограничен. Очень часто, для авторизации используют “логин” и “пароль”. В наше время уже создано множество вирусов, которые могут узнать ваши данные, а так же существует различные способы утечки информации.

Таким образом, нарушитель может получить доступ к конфиденциальной информации и использовать ее в своих целях. Поэтому, в наше время существует различные угрозы аутентификации.

Целью работы является представление возможных угроз для аутентификации web-сайта, а так же механизм защиты web-сайта.

Модель угроз

Если информационная система персональных данных [1] (ИСПД) реализована на базе локальной или распределенной информационной системы, то в ней могут быть реализованы угрозы безопасности информации путем использования протоколов межсетевого взаимодействия. При этом может обеспечиваться несанкционированный доступ (НСД) к персональным данным (ПД) или реализовываться угроза отказа в обслуживания. Особенно опасны угрозы, когда ИСПД представляет собой распределенную информационную систему, подключенную к сетям общего

пользования и (или) сетям международного информационного обмена.

В ее основу положено семь следующих первичных признаков классификации (рисунок.1).

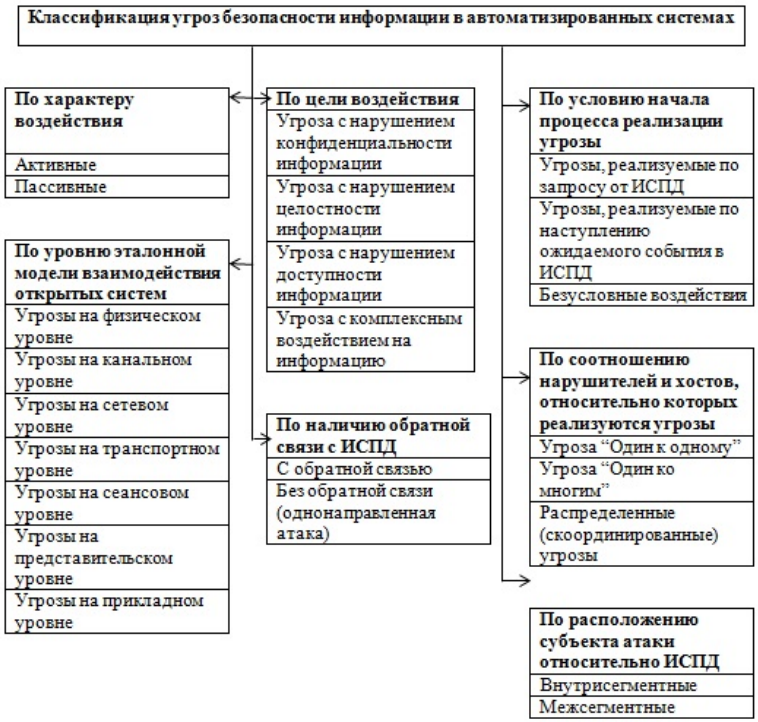


Рис. 1. Классификация угроз с использованием протоколов межсетевого взаимодействия

1. Характер угрозы. По этому признаку угрозы могут быть пассивные и активные.

Пассивная угроза – это угроза, при реализации которой не оказывается непосредственное влияние на работу ИСПД, но могут быть нарушены установленные правила разграничения доступа к ПД или сетевым ресурсам.

Активная угроза – это угроза, связанная с воздействием на ресурсы ИСПД, при реализации которой оказывается непосредственное влияние на работу системы (изменение конфигурации, нарушение работоспособности и т.д.), и с нарушением установленных правил разграничения доступа к ПД или сетевым ресурсам.

2. Цель реализации угрозы. По этому признаку угрозы могут быть направлены на нарушение конфиденциальности, целостности и доступности информации.

3. Условие начала осуществления процесса реализации угрозы. По этому признаку может реализовываться угроза:

- по запросу от объекта, относительно которого реализуется угроза;
- по наступлению ожидаемого события на объекте, относительно которого реализуется угроза;
- безусловное воздействие.

4. Наличие обратной связи с ИСПД. По этому признаку процесс реализации угрозы может быть с обратной связью и без обратной связи.

5. Расположение нарушителя относительно ИСПД. В соответствии с этим признаком угроза реализуется как внутрисегментно, так и межсегментно.

6. Уровень эталонной модели взаимодействия открытых систем (ISO/OSI), на котором реализуется угроза.

7. Соотношение количества нарушителей и элементов ИСПД, относительно которых реализуется угроза.

Методы защиты web-сайтов

Web-сайты [2, 43-45с.] в современном Интернете занимают очень важную роль, так как их основная функция заключается в представлении интересов личности, компании, организации, государства в Мировой сети. На web-сайте может размещаться информация с публичным доступом либо с ограниченным доступом.

Web-сайты с публичным доступом используют, например, для представлении фирм, компаний или для распространения общедоступной информации.

Информация с ограниченным доступом включает в себя конфиденциальную информацию, доступную ограниченному кругу лиц - например, предоставление сотрудникам организации удобного доступа к внутренним корпоративным информационным ресурсам.

Так как к web-сайту можно подключиться с любой точки мира, где есть выход в Интернет, то появляется множество угроз нарушения конфиденциальной информации на web-сайте. Угрозы нарушения конфиденциальности возможны в двух случаях:

- угрозы несанкционированного доступа к информации, хранимой на web-сайте локально;
- угрозы несанкционированного доступа к информации в процессе ее передачи по каналам связи.

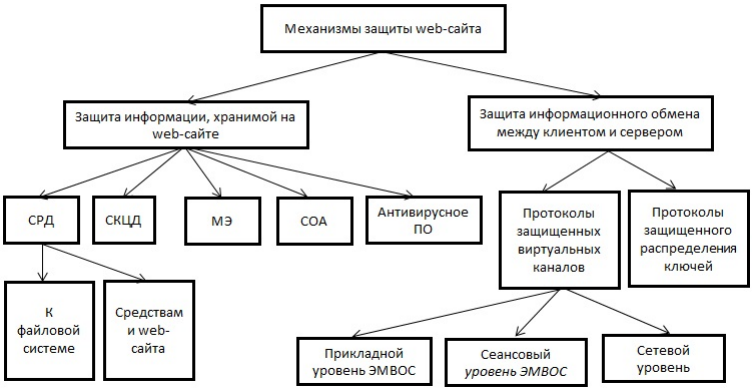


Рис. 2. Классификация механизмов защиты web-сайтов

СРД- система разграничения доступа;
СКЦД – система контроля целостности данных;
МЭ – межсетевой экран;
СОА – межсетевой экран;
ЭМВОС- модель взаимодействия открытых систем.
Собственно механизмы защиты веб-сервера препятствуют получению какой-либо дополнительной информации, способствующей злоумышленнику в преодолении этой системы защиты.

Протокол SSL/TLS

Целями протокола SSL/TLS в порядке приоритетности являются:

- Криптографическая безопасность. SSL/TLS должен использоваться для установления безопасного соединения между двумя партнерами;

- Совместимость. Независимые программисты должны быть способны разрабатывать приложения, использующие SSL/TLS, которые будут способны успешно обмениваться криптографическими параметрами без знания особенностей программ друг друга;

- Расширяемость. SSL/TLS ищет способ, как при необходимости встроить в систему новые ключи и методы шифрования. Здесь имеются две побочные цели: исключить необходимость создания нового протокола (что может быть

сопряжено с введением новых слабых мест) и сделать ненужным внедрение новой библиотеки, обеспечивающей безопасность;

- Относительная эффективность. Криптографические операции требуют больших мощностей ЦПУ, особенно этим славятся операции с открытыми ключами.

SSL/TLS противостоят следующим угрозам:

- подмена идентификатора клиента или сервера (с помощью надежной аутентификации, функциональные требования безопасности FIA_U1D (Идентификация пользователя) и FIA_UAU (Аутентификация пользователя) согласно ГОСТ ИСО/МЭК 15408-2);

- раскрытие информации (с помощью шифрования канала связи, функциональное требование безопасности FDPUCT (Защита конфиденциальности данных пользователя при передаче между функциями безопасности объекта оценки (ФБО));

- модификации данных (с помощью кодов целостности сообщений, функциональное требование безопасности FDPU1T (Защита целостности данных пользователя при передаче между ФБО))

Выводы

Таким образом, в наше время существуют угрозы аутентификации web-сайтов, так как без них не обходятся многие пользователи сети Интернет. Была рассмотрена возможная угроза с использованием протоколов межсетевого взаимодействия и ее классификация. Также были представлены возможные методы защиты, включая протокол SSL/TLS. В работе представлены основные цели данного протокола и достоинства.

Список используемых источников

1. Базовая модель угроз безопасности персональных данных при их обработке в информационных системах персональных данных/за состоянием на 15 февраля 2008/ Федеральная служба по техническому и экспортному контролю. офиц. изд. – К: Парлам. изд-во, 2009. – 69 стр.

2. Биячуев Т.А. Модель и методы мониторинга и оценки защищенности веб-сайтов сети Интернет: автореф. дис. на получения научной степени канд. техн. наук: спец. 05.13.19 «Методы и системы защиты информации, информационная безопасность» / Т.А Биячуев.; Санкт-Петербургский государственный университет информационных технологий, механики и оптики – Санкт-Петербург, 2004. – 20 с.

ВЫЧИСЛЕНИЕ КОЭФФИЦИЕНТОВ ПРИНАДЛЕЖНОСТИ ПРИ КЛАССИФИКАЦИИ С ИСПОЛЬЗОВАНИЕМ ВЕКТОРНОГО КРИТЕРИЯ РАСПОЗНАВАНИЯ ОБЪЕКТОВ

Ю.П. Зайченко[1], П.В. Четырбок[2]
[1]ИПСА НТУУ «Киевский политехнический университет», Украина
[2]Институт экономики и управления Крымского гуманитарного университета, Ялта
petr58@ukr.net

Введение

При нахождении критериев близости распознавания объектов нейронной сетью обычно рассматривается следующая задача: задан набор объектов, каждому объекту сопоставлен вектор значений признаков; требуется разбить эти объекты на классы эквивалентности.

Для каждого нового объекта мы должны выполнить две операции:

1) найти класс, к которому он принадлежит;

2) использовать новую информацию, полученную об этом объекте, для исправления (коррекции) правил классификации.

Отнесение объекта к классу проводится путем его сравнения с типичными элементами разных классов и выбора элемента, наиболее близкого к исследуемому объекту. Простейшая мера близости объектов – квадрат евклидового расстояния между векторами значений их признаков: чем меньше расстояние – тем ближе объекты. Соответствующее определение признаков типичного объекта – среднее арифметическое значение признаков по выборке, представляющей класс.

Другая мера близости, возникающая при обработке сигналов и изображений – квадрат коэффициента корреляции: чем он больше, тем ближе объекты. Возможны и иные варианты мер близости, некоторые из них рассмотрены в настоящей работе.

В докладе вводятся меры близости между входными и эталонными образами типичных представителей классов, при использовании разной размерности векторного критерия. В качестве меры берется коэффициент принадлежности объекта к классу. В одномерном случае в качестве меры выступает Евлидово расстояние, для размерности 2 и более коэффициент принадлежности, вычисленный по формуле:

$$FP_i = (1 - \frac{(1 - \boldsymbol{COS}(\ \lambda_{\boldsymbol{i}}))}{\sum_{j=1}^{m}(1 - COS\ (\lambda_j))}) /(m - 1)\,,$$ где i – номер класса, m – количество классов.

В работе рассматривается задача вычисления коэффициентов принадлежности объектов к классам с использованием векторного критерия распознавания объектов нейронной сетью.

