УДК 004.4’232

В. Овсяк[1,2], Т. Маркушевський[2], Ю. Петрушка[1]
[1]Українська академія друкарства, Львів, Україна;
[2]Політехніка Опольська, Ополє, Польща

СИНТЕЗ МОДЕЛІ ПІДСИСТЕМИ ПОШУКУ СТЕЖОК ДОСТУПУ ДО УНІТЕРМІВ XML-ФОРМУЛ АЛГОРИТМІВ

Синтезовано абстрактний алгоритм і його модель для віднаходження стежок доступу до унітермів, описаних спеціалізованим xml-форматом формул алгоритмів.

Ключові слова: **модель, синтез, підсистема, функційний унітерм, унітерм, мінімізація, алгоритм.**

Synthesized abstract algorithm and model for finding paths access to uniterms of described specialized xml - format formulae of algorithms.

Keywords: model, synthesis, subsystem, functional uniterm, uniterm, minimization, algorithm.

Вступ

Алгоритми, описані формулами алгебри алгоритмів [1] або розширенням алгебри алгоритмів [2], на відміну від алгоритмів, описаних інтуїтивними методами [3–19], можуть, як звичайні математичні вирази, перетворюватися на основі властивостей операцій алгебри алгоритмів. Доцільність виконання перетворень полягає у можливості зменшення виразів, а в найкращому випадку – в отриманні виразів з мінімальною кількістю складових.

Мінімізація формул алгоритмів може виконуватися вручну та автоматично. Для автоматичного виконання тотожних перетворень необхідно синтезувати і запрограмувати модель системи мінімізації формул алгоритмів.

Формули алгоритмів набирають і редагують у комп’ютерній системі “Генкод” [20]. Створені у редакторі формули записуються у комп’ютерну пам'ять і базу алгоритмів. Для запису формул алгоритмів, які є графічно-текстовими об’єктами, створено спеціальний *xml*-подібний формат даних.

Автоматизація процесів перетворень формул алгоритмів можлива на підставі аналізу *xml*-файлів алгоритмів, отримуваних системою “Генкод”. Виконання такого аналізу необхідне передусім для пошуку в *xml*-файлі унітермів, які будуть мінімізуватись, а після цього, на підставі аналізу унітермів і операцій, які над ними виконують, та властивостей операцій виконання тотожних перетворень формул алгоритмів. У дослідженні розглянуто синтез моделі підсистеми автоматичного пошуку шляхів знаходження унітермів формул алгоритмів, описаних *xml*-форматом даних.

1. Декомпозиція абстрактної підсистеми

Підсистема автоматизованого пошуку повинна описуватися як алгоритм, який матиме програмну реалізацію. У зв’язку з тим, що модель підсистеми матиме багато деталей, до яких належать типи змінних і доступ до них, функційні унітерми з методами доступу, властивостями, вхідними і вихідними параметрами та їхніми типами, то для спрощення синтезу підсистеми доцільно спочатку синтезувати абстрактний алгоритм функціонування підсистеми, а після цього створити модель абстрактного алгоритму. Ця методологія запропонована у дослідженні [21] та успішно використана у роботах [22–25].

Введемо абстрактні складові підсистеми. Перш за все абстрактні змінні, які та параметри яких будуть описані в моделі, позначмо *Z*.

Вихідними даними для пошуку унітермів є формули алгоритмів, записані у пам’яті комп’ютера у вигляді *xml*-файла. Необхідно задавати адресу, під якою файл зберігається у пам’яті. Нехай абстрактним функційним унітермом задання адреси і зчитування *xml*-файла буде *A()*.

До початку виконання аналізу *xml*-файла його потрібно зчитати з пам'яті комп'ютера з бази алгоритмів. Позначимо абстрактний функційний унітерм зчитування формул алгоритмів з бази алгоритмів як *B()*.

Маючи *xml*-файл, потрібно вибрати з нього окремий рядок. Нехай він описує абстрактний унітерм *C()*.

У зчитаному рядку має бути ідентифікований унітерм формули алгоритму. Позначимо абстрактний функційний унітерм ідентифікації унітермів формул алгоритмів як *D()*.

Нарешті потрібно сформувати шлях доступу до унітерма формули алгоритму і зберегти його. Нехай це описується абстрактним функційним унітермом *E()*.

Отже, абстрактна підсистема пошуку шляхів доступу до унітермів формул алгоритмів містить всі описані вище абстрактні функційні унітерми й абстрактні змінні. Оскільки усі вони є складовими абстрактної підсистеми, то для її запису у вигляді формули використовуємо операцію секвентування. Розташування абстрактних змінних і функційних унітермів не має значення, тому що описуються тільки складові абстрактної підсистеми, а не черговість їх виконання. Враховуючи це, формула абстрактного алгоритму підсистеми *@Gi* матиме такий вигляд

$$@Gi=\left(\begin{array}{l} Z, \\ \left(\begin{array}{l} A(), \\ \left(\begin{array}{l} B(), \\ \left(\begin{array}{l} C(), \\ \left(\begin{array}{l} D(), \\ E(), \end{array}\right) \end{array}\right) \end{array}\right) \end{array}\right) \end{array}\right) \qquad (1)$$

де *@* – ідентифікатор підсистеми; *Gi* – назва підсистеми пошуку доступу до унітермів.