Формулировка векторного критерия близости образов в пространстве ошибок

Предложен метод перехода из пространства параметров объектов в пространство ошибок их распознавания с помощью нейронной сети. Для эталонных пар (объект, образ объекта) вычисляются значения каждой координаты вектора ошибок $E = (E_1, E_2, E_3)$, в котором E_1 – среднеквадратическая ошибка, E_2 – линейная ошибка сети, E_3 – максимальная ошибка поразрядного отклонения образа от эталона по таким формулам:

$$E_1 = \frac{1}{n}\sqrt{\sum_{j=1}^{n}(y^1_j - d_j)^2}\,,$$

где y^1_j – реальное выходное состояние нейрона j выходного слоя НС при подаче на ее входы образа, соответствующее первой координате, d_j – идеальное (желаемое) выходное состояние этого нейрона, а суммирование ведется по всем нейронам выходного слоя n;

$$E_2 = \frac{1}{n}\sum_{j=1}^{n}\left|y^2_j - d_j\right|\,,$$

где y^2_j – реальное выходное состояние нейрона j выходного слоя НС при подаче на ее входы образа, соответствующее второй координате, d_j – идеальное (желаемое) выходное состояние этого нейрона, и суммирование ведется по всем нейронам выходного слоя n;

$$E_3 = \max_{j=1,\ldots,n} \left| y_j^3 - d_j \right|,$$

где y_j^3 – реальное выходное состояние нейрона j выходного слоя НС при подаче на ее входы образа, d_j – идеальное (желаемое) выходное состояние этого нейрона.

Вычисление коэффициентов принадлежности

Обучение нейронной сети осуществляется на выборке, состоящей из набора эталонных объектов, с помощью алгоритма обратного распространения ошибки.

В результате обучения нейронной сети получим m векторов ошибок. Для каждого распознаваемого неизвестного объекта находим вектор ошибок $X = (X_1, X_2, X_3)$, каждая компонента которого вычисляется по формулам, аналогичным формулам, вычисляющим координаты вектора E. Затем для нахождения векторного критерия произвольного объекта используем следующий алгоритм:

1) Подаем на вход нейронной сети параметры объекта.

2) На выходе нейронной сети сравниваем значение всех нейронов выходного слоя с первым эталонным образом и вычисляем вектор ошибок.

3) Находим скалярное произведения вектора ошибок распознаваемого объекта и вектора ошибок первого эталонного объекта.

4) Шаги 1-3 повторяем для всех m эталонных объектов. В результате получим m значений скалярных произведений.

5) Функция F равна скалярному произведению вектора ошибок эталона $E = (E_1, E_2, E_3)$ и вектора ошибок исследуемого объекта $X = (X_1, X_2, X_3)$ $F = (\vec{E}, \vec{X}) = \cos(\lambda) \left\| \vec{E} \right\| \left\| \vec{X} \right\|$.

Найдем косинус угла между векторами ошибок:

$$\cos(\lambda) = \frac{(\overline{E}, \overline{X})}{\left\| \overline{E} \right\| \left\| \overline{X} \right\|}.$$

Аналогично находим косинусы для всех $F_1, F_2, \ldots, F_m$ и находим максимум $\cos(\lambda)$. После этого находим

соответствующую этому максимальному значению функцию F_i, где $i = 1,\ldots,m$, которая и определяет эталонный образ соответствующий входному сигналу. Соответственно эталонный образ определяет класс, к которому принадлежит входной сигнал.

Для нахождения класса, к которому принадлежит входной сигнал, вычислим коэффициенты принадлежности FP_i сигнала ко всем классам. Класс, на котором достигается максимум коэффициента принадлежности, является искомым.

$$FP_i = (1 - \frac{(1 - \mathrm{COS}(\lambda_i))}{\sum_{j=1}^{m}(1 - \mathrm{COS}(\lambda_j))}) / (m - 1)$$

где i – номер класса, m – количество классов.

Выводы

В качестве меры близости между входными и эталонными образами типичных представителей классов, при использовании разной размерности векторного критерия, берется коэффициент принадлежности объекта к классу (1). В докладе показано, что с увеличением размерности векторного критерия качество распознавания объектов увеличивается.

Список использованных источников

1. Зайченко Ю.П. Нечеткие модели и методы в интеллектуальных системах. Учебное пособие для студентов высших учебных заведений / Ю.П. Зайченко. – Киев: Слово, 2008. – 344 с.
2. Згуровський М.З. Основы вычислительного интеллекта / М.З. Згуровський, Ю.П. Зайченко. – Киев: Наукова думка, 2013. – 408 с.
3. Ивахненко А.Г. Самоорганизующиеся системы распознавания и автоматического управления / А.Г. Ивахненко. – Киев: Техніка, 1969. – 216 с.
4. Колмогоров А.Н. О представлении непрерывных функций нескольких переменных суперпозициями функций меньшего числа переменных / А.Н. Колмогоров // Докл. АН СССР. – М., 1957. – С. 179 – 182.

КОМПОЗИЦІЙНО-НОМІНАТИВНІ МОДАЛЬНІ ЛОГІКИ ЧАСТКОВИХ ПРЕДИКАТІВ БЕЗ ОБМЕЖЕННЯ МОНОТОННОСТІ

О.С. Шкільняк
Київський національний університет імені Тараса Шевченка,
Україна
me.oksana@gmail.com

Вступ

Розвиток інформаційних технологій зумовлює розширення сфери застосування математичної логіки. Особливе місце серед логічних систем посідають модальні логіки. Для опису інформаційних та експертних систем, баз даних і баз знань з неповною інформацією використовуються епістемічні логіки. Темпоральні логіки застосовуються для моделювання динамічних систем, специфікації та верифікації програм. Можливості традиційних модальних логік і композиційно-номінативних логік квазіарних предикатів [1] поєднують композиційно-номінативні модальні логіки (КНМЛ). Важливим класом КНМЛ є транзиційні модальні логіки (ТМЛ), вони відбивають аспект зміни й розвитку предметних областей. В межах ТМЛ природним чином можуть розглядатися традиційні модальні логіки. Різноманітні класи ТМЛ еквітонних (монотонних) предикатів запропоновано та досліджено, зокрема, в роботах [2, 3], для таких логік збудовано числення секвенційного типу.

В даній роботі запропоновано нові класи КНМЛ – ТМЛ часткових квазіарних предикатів, не обмежених умовою монотонності. Підкласами ТМЛ є мультимодальні (ММЛ) та темпоральні (ТмМЛ) логіки, в межах ММЛ виділено ТМЛ епістемічного типу та загальні ТМЛ. Описано семантичні моделі та мови чистих першопорядкових ТМЛ зазначених класів, розглянуто їх семантичні властивості. Показана відмінність ТМЛ немонотонних предикатів від ТМЛ еквітонних предикатів. Наведено властивості відношень логічного наслідку для множин специфікованих станами формул.

Поняття, які тут не визначено, тлумачимо в сенсі робіт [1–3].

Композиційно-номінативні модальні системи

Основним семантичним поняттям КНМЛ є поняття композиційно-номінативної модальної системи (КНМС).

КНМС – це об'єкт вигляду $\boldsymbol{M} = (Cms, Fm, Im)$. Тут Cms – композиційна модальна система (КМС), вона задає семантичні аспекти світу; Fm – множина формул відповідної мови КНМЛ; Im – відображення інтерпретації формул на станах світу.

КМС є семантичними моделями реляційного типу. Вони ма-

ють вигляд $Cms = (S, R, Pr, C)$, де S – множина станів світу, R – множина відношень на S вигляду $\rho \subseteq S \times S^n$, Pr – множина предикатів на станах світу, C – множина композицій на Pr.

Уточнимо поняття КМС для чистих першопорядкових КНМЛ. Для них S – це множина алгебраїчних систем вигляду $\alpha = (A_\alpha, Pr_\alpha)$. Тут Pr_α – множина квазіарних предикатів вигляду ${}^V A_\alpha \to \{T, F\}$, їх називаємо предикатами стану α. Предикати вигляду ${}^V A \to \{T, F\}$, де $A = \bigcup\limits_{\alpha \in S} A_\alpha$, назвемо глобальними.

Нагадаємо [1], що V-квазіарний предикат на множині D – це функція вигляду ${}^V D \to \{T, F\}$, де ${}^V D$ – множина всіх V-іменних множин (V-ІМ) над D, $\{T, F\}$ – множина істиннісних значень. V-ІМ над D – це однозначна функція вигляду $d : V \to D$. Подаємо V-ІМ як $[v_1 \mapsto a_1, ..., v_n \mapsto a_n, ...]$, де $v_i \in V$, $a_i \in D$. Для V-ІМ задаємо операції вилучення компоненти $\|_{-x}$, накладки ∇, реномінації $\mathrm{r}_{\bar{x}}^{\bar{v}}$.

Для чистих першопорядкових КНМЛ множина композицій задається базовими загальнологічними композиціями кванторного рівня $\neg, \vee, \mathrm{R}_{\bar{x}}^{\bar{v}}, \exists x$ та базовими модальними композиціями.

Визначення композицій $\neg, \vee, \mathrm{R}_{\bar{x}}^{\bar{v}}, \exists x$ наведено в [1].

Області істинності та хибності предиката $P : {}^V A \to \{T, F\}$ визначаються так: $T(P) = \{d \in {}^V A \mid P(d) = T\}$, $F(P) = \{d \in {}^V A \mid P(d) = F\}$.

Предикат $P : {}^V A \to \{T, F\}$ (частково) істинний, або неспростовний, якщо $F(P) = \varnothing$.

Ім'я $x \in V$ (строго) неістотне для квазіарного предиката P, якщо для довільних $d_1, d_2 \in {}^V A$ таких, що $d_1\|_{-x} = d_2\|_{-x}$, маємо $P(d_1) = P(d_2)$.

Предикат P еквітонний (монотонний), якщо:

$P(d)\downarrow$ та $d \subseteq d' \Rightarrow P(d')\downarrow = P(d)$.

Опишемо мову чистих першопорядкових КНМС. Алфавіт мови: множина V предметних імен; множина Ps предикатних символів (сигнатура мови); символи базових композицій $\neg, \vee, R_{\bar{x}}^{\bar{v}}, \exists x$; множина Ms символів базових модальних композицій (модальна сигнатура). Множина Fm формул мови визначається індуктивно.

Маємо $Ps \subseteq Fm$; а далі:

– $\Phi, \Psi \in Fm \Rightarrow \neg\Phi, \vee\Phi\Psi, R_{\bar{x}}^{\bar{v}}\Phi, \exists x\Phi \in Fm$;

– $\Phi \in Fm$ та $\mathfrak{M} \in Ms \Rightarrow \mathfrak{M}\Phi \in Fm$.

Tun КНМС задається її модальною сигнатурою Ms, однотипністю відношень із R для кожного $\mathfrak{M} \in Ms$ та сигнатурою синтетичної неістотності [2].

Задамо відображення інтерпретації атомарних формул на станах $Im : Ps \times S \to Pr$. При цьому постулюємо умову $Im(p, \alpha) \in Pr_\alpha$.

Це означає: базові предикати є предикатами станів. Продовжимо *Im* до відображення $Im : Fm \times S \to Pr$ інтерпретації формул на станах:

$Im(\neg, \alpha) = \neg(Im(\Phi, \alpha))$;

$Im(\vee\Phi\Psi, \alpha) = \vee(Im(\Phi, \alpha), Im(\Psi, \alpha))$;

$Im(R_{\bar{x}}^{\bar{v}}\Phi, \alpha) = \mathrm{R}_{\bar{x}}^{\bar{v}}(Im(\Phi, \alpha))$;

для кожних $d \in {}^{V}A$ задаємо

$$Im(\exists x\Phi, \alpha)(d) = \begin{cases} T, \text{ якщо існує } a \in A_{\alpha}: Im(\Phi, \alpha)(d\nabla x \mapsto a) = T, \\ F, \text{ якщо } Im(\Phi, \alpha)(d\nabla x \mapsto a) = F \text{ для всіх } a \in A_{\alpha}, \\ \text{невизначене в усіх інших випадках.} \end{cases}$$

Предикати, які є значеннями немодалізованих формул (такі формули не містять символи із *Ms*), належать до предикатів станів.

Визначення *Im* для формул вигляду ⌘Φ конкретизуємо залежно від різновидності КНМС.

Предикат $Im(\Phi, \alpha)$ – значення Φ у стані α – позначаємо Φ_{α}.

Формула Φ *істинна в стані* α (позн. $\alpha \models \Phi$), якщо Φ_{α} – істинний предикат. Формула Φ *істинна в КНМС* ***M*** (позн. $\boldsymbol{M} \models \Phi$), якщо для кожного $\alpha \in S$ предикат Φ_{α} є істинним.

Формула Φ *усюди істинна* (позн. $\models \Phi$), якщо $\boldsymbol{M} \models \Phi$ для всіх КНМС ***M*** одного типу.

Транзиційні модальні системи

Транзиційні модальні системи (ТМС) – це КНМС, у яких *R* складається з відношень вигляду $\mathrm{R} \subseteq S \times S$. Трактуємо їх як відношення переходу (досяжності) на станах.

ТМС із множиною відношень $R = \{\triangleright_i \mid i \in I\}$ та базовими модальними композиціями K_i, $i \in I$, у яких кожному $\triangleright_i \in R$ зіставлено відповідну K_i, назвемо *мультимодальними* (ММС).

Загальні ТМС є окремим випадком ММС, для них *R* складається з єдиного відношення ▷ та маємо єдину базову модальну композицію □ (необхідно).

ТМС, у яких *R* складається з єдиного бінарного відношення, а базовими модальними композиціями є $\Box_{\uparrow}$ (завжди буде) та $\Box_{\downarrow}$ (завжди було), називають *темпоральними* (ТмМС).

У випадку ММС для формул вигляду $\mathrm{K}_i\Phi$, де $\mathrm{K}_i \in Ms$, для кожних $\alpha \in S$ та $d \in {}^{V}A$ задаємо

$$Im(K_i\Phi, \alpha)(d) = \begin{cases} T, \text{ якщо } Im(\Phi, \delta)(d) = T \text{ для всіх } \delta \in S: \alpha \triangleright_i \delta, \\ F, \text{ якщо існує } \delta \in S: \alpha \triangleright_i \delta \text{ та } Im(\Phi, \delta)(d) = F, \\ \text{невизначене в усіх інших випадках.} \end{cases}$$

Якщо для $\alpha \in S$ не існує такого β, що $\alpha \triangleright_i \beta$, то вважаємо $Im(K_i\Phi, \alpha)(d)\uparrow$ для кожного $d \in {}^{V}A$.

У випадку ТмМС для формул вигляду $\Box_{\uparrow}\Phi$ та $\Box_{\downarrow}\Phi$ для кожних $\alpha \in S$ та $d \in {}^{V}A$ задаємо:

$$Im(\Box_{\uparrow}\Phi, \alpha)(d) = \begin{cases} T, \text{ якщо } Im(\Phi, \delta)(d) = T \text{ для всіх } \delta \in S : \alpha \triangleright \delta, \\ F, \text{ якщо існує } \delta \in S : \alpha \triangleright \delta \text{ та } Im(\Phi, \delta)(d) = F, \\ \text{невизначене в усіх інших випадках.} \end{cases}$$

$$Im(\Box_{\downarrow}\Phi, \alpha)(d) = \begin{cases} T, \text{ якщо } Im(\Phi, \delta)(d) = T \text{ для всіх } \delta \in S : \delta \triangleright \alpha, \\ F, \text{ якщо існує } \delta \in S : \delta \triangleright \alpha \text{ та } Im(\Phi, \delta)(d) = F, \\ \text{невизначене в усіх інших випадках.} \end{cases}$$

Якщо для $\alpha \in S$ не існує такого β, що $\alpha \triangleright \beta$, то для кожного $d \in {}^{V}A$ вважаємо $Im(\Box_{\uparrow}\Phi, \alpha)(d)\uparrow$. Якщо для $\alpha \in S$ не існує такого β, що $\beta \triangleright \alpha$, то для кожного $d \in {}^{V}A$ вважаємо $Im(\Box_{\downarrow}\Phi, \alpha)(d)\uparrow$.

Залежно від умов, накладених на відношення переходу, можна визначати різні класи ТМС. Традиційно ці відношення розглядають як рефлексивні, симетричні чи транзитивні.

Для ММС розглянемо випадки, коли всі відношення переходу однотипні (рефлексивні, симетричні чи транзитивні). Якщо вони рефлексивні, то в назві ММС пишемо *R*; якщо транзитивні, то пишемо *T*; якщо симетричні, то пишемо *S*. Маємо чисті типи ММС:

R-ММС, *T*-ММС, *S*-ММС, *RT*-ММС,
RS-ММС, *TS*-ММС, *RTS*-ММС.

ММС із скінченними множинами однотипних відношень переходу назвемо *епістемічними*, або ММС епістемічного типу. Загальні ТМС можна трактувати як окремий випадок епістемічних.

Розглядаючи для ТмМС випадки, коли $\triangleright$ може бути рефлексивним, симетричним чи транзитивним, маємо такі їх класи:

R-ТмМС, *T*-ТмМС, *S*-ТмМС, *RT*-ТмМС,
RS-ТмМС, *TS*-ТмМС, *RTS*-ТмМС.

Розглянемо властивості транзиційних модальних систем.

При умові $d \notin {}^{V}A_{\delta}$ постає питання: як задати значення $\Phi_{\delta}(d)$? Залежно від відповіді на нього для логік еквітонних предикатів виділено [2] ТМС із сильною та із загальною умовами визначеності на станах. Для загального випадку КНМЛ квазіарних предикатів ми виділяємо предикати станів та глобальні предикати. Предикати, які є значеннями немодалізованих формул, належать до предикатів станів. Вони задаються так: для $d \in {}^{V}A$ вважаємо $\Phi_{\delta}(d) = \Phi_{\delta}(d_{\delta})$, де d_{δ} позначає ІМ $[v \mapsto a \in d \mid a \in A_{\delta}]$. Неформально це означає, що предикати стану δ "відчувають" лише компоненти вигляду $v \mapsto a$ із $a \in A_{\delta}$.

Модальні композиції можна проносити через реномінації.

Теорема 1. Маємо $R^{\bar{v}}_{\bar{x}} \mathbin{⌘} \Phi(d) = \mathbin{⌘} R^{\bar{v}}_{\bar{x}} \Phi(d)$ для кожних $⌘ \in Ms$, $\Phi \in Fm$, $d \in {}^{V}A$.

Тут і надалі вважаємо: $Ms=\{\Box\}$ для загальних ТМС, $Ms=\{\Box_\uparrow, \Box_\downarrow\}$ для ТмМС, $Ms=\{K_i \mid i\in I\}$ для ММС.

Наслідок 1. Кожна формули вигляду $R_{\bar{x}}^{\bar{v}}\, \text{⌘}\, \Phi \leftrightarrow \text{⌘}\, R_{\bar{x}}^{\bar{v}}\, \Phi$, де ⌘ $\in Ms$, всюди істинна.

Розглянемо взаємодію в ТМС модальних композицій та кванторів. Для ТМЛ еквітонних предикатів формули $\text{⌘}\forall x\Phi \to \forall x\text{⌘}\Phi$ та $\exists x\text{⌘}\Phi \to \text{⌘}\exists x\Phi$ всюди істинні [2, 3]. Це вже не так для загального випадку ТМЛ немонотонних (нееквітонних) предикатів.

Приклад 1. Формули $\Box\forall x\Phi\to\forall x\Box\Phi$ та $\exists x\Box\Phi\to\Box\exists x\Phi$ не є всюди істинними.

Побудуємо загальну ТМС, в якій спростовується формула вигляду $\Box\forall x\Phi\to\forall x\Box\Phi$. Нехай $S=\{\alpha, \beta\}$, $R=\{\alpha\triangleright\beta\}$, $A_\alpha=\{a\}$, $A_\beta=\{b\}$. Візьмемо $p\in Ps$, для якого неістотні усі імена, окрім x. Задамо $p_\alpha(\varnothing)=F$, $p_\beta(\varnothing)=F$, $p_\beta([x\mapsto b])=T$. Тоді маємо $(\forall x\, p)_\beta(\varnothing)=T$, звідки $(\Box\forall x\, p)_\alpha(\varnothing)=T$. Водночас $(\Box p)_\alpha([x\mapsto a])=F$, адже в силу $[x\mapsto a]_\beta=\varnothing$ маємо $p_\beta([x\mapsto a]_\beta)=p_\beta(\varnothing)=F$. Звідси $(\forall x\Box p)_\alpha(\varnothing)=F$. Отже, $(\Box\forall x\, p\to\forall x\Box p)_\alpha(\varnothing)=F$, тому $\alpha\not\models\Box\forall x\, p\to\forall x\Box p$.

Побудуємо загальну ТМС, в якій спростовується формула вигляду $\exists x\Box\Phi\to\Box\exists x\Phi$. Нехай $S=\{\alpha, \beta\}$, $R=\{\alpha\triangleright\beta\}$, $A_\alpha=\{a, b\}$, $A_\beta=\{b\}$. Візьмемо $p\in Ps$, для якого неістотні усі імена, окрім x, y. Задамо $p_\beta([x\mapsto b, y\mapsto b])=F$, $p_\beta([y\mapsto b])=T$. Із $p_\beta([x\mapsto b, y\mapsto b])=F$, згідно $A_\beta=\{b\}$, маємо $\exists x\, p_\beta([y\mapsto b])=F$, звідки $\Box\exists x\, p_\alpha([y\mapsto b])=F$. Із $p_\beta([y\mapsto b])=T$ маємо $\Box p_\alpha([x\mapsto a, y\mapsto b])=T$, звідки $\exists x\Box p_\alpha([y\mapsto b])=T$. Тому $(\exists x\Box p\to\Box\exists x\, p)_\alpha([y\mapsto b])=F$, звідки $\alpha\not\models\exists x\Box p\to\Box\exists x\, p$.

Зауважимо, що предикати p_β нееквітоні в обох цих ТМС.

Водночас маємо (див. [2]):

Приклад 2. Формули $\Box\exists x\Phi\to\exists x\Box\Phi$ та $\forall x\Box\Phi\to\Box\forall x\Phi$ не є всюди істинними вже для ТМЛ еквітонних предикатів.

Формула $\forall x\Box\Phi\to\Box\forall x\Phi$ відома як формула Баркан, формула $\Box\forall x\Phi\to\forall x\Box\Phi$ – це її конверсія. Як показують приклади 1 та 2, формула Баркан та її конверсія не є всюди істинними. Водночас [2] конверсія формули Баркан істинна в кожній загальній ТМС еквітонних предикатів.

Відношення логічного наслідку для множин формул

Специфікована станом формула має вигляд Φ^α, де Φ – формула мови, $\alpha\in$S – її специфікація, S – множина імен станів світу.

Нехай M – КНМС із множиною станів S, Γ – множина специфікованих станами формул, причому ці специфікації утворюють множину S. Γ *узгоджена* із КНМС M, якщо задана

ін'єкція S у S.

Нехай Δ та Γ – множини специфікованих станами формул.

Δ є логічним наслідком Γ в узгодженій із ними КНМС M (це позн. $\Gamma_M|=\Delta$), якщо для всіх $d\in{}^{V}A$ маємо: із того, що $\Phi_\alpha(d)=T$ для всіх $\Phi^\alpha\in\Gamma$, випливає, що неможливо $\Psi_\beta(d)=F$ для всіх $\Psi^\beta\in\Delta$.

Запис $\Gamma_M|=\Delta$ завжди означає узгодженість КНМС M із Γ та Δ.

Δ є логічним наслідком Γ відносно КНМС певного типу (позначаємо $\Gamma\,|=\Delta$), якщо $\Gamma_M|=\Delta$ для всіх КНМС M відповідного типу.

Звідси маємо: $\Gamma\,|\neq\Delta \Leftrightarrow$ існують узгоджена із Γ та Δ КНМС M та $d\in{}^{V}A$ такі, що $\Phi_\alpha(d)=T$ для всіх $\Phi^\alpha\in\Gamma$ та $\Psi_\beta(d)=F$ для всіх $\Psi^\beta\in\Delta$.

Наведемо властивості відношення наслідку для множин специфікованих станами формул. Немодальні властивості повторюють наведені в [1] властивості логічного наслідку для множин формул логіки еквітонних предикатів U, C, $\neg\vdash$, $\neg\dashv$, $\vee\vdash$, $\vee\dashv$, $RT_\vdash$, $RT_\dashv$, $\Phi N_\vdash$, $RR_\vdash$, $RR_\dashv$, $R\neg\vdash$, $R\neg\dashv$, $R\vee\vdash$, $R\vee\dashv$, $R\exists R_\vdash$, $R\exists R_\dashv$, $R\exists p_\vdash$, $R\exists p_\dashv$.

Для опису елімінації кванторів КНМЛ немонотонних предикатів використаємо (див., напр., [4]) спеціальні предикати-індикатори εz наявності значення для відповідного імені $z\in V$.

Предикати εz задамо так: $T(\varepsilon z)=\{d\,|\,d(z)\uparrow\}$, $F(\varepsilon z)=\{d\,|\,d(z)\downarrow\}$.

Наведемо властивості елімінації кванторів:

$\exists\varepsilon_\vdash)$ $\exists x\Phi^\alpha, \Gamma_M|=\Delta \Leftrightarrow R_z^x(\Phi)^\alpha, \Gamma_M|=\Delta, \varepsilon z^\alpha$

за умови $z\in V_T$ та $z\notin nm(\Gamma,\Delta,\exists x\Phi))$;

$\exists R\varepsilon_\vdash)$ $R_{\bar v}^{\bar u}(\exists x\Phi)^\alpha, \Gamma_M|=\Delta \Leftrightarrow R_{\bar v,z}^{\bar u,x}(\Phi)^\alpha, \Gamma\ R_z^x(\Phi), \Gamma_M|=\Delta, \varepsilon z^\alpha$

за умови $z\in V_T$ та $z\notin nm(\Gamma,\Delta,R_{\bar v}^{\bar u}(\exists x\Phi))$;

$\exists v_\dashv)$ $\Gamma_M|=\Delta,\exists x\Phi^\alpha,\varepsilon y^\alpha \Leftrightarrow \Gamma_M|=\Delta,\exists x\Phi^\alpha, R_z^x(\Phi)^\alpha,\varepsilon y^\alpha$;

$\exists Rv_\dashv)$ $\Gamma_M|=\Delta, R_{\bar v}^{\bar u}(\exists x\Phi)^\alpha,\varepsilon y^\alpha \Leftrightarrow$

$\Gamma_M|=\Delta, R_{\bar v}^{\bar u}(\exists x\Phi)^\alpha, R_{\bar v,y}^{\bar u,x}(\Phi)^\alpha,\varepsilon y^\alpha$.

Допоміжні властивості ε-розподілу:

εd) $\Gamma_M|=\Delta \Leftrightarrow \varepsilon y^\alpha,\Gamma_M|=\Delta$ та $\Gamma_M|=\Delta,\varepsilon y^\alpha$;

εv) $\Gamma_M|=\Delta \Leftrightarrow \Gamma_M|=\Delta,\varepsilon z^\alpha$ за умови $z\in V_T$ та $z\notin nm(\Gamma,\Delta)$.

Властивості, пов'язані з реномінаціями та модальностями:

$R\mathbin{⌘}_\vdash)$ $\Gamma, R_{\bar x}^{\bar y}(\mathbin{⌘}\Phi)^\alpha\ {}_M|=\Delta \Leftrightarrow \Gamma, \mathbin{⌘}R_{\bar x}^{\bar y}(\Phi)^\alpha\ {}_M|=\Delta$, де $\mathbin{⌘}\in Ms$;

$R\mathbin{⌘}_\dashv)$ $\Gamma\ {}_M|=\Delta, R_{\bar x}^{\bar y}(\mathbin{⌘}\Phi)^\alpha \Leftrightarrow \Gamma_M|=\Delta, \mathbin{⌘}R_{\bar x}^{\bar y}(\Phi)^\alpha$, де $\mathbin{⌘}\in Ms$.

Зокрема, для ММС маємо $RK_{i\vdash}$ та $RK_{i\dashv}$, де $K_i\in Ms$; для загальних ТМС – $R\Box_\vdash$ та $R\Box_\dashv$; для ТмМС – $R\Box_{\uparrow\vdash}$, $R\Box_{\downarrow\vdash}$, $R\Box_{\uparrow\dashv}$ та $R\Box_{\downarrow\dashv}$.

Розглянемо властивості, пов'язані з елімінацією модальностей. У випадку загальних ТМС маємо:

$\Box_{|-}$) $\Box\Phi^{\alpha}, \Gamma_M| = \Delta \Leftrightarrow \{\Phi^{\beta} \mid \alpha \triangleright \beta\} \cup \Gamma_M| = \Delta$;

$\Box_{-|}$) $\Gamma_M| = \Delta, \Box\Phi^{\alpha} \Leftrightarrow \Gamma_M| = \Delta, \Phi^{\beta}$ для всіх $\beta \in S$ таких, що $\alpha \triangleright \beta$.

У випадку ТмМС властивості $\Box_L$ та $\Box_R$ розщеплюються:

$\Box_{\uparrow|-}$) $\Box_{\uparrow}\Phi^{\alpha}, \Gamma_M| = \Delta \Leftrightarrow \{\Phi^{\beta} \mid \alpha \triangleright \beta\} \cup \Gamma_M| = \Delta$;

$\Box_{\uparrow-|}$) $\Gamma_M| = \Delta, \Box_{\uparrow}\Phi^{\alpha} \Leftrightarrow \Gamma_M| = \Delta, \Phi^{\beta}$ для всіх $\beta \in S$: $\alpha \triangleright \beta$;

$\Box_{\downarrow|-}$) $\Box_{\downarrow}\Phi^{\alpha}, \Gamma_M| = \Delta \Leftrightarrow \{\Phi^{\beta} \mid \beta \triangleright \alpha\} \cup \Gamma_M| = \Delta$;

$\Box_{\downarrow-|}$) $\Gamma_M| = \Delta, \Box_{\downarrow}\Phi^{\alpha} \Leftrightarrow \Gamma_M| = \Delta, \Phi^{\beta}$ для всіх $\beta \in S$: $\beta \triangleright \alpha$.

Для ММС маємо аналогічні до $\Box_{|-}$ та $\Box_{-|}$ властивості $K_{i\,|-}$ та $K_{i\,-|}$ (тут $K_i \in Ms$).

Властивості відношення наслідку для множин формул в даній КНМС індукують аналогічні властивості відношення логічного наслідку для множин формул (пишемо |= замість $_M|=$).

Властивості відношення логічного наслідку для множин специфікованих станами формул є семантичною основою для побудови секвенційних числень різних класів ТМЛ.

Висновки

В роботі запропоновано і досліджено транзиційні модальні логіки часткових квазіарних предикатів, не обмежених умовою монотонності (еквітонності). Описано семантичні моделі та мови чистих першопорядкових ТМЛ, досліджено їх семантичні властивості. Залежно від умов, накладених на відношення переходу, визначено різні класи ТМЛ. Розглянуто взаємодію модальних композицій із реномінаціями та кванторами. Показана відмінність властивостей ТМЛ немонотонних предикатів та ТМЛ монотонних (еквітонних) предикатів. Наведено властивості відношень логічного наслідку для множин специфікованих станами формул.

Список використаних джерел

1. Нікітченко М.С. Прикладна логіка / М.С. Нікітченко, С.С. Шкільняк. – Київ: ВПЦ Київський університет, 2013. – 278 с.
2. Шкільняк О.С. Семантичні властивості композиційно-номінативних модальних логік / О.С. Шкільняк // Проблеми програмування. – 2009. – № 4. – С. 11–23.
3. Шкільняк О.С. Семантичні моделі та секвенційні числення транзиційних модальних логік / О.С. Шкільняк // Компьютерная математика. – 2013. – Вып. 1. – С. 141–150.
4. Нікітченко М.С. Першопорядкові композиційно-номінативні логіки із узагальненими реномінаціями / М.С. Нікітченко, О.С. Шкільняк, С.С. Шкільняк // Проблеми програмування. – 2014. – № 2–3 – С. 17–28.

БЕЗКВАНТОРНІ КОМПОЗИЦІЙНО-НОМІНАТИВНІ ЛОГІКИ ФУНКЦІОНАЛЬНИХ РІВНІВ

С.С. Шкільняк, Д.Б. Волковицький
Київський національний університет імені Тараса Шевченка,
Україна
sssh@unicyb.kiev.ua, ttp@unicyb.kiev.ua

Вступ

Розроблено низку різноманітних логічних систем, які успішно використовуються в програмуванні. Такі системи зазвичай базуються на класичній логіці предикатів. Проте принципові обмеження класичної логіки зумовлюють необхідність побудови нових, програмно-орієнтованих логічних формалізмів. Таку побудову доцільно проводити на основі спільного для логіки й програмування композиційно-номінативного підходу (див. [1, 2]). На його базі збудовано і досліджено широкий спектр логік часткових предикатів. Такі логіки названо *композиційно-номінативними* (КНЛ). Вони базуються на загальних класах часткових відображень, заданих на наборах іменованих значень, – квазіарних відображень.

В роботі досліджуються нові класи КНЛ – безкванторні логіки квазіарних предикатів. Вони займають проміжне становище між пропозиційною логікою та першопорядковими КНЛ. В безкванторних логіках, що засвідчує їх назва, не використовуються потужні композиції квантифікації, характерні для першопорядкових логік.

Безкванторний рівень розпадається на декілька підрівнів:

– реномінативний (рівень РНЛ);

– реномінативний з рівністю (рівень РНЛР);

– реномінативний з строгою рівністю (рівень РНЛРС);

– безкванторно-функціональний (рівень БКНЛ);

– безкванторно-функціональний з рівністю (рівень БКНЛР);

– безкванторно-функціональний з строгою рівністю (рівень БКНЛРС).

Найабстрактнішим з цих підрівнів є реномінативний. Реномінативні логіки добре досліджені (див., напр., [1, 2]), започатковано [3] вивчення реномінативних логік з рівністю. В даній роботі пропонуються безкванторні логіки функціональних рівнів.

Поняття, які тут не визначаються, тлумачимо в сенсі [1, 2].

Рівні безкванторних логік квазіарних предикатів

На безкванторних рівнях функції та предикати – квазіарні, вони задаються на іменних множинах.

V-іменна множина (V-ІМ) над A – це однозначна функція вигляду $d : V \to A$. Множину всіх V-ІМ над A будемо позначати ${}^{V}A$.

Функції вигляду ${}^{V}A \to A$ назвемо *V*-квазіарними функціями на *A*.

Функції вигляду ${}^{V}A \to \{T, F\}$ назвемо *V*-квазіарними предикатами на *A*. Тут $\{T, F\}$ – множина істиннісних значень.

В цій роботі розглядаємо однозначні функції та предикати.

Область істинності та *область хибності* *V*-квазіарного предиката *P* на *A* – це множини

$$T(P) = \{d \in {}^{V}A \mid P(d) = T\};$$
$$F(P) = \{d \in {}^{V}A \mid P(d) = F\}.$$

Предикат $P : {}^{V}A \to \{T, F\}$ називаємо:

– неспростовним (частково істинним), якщо $F(P) = \varnothing$;

– тотожно істинним, якщо $T(P) = {}^{V}A$ та $F(P) = \varnothing$.

Класи *V*-квазіарних функцій на *A* і *V*-квазіарних предикатів на *A* позначаємо Fn^A і Pr^A.

V-квазіарна функція (предикат) *g* *еквітонна*, якщо:

$$g(d)\downarrow \text{ та } d \subseteq d' \;\Rightarrow\; g(d')\downarrow = g(d).$$

Стисло опишемо виділені рівні безкванторних логік.

Рівень РНЛ. Тут можна перейменовувати компоненти вхідних даних. Це дає змогу ввести композицію реномінації $\mathrm{R}^{\bar{v}}_{\bar{x}}$. Базові композиції РНЛ: логічні зв'язки $\neg$, $\vee$ та реномінація $\mathrm{R}^{\bar{v}}_{\bar{x}}$.

На рівнях РНЛР та РНЛРС додатково можна ототожнювати й розрізняти значення предметних імен за допомогою спеціальних 0-арних композицій – параметризованих за іменами предикатів рівності. Можна розглядати дві різновидності таких предикатів:

– слабкої (з точністю до визначеності) рівності $=_{xy}$;

– строгої (точної) рівності $\equiv_{xy}$.

Рівень РНЛР. Базові композиції: $\neg$, $\vee$, $\mathrm{R}^{\bar{v}}_{\bar{x}}$, $=_{xy}$.

Рівень РНЛРС. Базові композиції: $\neg$, $\vee$, $\mathrm{R}^{\bar{v}}_{\bar{x}}$, $\equiv_{xy}$.

На функціональних рівнях можна формувати нові базові значення для вхідних даних. Це дозволяє ввести композицію суперпозиції.

Нехай F^A – клас функцій вигляду $f : {}^{V}A \to R$. Параметрична $(n+1)$-арна композиція суперпозиції $\mathrm{S}^{v_1,\ldots,v_n} : F^A \times (Fn^A)^n \to F^A$ кожним функціям $f, g_1, \ldots, g_n$ зіставляє функцію $\mathrm{S}^{v_1,\ldots,v_n}(f, g_1, \ldots, g_n)$, значення якої для кожного $d \in {}^{V}A$ обчислюється так:

$$\mathrm{S}^{v_1,\ldots,v_n}(f, g_1, \ldots, g_n)(d) = f(d\nabla[v_1 \mapsto g_1(d), \ldots, v_n \mapsto g_n(d)]).$$

Виділення квазіарних функцій на *A* та квазіарних предикатів на *A* індукує виділення суперпозицій двох типів:

$$(Fn^A)^{n+1} \to Fn^A \;\text{ та }\; Pr^A \times (Fn^A)^n \to Pr^A.$$

Для роботи з окремими компонентами даних виділимо спеціальні 0-арні композиції – функції деномінації (розіменування).

Функції деномінації $'\nu$, де $\nu \in V$, задаємо так: $'\nu(d) = d(\nu)$.

Реномінації можна промоделювати за допомогою суперпозиції: маємо $\mathrm{R}^{u_1,\ldots,u_n}_{v_1,\ldots,v_n}(f) = \mathrm{S}^{v_1,\ldots v_n}(f, '\nu_1, \ldots, '\nu_n)$ для кожної $f \in Fn^A \cup Pr^A$.

Рівень БКНЛ. Базові композиції: $\neg$, $\vee$, $\mathrm{S}^{\bar{x}}$, $'\nu$.

На функціональних рівнях з рівністю додатково можна ототожнювати й розрізняти предметні значення, що дає змогу ввести спеціальну композицію рівності. Розглядаємо дві її різновидності:

– слабкої (з точністю до визначеності) рівності $=$.

– строгої (точної) рівності $\equiv$.

Композиції $=$ та $\equiv$ мають вигляд $Fn^A \times Fn^A \to Pr^A$, вони задаються так. Для кожних $f \in Fn^A$ та $d \in {}^V A$:

$$=\!(f, g)(d) = \begin{cases} T, \text{ якщо } f(d)\downarrow \text{ та } g(d)\downarrow \text{ та } f(d) = g(d), \\ F, \text{ якщо } f(d)\downarrow \text{ та } g(d)\downarrow \text{ та } f(d) \neq g(d), \\ \text{невизначене, якщо } f(d)\uparrow \text{ або } g(d)\uparrow; \end{cases}$$

$$\equiv\!(f, g)(d) = \begin{cases} T, \text{ якщо } (f(d)\downarrow,\ g(d)\downarrow \text{ та } f(d) = g(d)) \\ \quad \text{або } (f(d)\uparrow \text{ та } g(d)\uparrow), \\ F, \text{ якщо } (f(d)\downarrow,\ g(d)\downarrow,\ f(d) \neq g(d)) \\ \quad \text{або } (f(d)\downarrow,\ g(d)\uparrow) \text{ або } (f(d)\uparrow,\ g(d)\downarrow). \end{cases}$$

Рівень БКНЛР. Базові композиції: $\neg$, $\vee$, $\mathrm{S}^{\bar{x}}$, $'\nu$, $=$.

Рівень БКНЛРС. Базові композиції: $\neg$, $\vee$, $\mathrm{S}^{\bar{x}}$, $'\nu$, $\equiv$.

Теорема 1. Композиції $\mathrm{S}^{\bar{v}}$ та $=$ зберігають еквітонність.

Водночас композиція $\equiv$ еквітонність не зберігає. Тому на рівні БКНЛРС логіки еквітонних предикатів розглядати не можна.

Теорема 2. $=\!(f, g)$ неспростовний $\Leftrightarrow f; g$; $\equiv\!(f, g)$ тотожно істинний $\Leftrightarrow f = g$.

Зауважимо, що композиція еквіваленції $\leftrightarrow$ для предикатів є аналогом композиції слабкої рівності для функцій.

Властивості композицій суперпозиції та рівності

S$\neg$) Дистрибутивність суперпозиції щодо $\neg$:

$$\mathrm{S}^{\bar{v}}(\neg P, \bar{f}) = \neg \mathrm{S}^{\bar{v}}(P, \bar{f}).$$

S$\vee$) Дистрибутивність суперпозиції щодо $\vee$:

$$\mathrm{S}^{\bar{v}}(P \vee Q, \bar{f}) = \mathrm{S}^{\bar{v}}(P, \bar{f}) \vee \mathrm{S}^{\bar{v}}(Q, \bar{f}).$$

SS) Згортка суперпозицій: (тут $\varphi \in Fn^A \cup Pr^A$, введені позначення $\overline{u}, \overline{t}, \overline{x}, \overline{r}, \overline{w}, \overline{v}, \overline{s}$ відповідно для $u_1,...,u_n$; $t_1,...,t_n$; $x_1,...,x_k$; $r_1,...,r_k$; $w_1,...,w_k$; $v_1,...,v_m$; $s_1,...,s_m$): $S^{\overline{u},\overline{x}}(S^{\overline{x},\overline{v}}(\varphi,\overline{r},\overline{s}),\overline{t},\overline{w}) =$

$= S^{\overline{u},\overline{x},\overline{v}}(\varphi,\overline{t},S^{\overline{u},\overline{x}}(r_1,\overline{t},\overline{w}),...,S^{\overline{u},\overline{x}}(r_k,\overline{t},\overline{w}),S^{\overline{u},\overline{x}}(s_1,\overline{t},\overline{w}),...,S^{\overline{u},\overline{x}}(s_m,\overline{t},\overline{w}))$.

ZS) Згортка імен (тут $\varphi \in Fn^A \cup Pr^A$):

$$S^{x_1,...,x_m,v_1,...,v_n}(\varphi,'x_1,...,'x_m,g_1,...,g_n) = S^{v_1,...,v_n}(\varphi,g_1,...,g_n).$$

Зокрема, маємо $S^{x_1,...,x_m}(\varphi,'x_1,...,'x_m) = \varphi$.

DD) Згортка неістотних імен для функцій розіменування:

$$S^{v_1,...,v_n}('x,g_1,...,g_n) = 'x \text{ за умови } x \notin \{\overline{v}\}.$$

DS) Спрощення для функцій розіменування:

$S^{x,v_1,...,v_n}('x,f,g_1,...,g_n) = f$. Зокрема, маємо $S^x('x,f) = f$.

ZN) Згортка за неістотним іменем (тут $\varphi \in Fn^A \cup Pr^A$):

$S^{x,v_1,...,v_n}(\varphi,f,g_1,...,g_n) = S^{v_1,...,v_n}(\varphi,g_1,...,g_n)$, якщо x строго неістотне для φ. Зокрема, при цій умові маємо $S^x(\varphi,f) = \varphi$.

$S^{x,v_1,...,v_n}(\varphi,f,g_1,...,g_n)$; $S^{v_1,...,v_n}(\varphi,g_1,...,g_n)$, якщо x неістотне для φ. Зокрема, при цій умові маємо $S^x(\varphi,f)$; φ.

Далі вважаємо, що $P \in Pr^A$ та $h, f, f_1,...,f_n, g, g_1,..., g_n \in Fn^A$.

Наведемо властивості слабкої рівності.

Rf) рефлексивність: кожний предикат $=(f,f)$ неспростовний;

Sm) симетричність: $=(f,g) = =(g,f)$;

Tr) транзитивність: $=(f,g)$ & $=(g,h) \Rightarrow =(f,h)$;

EF) $=(f_1,g_1)$ &...& $=(f_n,g_n) \Rightarrow$

$\Rightarrow = (S^{v_1,...,v_n}(f,f_1,...,f_n), S^{v_1,...,v_n}(f,g_1,...,g_n))$;

EP) $=(f_1,g_1)$ &...& $=(f_n,g_n) \Rightarrow$

$\Rightarrow S^{v_1,...,v_n}(P,f_1,...,f_n) \leftrightarrow S^{v_1,...,v_n}(P,g_1,...,g_n)$;

SE) дистрибутивність суперпозиції щодо рівності:

$S^{v_1,...,v_n}(=(g,h),f_1,...,f_n) = = (S^{v_1,...,v_n}(g,f_1,...,f_n), S^{v_1,...,v_n}(h,f_1,...,f_n))$.

Наведемо властивості строгої рівності.

Rfs) кожний предикат $\equiv(f,f)$ тотожно істинний;

Sms) $\equiv(f,g) = \equiv(g,f)$;

Trs) $f \equiv g$ та $g \equiv h$ тотожно істинні $\Rightarrow f \equiv h$ тотожно істинний;

EsF) якщо $f_1 \equiv g_1, ..., f_n \equiv g_n$ тотожно істинні, то

$S^{v_1,...,v_n}(h,f_1,...,f_n) = S^{v_1,...,v_n}(h,g_1,...,g_n)$, тобто предикат

$\equiv (S^{v_1,...,v_n}(h,f_1,...,f_n), S^{v_1,...,v_n}(h,g_1,...,g_n))$ тотожно істинний;

EsP) якщо $f_1 \equiv g_1, ..., f_n \equiv g_n$ тотожно істинні, то

$$S^{v_1,...,v_n}(P, f_1,..., f_n) = S^{v_1,...,v_n}(P, g_1,..., g_n);$$

SsE) $S^{v_1,...,v_n}(\equiv(g,h), f_1,..., f_n) =$

$= \equiv(S^{v_1,...,v_n}(g, f_1,..., f_n),\ S^{v_1,...,v_n}(h, f_1,..., f_n))$.

Мови та семантичні властивості логік безкванторних функціональних рівнів

Семантичними моделями КНЛ функціональних рівнів є композиційні системи *V*-квазіарних функцій і предикатів. Це трійки вигляду (^{V}A, $Fn^{A}\cup Pr^{A}$, C), де Pr^{A} – множина *V*-квазіарних предикатів на A, Fn^{A} – множина *V*-квазіарних функцій на A, C – множина композицій над $Fn^{A}\cup Pr^{A}$. Така C визначається базовими композиціями відповідного рівня.

Композиційна система (^{V}A, $Fn^{A}\cup Pr^{A}$, C) визначає композиційну алгебру *V*-квазіарних функцій і предикатів ($Fn^{A}\cup Pr^{A}$, C) та алгебру (алгебраїчну систему) даних із квазіарними функціями і предикатами (A, $Fn^{A}\cup Pr^{A}$). Побудова композиційної алгебри дає змогу визначити мову КНЛ.

Опишемо спочатку мову БКНЛ. Алфавіт мови: множина предметних імен *V*, множина $Dns = \{'v \mid v \in V\}$ деномінаційних символів (ДНС), тобто імен функцій розіменування, множини *Fns* та *Ps* функціональних (ФНС) і предикатних символів (ПС), символи базових композицій $\neg$, $\vee$, $S^{\bar{v}}$, $'v$. Множину $Fns\cup Dns$ позначимо *Fs*. Множину $\sigma = Fns\cup Ps$ назвемо сигнатурою мови БКНЛ.

Множини термів *Tr* і формул *Fr* вводимо індуктивно.

Т0. Кожний ФНС та кожний ДНС є термом (атомарним).

Т1. Нехай $\tau, t_1,..., t_n \in Tr$; тоді $S^{v_1,...,v_n}\tau t_1...t_n \in Tr$.

Ф0. Кожний ПС є формулою (атомарною).

Ф1. Нехай $\Phi, \Psi \in Fr$, $t_1,..., t_n \in Tr$; тоді $S^{v_1,...,v_n}\Phi t_1...t_n \in Fr$,

$\neg\Phi \in Fr$, $\vee\Phi\Psi \in Fr$.

Мови БКНЛР та БКНЛРС відрізняються від мови БКНЛ розширенням алфавіту символом = чи $\equiv$ та доповненням індуктивного визначення формули пунктом ФЕ для мови БКНЛР і пунктом ФЕS для мови БКНЛРС.

ФЕ. Нехай $t, s \in Tr$; тоді $=ts \in Fr$.

ФЕS. Нехай $t, s \in Tr$; тоді $\equiv ts \in Fr$.

Інтерпретуємо мови КНЛ функціональних рівнів на відповідних композиційних системах квазіарних функцій та предикатів. Відображення інтерпретації $I : Tr\cup Fr \to Fn^{A}\cup Pr^{A}$ є продовженням відображення $I : Fs\cup Ps \to Fn^{A}\cup Pr^{A}$ згідно побудови термів та фор-

мул із простіших за допомогою символів композицій (при цьому $I('v)=$'v для кожного $'v \in Dns$):

IT) $I(S^{v_1,\dots v_n}\tau t_1 \dots t_n) = \mathrm{S}^{v_1,\dots v_n}(I(\tau), I(t_1), \dots, I(t_n))$;

IФ) $I(S^{v_1,\dots v_n}\Phi t_1 \dots t_n) = \mathrm{S}^{v_1,\dots v_n}(I(\Phi), I(t_1), \dots, I(t_n))$;

$I(\neg\Phi) = \neg(I(\Phi))$, $I(\vee\Phi\Psi) = \vee(I(\Phi), I(\Psi))$.

Для мови БКНЛР та мови БКНЛРС додатково відповідно маємо $I(=ts) = =(I(t), I(s))$ та $I(\equiv ts) = \equiv(I(t), I(s))$.

Відображення I прив'язує алгебраїчну систему (АС) даних $(A, Fn^A \cup Pr^A)$ до мови логіки. Отримуємо об'єкти вигляду $\boldsymbol{A} = ((A, Fn^A \cup Pr^A), I)$ – АС з доданою сигнатурою (мовою) [1, 2], які позначаємо (A, I). Кожна така $\boldsymbol{A}$ задає композиційну систему $({}^V A, Fn^A \cup Pr^A, C)$, тому АС з доданою сигнатурою є семантичними моделями мови.

Предикат $I(\Phi)$, який є значенням формули Ф при інтерпретації на $\boldsymbol{A} = (A, I)$, позначаємо Φ_A.

Формула Ф частково істинна при інтерпретації на моделі мови $\boldsymbol{A}$, або $\boldsymbol{A}$-неспростовна (позн. $\boldsymbol{A} \models \Phi$), якщо Φ_A – частково істинний (неспростовний) предикат.

Формула Ф частково істинна, або неспростовна (позн. $\models \Phi$), якщо Ф є $\boldsymbol{A}$-неспростовною на кожній моделі мови $\boldsymbol{A}$.

Подібним чином даємо визначення тотожно істинної на $\boldsymbol{A}$ (позн. $\boldsymbol{A} \models \Phi$), та тотожно істинної (позн. $\models \Phi$) формули.

Традиційним чином [1, 2] визначаємо відношення логічного наслідку, слабкого логічного наслідку, тавтологічного наслідку, логічної еквівалентності $\sim$, строгої логічної еквівалентності $\sim_{TF}$.

Семантичні властивості формул КНЛ функціональних рівнів індуковані відповідними властивостями композицій та записуються аналогічно. Наведемо для прикладу деякі такі властивості:

$S\neg$) $S^{\overline{v}}(\neg\Phi, \overline{t}) \sim_{TF} \neg S^{\overline{v}}(\Phi, \overline{t})$;

$S\vee$) $S^{\overline{v}}(\Phi \vee \Psi, \overline{t}) \sim_{TF} S^{\overline{v}}(\Phi, \overline{t}) \vee S^{\overline{v}}(\Psi, \overline{t})$;

DD) $\models S^{v_1,\dots,v_n}('x, g_1, \dots, g_n) \equiv 'x$ за умови $x \notin \{\overline{v}\}$;

DS) $\models S^{x,v_1,\dots,v_n}('x, f, g_1, \dots, g_n) \equiv f$; зокрема, $\models S^x('x, f) \equiv f$;

Rf) $\models t = t$;

Sm) $\models s = t \leftrightarrow t = s$;

Tr) $\models s = t \to t = h \to s = h$;

ET) $\models t_1 = s_1 \to \dots \to t_n = s_n \to S^{v_1,\dots v_n}(\tau, t_1, \dots, t_n) = S^{v_1,\dots v_n}(\tau, s_1, \dots, s_n)$;

$E\Phi$) $\models t_1 = s_1 \to \dots \to t_n = s_n \to S^{v_1,\dots v_n}(\Phi, t_1, \dots, t_n) \to S^{v_1,\dots v_n}(\Phi, s_1, \dots, s_n)$;

SE) $\models S^{\overline{v}}(r = s, \overline{t}) \leftrightarrow S^{\overline{v}}(r, \overline{t}) = S^{\overline{v}}(s, \overline{t})$;

Rfs) $|\equiv t \equiv t$;

Sms) $s = t \sim_{TF} t = s$;

SsE) $S^{\bar{v}}(r = s, \bar{t}) \square_{TF} S^{\bar{v}}(r, \bar{t}) = S^{\bar{v}}(s, \bar{t})$.

Основою еквівалентних перетворень формул КНЛ функціональних рівнів є теорема еквівалентності.

Теорема 3. Нехай Φ' отримано з формули Φ заміною деяких входжень $\Phi_1, ..., \Phi_n$ на $\Psi_1, ..., \Psi_n$.

Якщо $\Phi_1 \sim \Psi_1, ..., \Phi_n \sim \Psi_n$, то $\Phi \sim \Phi'$.

Теорема 4. Нехай Φ' отримано з формули Φ заміною деяких входжень $\Phi_1, ..., \Phi_n$ на $\Psi_1, ..., \Psi_n$.

Якщо $\Phi_1 \sim_{TF} \Psi_1, ..., \Phi_n \sim_{TF} \Psi_n$, то $\Phi \sim_{TF} \Phi'$.

У випадку БКНЛР та БКНЛРС до них додаються теореми рівності для термів та для формул.

Теорема 5. Нехай терм τ' отримано з терма τ заміною деяких входжень термів $s_1, ..., s_n$ на терми $t_1, ..., t_n$ відповідно.

Якщо $|= s_1 = t_1, ..., |= s_n = t_n$, *mo* $|= \tau = \tau'$.

Якщо $|\equiv s_1 \equiv t_1, ..., |\equiv s_n \equiv t_n$, *mo* $|\equiv \tau \equiv \tau'$.

Теорема 6. Нехай Φ' отримано з формули Φ заміною деяких входжень термів $s_1, ..., s_n$ на терми $t_1, ..., t_n$ відповідно.

Якщо $|= s_1 = t_1, ..., |= s_n = t_n$, то $\Phi \sim \Phi'$.

Якщо $|\equiv s_1 \equiv t_1, ..., |\equiv s_n \equiv t_n$, *mo* $\Phi \sim_{TF} \Phi'$.

Висновки

Досліджено безкванторні композиційно-номінативні логіки квазіарних предикатів функціональних рівнів. Вони є розширенням відомих реномінативних логік. Запропоновано безкванторні логіки функціонального рівня та дві різновидності таких логік функціонального рівня з рівністю: із композицією слабкої (з точністю до визначеності) рівності = та композицією строгої (точної) рівності ≡. Описано властивості композицій суперпозиції та слабкої і строгої рівності. Розглянуто семантичні моделі та мови пропонованих логік, досліджено їх семантичні властивості. В наступних роботах планується дослідити властивості відношень логічного наслідку для множин формул цих логік, що дасть змогу побудувати для запропонованих класів логік відповідних числень секвенційного типу.

Список використаних джерел

1. Нікітченко М.С. Математична логіка та теорія алгоритмів / М.С. Нікітченко, С.С. Шкільняк. – Київ: ВПЦ Київський університет, 2008. – 528 с.

2. Нікітченко М.С. Прикладна логіка / М.С. Нікітченко, С.С. Шкільняк. – Київ: ВПЦ Київський університет, 2013. – 278 с.
3. Шкільняк С.С. Реномінативні композиційно-номінативні логіки з предикатами рівності / С.С. Шкільняк, Д.Б. Волковицький // Вісник Київського національного університету імені Тараса Шевченка. Сер.: фіз.-мат. науки. – 2014. – Вип. 3. – С. 195–202.

АНАЛІЗ ВПЛИВУ ЗНАЧЕНЬ ФАКТОРІВ РАНЖУВАННЯ ВЕБ-СТОРІНОК

І.С.Яковлєв, І.Д. Яковлєва, І.Д. Лісовенко
Чернівецький національний університет імені Юрія Федьковича, Україна
igorroka@gmail.com, iakovlieva@ya.ru, liss_i@mail.ru

Вступ

Процедура аналізу пошукових запитів необхідна для вивчення методів, якими користуються пошукові системи безпосередньо вивчаючи сайти, що займають верхні рядки рейтингу (ТОП) пошукової видачі та визначення принципів створення веб-сторінок з метою потрапляння в ТОП пошукової видачі.

Щоб краще розібратися в роботі пошукових алгоритмів необхідно виявити кореляцію між характеристиками веб-сторінок і їх позиціями у видачі пошукових систем. Традиційно дослідження складається з двох частин – опитування думки експертів та аналізу пошукової видачі [1].

До основних факторів, від яких залежить ранжування сайту можна віднести наступні [2, 3]:

- контент - кількість знаків у статті, співвідношення основної частини до частини, що повторюється на всіх сторінках, щільність входження ключового запиту, частота оновлення контенту на сайті;
- засоби внутрішньої оптимізації, що вказують, як використовуються метатеги, теги форматування, як організована навігація по сайту і структура внутрішніх посилань;
- засоби зовнішньої оптимізації - аналіз зовнішніх посилань, їх кількість, тематичність аналізованого ресурсу і вказівка на розташування посилання.

Як показано в табл. 1, на даний момент існуючі програмні засоби [4-7] виконують аналіз пошукової видачі по невеликим множинам параметрів та запитів і не надають автоматичний аналіз впливу значень факторів ранжування веб-сторінок на релевантність

до відповідного пошукового запиту; тому актуальним є розробка системи, що узагальнює дані, отримані в ході аналізу пошукової видачі, і робить висновки, ґрунтуючись на цих даних.

Особливості формування релевантності

Пошукова система (ПС) - це технічний засіб, за допомогою якого можна знайти дані, які вже розміщені в мережі. Головне завдання ПС – це пошук інформації, релевантної інформаційним потребам користувача. Поняття релевантності є одним з ключових в оптимізації сайтів під ті або інші пошукові запити. Технічно, релевантність – це відсоток входження ключового запиту в загальному об'ємі тексту, що розраховується через співвідношення кількості входжень запиту до загальної кількості слів в тексті. В термінах пошуку – це міра відповідності результатів пошуку завданню, поставленому в пошуковому запиті. Нерелевантний документ — документ, що був відібраний у результаті інформаційного пошуку, але зміст якого не відповідає запиту користувача.

Таблиця 1. Аналіз існуючих програмних засобів впливу значень факторів ранжування сайту на релевантність до відповідного пошукового запиту

	Програмні засоби	Аналіз				
		кожного сайту з десятки	контенту	зовнішніх посилань	внутрішніх посилань	отриманих результатів
1.	Seolib	+	-	+ -	-	-
2.	Samurai Market	+	-	+	-	-
3.	SEOquake	+	+ -	+	+ -	-
4.	FirstPage Analyzer	+	+ -	-	-	+ -

Отже, релевантність визначає, наскільки повно той або інший документ відповідає критеріям, вказаним у запиті користувача. Необхідно враховувати, що в кожній пошуковій системі працює власна програма, що індексує веб-сторінки, кожна система індексує сторінки своїм особливим способом і пріоритети при пошуку за індексами теж різні. Тому запит за одними і тими ж ключовими словами в кожній з пошукових систем породжує різні результати.

Для пошукових систем високорелевантним текстом вважається такий, у якому входження запиту в текст приблизно дорівнює 4-7% [8] – меншого відсотку може не вистачити, більший відсоток може призвести до того, що система визнає текст за пошуковий спам і накладе на сторінку певний знижуючий фільтр.

Щоб оцінити ефективність пошукової системи, обчислюють наступні статистичні величини:

1. Точність – кількість релевантних документів в результаті, розділене на загальну кількість документів в результаті.
2. Повнота – кількість релевантних документів в результаті, розділене на загальну кількість релевантних документів в колекції.

В той же час, релевантність пошуку є суб'єктивним поняттям, оскільки результати пошуку, які підходять для одного користувача, можуть не підходити для іншого.

Підсистема аналізу впливу факторів ранжування на релевантність веб-сторінки в пошуковій системі

У даній роботі для розв’язання поставленої задачі створене програмне забезпечення зі зручним користувальницьким інтерфейсом і набором необхідних інструментів, за допомогою яких спрощується процес автоматичного аналізу впливу значень факторів ранжування веб-сторінок на релевантність до відповідного пошукового запиту.

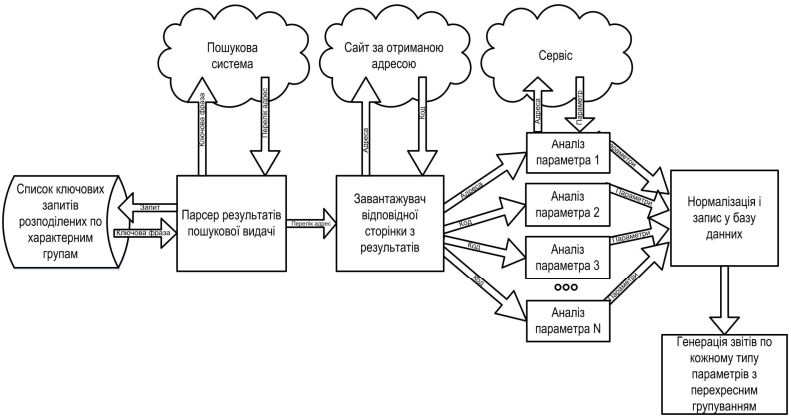


Рис. 1. Схема розробленої підсистеми аналізу впливу факторів ранжування на релевантність веб-сторінки в пошуковій системі

Основним призначення даної програми є визначення оптимальних числових для підвищення рейтингу веб-сторінки в пошуковій системі

Процедура аналізу відбувається таким чином: в заданий момент часу сервер запускає на виконання модуль, який займається збиранням та обробкою вхідних даних. Модуль опитує пошукові системи за пошуковими запитами, що взяті з бази даних. Отримавши, по 10 сайтів в результатах пошуку для кожного запиту, дані передаються до модуля аналізу, де відповідні адреси веб-веб-сторінок знову опитуються, завантажується їх html-код, інтегровані скрипти та дані, що стосуються цієї веб-сторінки з інших серверів та ін. Таким чином, вся отримана інформація записується в базу даних. Проаналізовані дані обробляються у відповідності до допустимих і граничних параметрів та генеруються графічні звіти. Користувач може переглянути ці дані і зробити власні висновки або користуватися запропонованими системою звітами.

Наведемо покроковий алгоритм аналізу числових характеристик факторів ранжування веб-сторінок:

Крок 1. З бази даних циклічно вибираються пошукові запити по одному та передаються на крок 2.

Крок 2. Для кожного пошукового запиту завантажується сторінка результатів пошукової видачі, з якої вибираються адреси веб-сторінок релевантних даному запиту. Кожна сторінка передається на крок 3.

Крок 3. За вхідною адресою завантажується html-код та формується дерево вузлів DOM для подальшого аналізу на наступних кроках.

Крок 4. Кожен блок аналізу визначає чисельне значення відповідного аналізованого параметру.

Крок 5. Усі опрацьовані дані записуються у базу даних.

Крок 6. Для кожного параметру генерується сторінка з звітами по даному аналізу, що групується за тематикою та типом запиту.

Для того, щоб збільшити кількість переходів з пошукових систем на сайт, необхідно оптимізовувати характеристики веб-сторінок за критеріями, які висуваються пошуковими системами.

Процедура аналізу алгоритму роботи пошукової системи побудована на визначенні параметрів, що теоретично впливають на релевантність веб-сторінки. В програмі аналізуються такі характеристики як:

1. Фактори сайту (зовнішні параметри): вік домену; кількість унікальних веб-сторінок; Pagerank веб-сторінки; кількість гіперпосилань на сайт; кількість пінгів з соціальних мереж.
2. Текстові фактори веб-сторінки (внутрішні параметри): ключова фраза в тезі Title; чи тег Title починається ключовою фразою;

частота ключової фрази в документі; ключова фраза в тегах H1, H2, H3; останнє оновлення документа.

Висновки

В результаті проведених досліджень визначено параметри і зроблено наступні висновки:

У транзакційних запитах треба надавати перевагу збільшенню PageRank сторінки, а в інформаційних – посиланням на сайт загалом – в основному на головну сторінку сайту.

Для інформаційних запитів більше впливає на позицію текстове наповнення сайту ніж «авторитет» сторінки.

Підтверджено, що для новин та пошукових запитів, що набувають популярності можуть бути релевантними нові сторінки.

Високим позиціям веб-сторінки завдячують інтенсивній підтримці з соцмереж.

Наявність в ТОП пошукової видачі сайтів з більшим віком порівняно з новими сайтами.

Відмінність в активності згадувань сайту в соціальних мережах для двох типів запитів – «транзакційних» та «навігаційних».

Список використаних джерел

1. 2013 Search Engine Ranking Factors: [Електронний ресурс] . – Режим доступу.: http://moz.com/search-ranking-factors
2. Инфографика – 200 факторов ранжирования Google: [Електронний ресурс] // Режим доступу.: http://seoprofy.ua/blog/prodvizhenie-sajtov/200-google-factors
3. US patent 6285999, Page, Lawrence, G06F 17/00 (20060101)«Method for node ranking in a linked database», issued September 4, 2001, assigned to The Board of Trustees of the Leland Stanford Junior University
4. Что может Seolib.ru. [Електронний ресурс]. – Режим доступу: https://www.seolib.ru/info/capability
5. Keyword Analysis Research - Market Samurai [Електронний ресурс]. – Режим доступу: http://www.marketsamurai.com/full-version/wdcu5b/
6. SEOquake - seo toolbar. [Електронний ресурс]. – Режим доступу: http://www.seoquake.com/ru_index.php
7. Software – FirstPageAnalyzer. [Електронний ресурс]. – Режим доступу:http://www.boogl.ru/firstpageanalyzer.php
8. Jones K. S. A statistical interpretation of term specificity and its application in retrieval // Journal of Documentation : журнал. – MCB University: MCB University Press, 2004. – Т. 60.– № 5.– С. 493-502.

ПОБУДОВА ЦЕНТРАЛЬНИХ ОСЕЙ ПАЛЬЦІВ НА РЕНТГЕНОГРАМІ КИСТІ РУКИ В ЗАДАЧІ КОМП'ЮТЕРНОЇ ОЦІНКИ КІСТКОВОГО ВІКУ

С.І. Степанюк
Східноєвропейський національний університет імені Лесі Українки, Луцьк, Україна
StepanyukSI@gmail.com

Вступ

У педіатричній радіології часто виконується оцінка кісткового віку дитини. Вік встановлюється за допомогою аналізу рентгенограми кисті лівої руки. Невідповідність між отриманим віком та хронологічним вказує на порушення розвитку скелету в цілому. Існують два основні методи аналізу рентгенограми руки: Greulich and Pyle (GP, [1]) та Tanner-Whitehouse(TW, [2]). Перший базується на порівнянні вхідної рентгенограми із атласом еталонних, згрупованих за віком і статтю. Кожен еталон відповідає деякому віку. Обравши найбільш подібне еталонне зображення, одержується прогнозований кістковий вік. У порівнянні з GP методом, де рентгенограма розглядається як єдине ціле, TW метод передбачає детальний аналіз окремо усіх або частини кісток. Знаходяться певні числові характеристики, на основі яких і розраховується кістковий вік.

Комп'ютерні системи автоматичної оцінки кісткового віку звільняють рентгенолога від трудомісткої роботи, забезпечують об'єктивність аналізу і точнішу оцінку. Зазвичай, такі системи складаються із наступних функціональних блоків: попереднього покращання цифрової рентгенограми; ротації до вертикального положення; усунення фону, включаючи радіологічні маркери; побудови центральних осей пальців (осей симетрії); локалізації ділянок інтересу (region of interest, ROI); обробки ROI (масштабування, ротації, сегментації, проведення числових вимірювань тощо); ґенератора прогнозованого віку.

Побудова центральних осей є однією з критичних задач комп'ютерної оцінки кісткового віку, оскільки осі в подальшому використовуються для: 1) локалізації ділянок інтересу; 2) одержання довільних ділянок пальців; 3) вимірювання числових характеристик ROI (наприклад, проводяться перпендикулярні лінії до осей); 4) швидкої ротації зображень окремих пальців та ін. Отже, якість побудови центральних осей безпосередньо впливає на прецизійність оцінки віку. Виходячи з цього, актуальною є розробка технології побудови центральних осей пальців, що буде робастною до їх форми і положення.

Існуючі підходи до знаходження центральних осей

Базовий підхід до побудови центральних осей описаний в роботі [3]. Вхідне зображення з попередньо усуненим фоном сканується горизонтальними лініями (рис. 1).

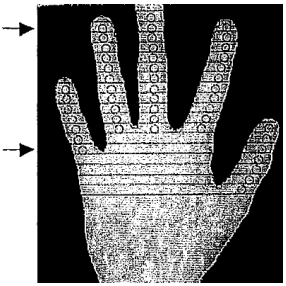


Рис. 1. Зображення із нанесеними профілями (отримано із [3])

Рис. 2. Приклад нестандартної рентгенограми (взято із http://www.ipilab.org/BAAweb/)

Опираючись на апріорні дані щодо розміщення руки на рентгенограмі, визначаються дві лінії: одна обмежує середній палець зверху; інша – нижню межу ділянки фалангів. Всі лінії, що знаходяться між ними, утворюють масиви горизонтальних профілів зображення. Кожен профіль містить стільки відрізків, скільки на його рівні розміщено незв'язних фрагментів. Вибравши центральні точки цих відрізків на кожному профілі, будуються масиви точок, що формують осі окремих пальців (на рис. 1 ці точки зображено кільцями). Даний підхід вимагає, щоб рука на рентгенограмі була розміщена в строго вертикальному положенні. З іншої сторони, точність виділення осей алгоритмом буде тим нижчою, чим більше положення окремого пальця відхиляється від вертикального. Оскільки алгоритм оперує із зображенням в цілому, то у багатьох зображеннях неможливо одночасно забезпечити положення усіх пальців близьким до вертикального. Наприклад, на рис. 2, незважаючи на те, що рука розміщена вертикально, положення великого пальця близьке до горизонтального. Якщо говорити про робастність системи оцінки до варіацій положення руки на рентгенограмі, то це обмеження є дуже вагомим. Тому актуальним є розроблення технології автоматичної побудови осей пальців, що не залежить від їх положення, розмірів та орієнтації на рентгенограмі.

У роботі [4] запропоновано алгоритм побудови осей, який будує вісь, оперуючи зображеннями локалізованих окремих пальців. Авторами цієї статті розроблено алгоритм, що знаходить

ділянки тільки трьох пальців (середнього та зліва і справа від нього). Варто відмітити, що цей алгоритм є нестійким до положення руки на зображенні та базується на певній апріорній інформації. Наступним кроком проводиться ротація ділянок локалізованих пальців до вертикального положення, знаходяться зовнішні контури. Після цього застосовується підхід аналогічний до [3]. Зображення із контуром сканується горизонтальними лініями і знаходяться центральні точки профілів. Основним обмеженням алгоритму є те, що він виділяє тільки осі трьох пальців.

У статті [5] вхідне зображення із усуненим фоном сканується за допомогою так званих вагових функцій (wedge functions), з метою отримання ключових точок пальців для їх подальшої локалізації і ротації фрагментів пальців. Наступним кроком на зображеннях (у відтінках сірого) окремих пальців знаходяться центральні точки, які і будуть представляти осі симетрії. Оскільки вагова функція зазвичай незадовільно знаходить ключові точки великого пальця, тому для його локалізації використовується додатковий алгоритм. Алгоритм базується на припущеннях щодо анатомічної форми і вимагає, аби рука на вхідному зображенні була розміщена в стандартизованій формі (вертикально вирівняна і по центру).

У роботі [6] запропоновано принципово інший від вищеописаних підхід до побудови осей. Суть його полягає в тому, що до бінарного зображення руки із відсутнім фоном застосовується алгоритм потоншення (скелетонізація), в результаті чого, в ідеальному випадку, кожному пальцеві буде відповідати деяка ламана лінія. Ці лінії інтерполюються прямими та коригуються, виходячи із апріорних знань про анатомічну форму руки. У статті наголошується, що цей алгоритм дає незадовільні результати для побудови осі великого пальця, тому потрібний додатковий алгоритм. З іншого боку, не можна гарантувати, що отримана ламана лінія займатиме центральне положення по всій довжині пальця. Як показують експерименти, такі ламані мають властивість на кінцівках пальців значно відхилятися від центру. Цей ефект, в свою чергу, спотворює геометричне розміщення осей.

Технологія побудови центральних осей

У даній роботі розроблено технологію побудови центральних осей, яка знімає обмеження вищеописаних існуючих підходів. Нехай на вхід подається рентгенографічне зображення руки розмірності $m \times n$. Будемо розглядати зображення як напівтонову функцію $F : D \to [0;255], D \in Z^2$. Запропонована

технологія передбачає наступні етапи обробки: 1) усунення фону і одержання бінарного зображення силуету руки; 2) виокремлення пальців; 3) ротація окремих пальців до вертикального положення на деякий кут φ_i, де $i = \overline{1,5}$ – номер пальця; 4) сканування горизонтальними лініями зображень кожного пальця окремо і знаходження центральних точок ; 5) ротація центральних точок на кут $-\varphi_i$ і розміщення їх на вхідному зображенні. На рис. 3 представлено функціональну схему побудови осей із візуалізацією проміжних етапів. Стандартизована позиція руки (у вертикальному положенні і по центру) не вимагається.

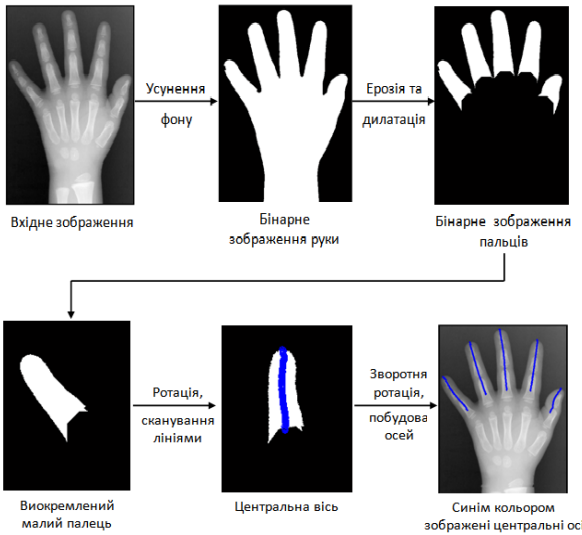


Рис. 3. Візуалізація етапів побудови центральних осей

Для усунення фону використано алгоритм [7], що базується на комбінації локальних гістограмних статистик зображення, операцій математичної морфології та пошуку компонент зв'язності. Цей алгоритм є робастним до шуму та нерівномірного освітлення, що робить ефективним його застосування для вирішення даної задачі. Після обробки отримуємо бінарне зображення руки *BW_hand*.

Другим кроком є виокремлення пальців. Це означає, що кожен з них повинен представлятися окремою компонентою зв'язності. Пропонується послідовно застосувати до *BW_hand* операції ерозії та дилатації [8]. Розміри структурного елементу підбираються експериментальним шляхом в залежності від

розмірності зображення. Структурний елемент має бути таким, щоб внаслідок ерозії на зображенні повністю зникли (трансформувалися у фон) ділянки пальців. Процедура одержання бінарного зображення пальців BW_fingers описується наступним псевдокодом:

BW_fingers = BW_hand - [imdilate(imerode(BW_hand))]. BW_hand*,

де imdilate та imerode – операції математичної морфології: дилатація та ерозія відповідно; операція (.*) – поелементне множення матриць.

Для кожної компоненти зв'язності зображення BW_fingers виконуються наступні операції. Утворюється бінарне зображення BW_component, на якому присутня лише одна компонента, її геометричне положення ідентичне як у відповідної компоненти на BW_fingers, а розмірності BW_component і BW_fingers одинакові. Після цього визначається кут повороту BW_component φ_i, при якому положення пальця повинно набувати максимально вертикального. Нами запропонований такий алгоритм знаходження кута φ_i: проводиться циклічна ротація зображення BW_component при $\varphi_i \in \left[-90^\circ; 90^\circ\right]$ із кроком в один градус; для кожного φ_i отримане зображення проектується на вісь Ox (якщо у векторі-стовпці є хоча б одне значення 1, то відповідний елемент на проекції буде 1, інакше – 0). Знаходиться сума елементів, що дорівнюють одиниці. Ця сума представлятиме ширину проекції пальця в пікселах. Очевидно, що мінімальна ширина буде ідентифікувати вертикальне положення пальця.

Після ротації зображення BW_component сканується горизонтальними лініями (крок – 1 піксел). На лініях, що перетинають силует пальця присутня послідовність ненульових елементів. Їх середина і обирається за центральну точку. Усі ці точки формують вектор, який ідентифікує вісь симетрії пальця.

Останнім етапом обробки в схемі рис. 3 є ротація центральних точок на кут $-\varphi_i$ і розміщення їх на вхідному зображенні *BW_hand*.

Розроблена технологія побудови центральних осей була реалізована в середовищі GNU Octave. Комп'ютерні експерименти проводилися, використовуючи спеціалізовану базу рентгенограм кисті руки (University of Southern California, http://www.ipilab.org/BAAweb/). Результати підтверджують, що технологія є робастною до положення руки в цілому, розмірів та орієнтації пальців на рентгенограмі.

Висновки

Розроблено і реалізовано технологію автоматичної побудови центральних осей пальців на рентгенограмі кисті руки. Завдяки робастності технології до розмірів та орієнтації пальців забезпечується точна локалізація осей. Це, в свою чергу, дозволить підвищити прецизійність комп'ютерних систем оцінки кісткового віку дітей. Щодо обмежень запропонованої технології, то вона вимагає присутність фону (хоча б мінімальну) між пальцями на зображенні. Ця умова ставиться також існуючими алгоритмами побудови осей. Практично усі рентгенограми із вищезазначеної бази задовольняють дану умову.

Автор щиро дякує за цінні поради науковому керівнику проф. Вороблю Р.А.

Список використаних джерел

9. Greulich W. Radiographic Atlas of Skeletal Developmentof Hand Wrist [2nd ed] / W. Greulich, S. Pyle. – Stanford, CA: Standford Univ. Press, 1971.
10. Tanner J. M. Assessment of Skeletal Maturity and Prediction of Adult Height / J. M. Tanner, R. H. Whitehouse. – London, U.K.: Academic, 1975.
11. Piętka E. Computer Automated Approach to the Extraction of Epiphyseal Regions in Hand Radiographs / E. Piętka and other // Journal of Digital Imaging. – 2001. – V. 14. – N. 4. – pp. 165–172.
12. Park K.H. Robust Epiphyseal Extraction Method Based on Horizontal Profile Analysis of Finger Images / K.H. Park, L.M. Lee, W.Y. Kim // Information Technology Convergence. – 2007. – p. 278–282.
13. Giordano D. Epiphysis and Metaphysis Extraction and Classification by Adaptive Thresholding and DoG Filtering for Automated Skeletal Bone Age Analysis / D. Giordano and other // Conf Proc IEEE Eng Med Biol Soc. – 2007. – p. 6551 – 6556.
14. Luis-García, R. A Fully Automatic Algorithm for Contour Detection of Bones in Hand Radiographs Using Active Contours / R. de Luis-García and other // Proc. of the IEEE International Conference on Image Processing, Barcelona, Spain, September. – 2003. – Part III. – p. 421-424.
15. Степанюк С. Усунення фону зі зображення у задачі оцінювання віку за рентгенограмою кисті руки / С. Степанюк // Матеріали III науково-технічної конференції „Обчислювальні методи і системи перетворення інформації". – 25–26 вересня 2014, Львів. – 2014. – с. 156–157.
16. Gonzalez R. C. Digital Image Processing / R. C. Gonzalez and R. E. Woods. – Prentice Hall, 2002. – 1050 p.

XI Міжнародна науково-практична конференція
XI Международная научно-практическая конференция
11th International Conference

ТЕОРЕТИЧНІ ТА ПРИКЛАДНІ АСПЕКТИ ПОБУДОВИ ПРОГРАМНИХ СИСТЕМ

ТЕОРЕТИЧЕСКИЕ И ПРАКТИЧЕСКИЕ АСПЕКТЫ ПОСТРОЕНИЯ ПРОГРАММНЫХ СИСТЕМ

THEORETICAL AND APPLIED ASPECT OF PROGRAM SYSTEMS DEVELOPMENT

TAAPSD'2014

Праці конференції
Труды конференции
Proceeding