3. Синтез моделей абстрактних змінних і функційних унітермів

3.1. Моделі змінних

Перш за все створюємо моделі абстрактної змінної *Z*. Оскільки у пам'яті комп'ютера формули алгоритмів зберігаються у вигляді *xml*-файлів, то для збереження зчитаного *xml*-файла вводимо змінну *x* стандартного типу *xml*-документа, який позначимо *xmld*, а належність значень змінної до цього типу даних запишемо як $x\in@xmld$ (знак підкреслення типу даних означає, що цей тип даних є стандартним типом). Задаємо стандартний загальний метод доступу (*pu*) до цієї змінної. Враховуючи це, модель опису змінної набуде вигляду $pu\ x\in@xmld$.

Для того, щоб зберегти відшукані шляхи доступу до унітермів формул алгоритмів, вводимо одновимірну таблицю [] *t* ([] – ідентифікатор таблиці). Шлях доступу до унітермів формул алгоритмів є текстом, який належить до стандартного типу даних, котрий позначимо *Str*. Задаємо 1 000 рядків таблиці. У такому разі отримуємо такий опис таблиці $[]\ t\in@Str(1000)$.

Індексування рядків таблиці виконується змінною *y* загального доступу *(pu)* і стандартного типу *In*, що опишеться як $pu\ \ y\in@In$.

3.2. Моделі функційних унітермів

Абстрактному функційному унітермові задання адреси *A()* надаємо загальний метод доступу і вхідний параметр *s*, яким буде назва *xml*-файла, отримуємо опис його заголовка $pu\ A(s\in@Str)$. Функційними складовими вибираємо унітерм присвоєння значення змінній ($y=0$), яка використовується для індексування таблиці, і зчитування *xml*-файла формули алгоритму ($x.A(s)$), які запишемо під знаком операції секвентування з розділювачем кома

$$\underline{pu}\ A(s\in@\underline{Str})=\overbrace{y=0,\ A(s).}$$

Зчитування формули алгоритму з бази алгоритмів опишемо через вхідний параметр *z* типу *xmld* з присвоєнням змінній значення вхідного параметра

$$B(z\in@xmld)=(x=z).$$

У загальному випадку *xml*-файл має багато рядків. Встановлення кількості невкладених рядків *xml*-файла (*n*) і організація циклу за цією кількістю рядків ($\lhd(i\leq n)$) описується стандартним функційним унітермом *(n)Fore(x)*. Якщо наявний *i*-й рядок, його вибір описується стандартною властивістю *Cn*. Опис вибору рядка має вигляд *x(i).Cn*. Заголовок функційного унітерма є таким *pu*

C(q∈@xmln), де *q∈@xmln* – вхідний параметр *q* стандартного типу *xmln*. Його зміст описується елімінуванням циклічної формули і знака завершення виконання функційного унітерма (знак крапки). Умовою елімінування є наявність рядків (*n≠0)*. Циклічна формула утворена операцією циклічного секвентування за умовою *i≤(n)Fore(x))* (тобто не допускається перебільшення кількості рядків, які не є вкладеними) над секвентуванням вибору функційного унітерма *D(x(i).Cn)* і ознакою повернення у цикл (c_i) за кількістю рядків:

$$pu\ C(q \in @xmln) = \overbrace{\prec(i \le (n)Fore(q)); . ; (n \ne 0)\text{-}?}^{\overbrace{\left(\begin{array}{l} D(q(i).Cn); .; (i \le n)\text{ -}? \\ ; \\ c_i \end{array}\right.}}$$

Синтез моделі абстрактного функційного унітерма *D()* виконуємо *секвенційним методом* [1]. Як видно із останньої формули, цей функційний унітерм має один вхідний параметр, яким є вибраний рядок *q(i)*. Тому в його заголовку запишемо параметр *k* стандартного типу *xmln* та задамо стандартний метод доступу , що утворює вираз *pu D(k∈@xmln)*.

1. Синтез секвенцій. Перша секвенція утворена формулою аналізу формули у формулі алгоритму, позначимо її *S*. Друга секвенція – це завершення опису (.) функційного унітерма. 2. Синтез елімінувань. Секвенції елімінуємо за умовою наявності формули у формулі (*u*), отримуємо елімінування

$$\overbrace{S; . ; u\text{-}?}$$

Формула, яка є в іншій формулі, теж може містити у собі інші формули. Тому секвенція *S*, у загальному випадку, є складною циклічною формулою, до синтезу якої і перейдемо. 1. Синтез секвенцій. Першу секвенцію (S_1) утворюємо функційним унітермом відшукання стежки доступу до унітерма, що запишемо як *p∈@Pat=E(k(j).Cn)*, та унітерма заглиблення у поточну формулу, що описується поверненням до вибору функційного унітерма *D()*, який описується як *D(k(j).Cn)*. Тому

$$S_1 = \overbrace{p \in @Pat = E(k(j); D(k(j).Cn)}.$$

Другу секвенцію (S_2) утворюємо унітермом повернення у цикл (c_j) для входу у поточні вкладені формули. Унітермом завершення виконання (.) циклічної формули утворена третя секвенція (S_3).

2. Синтез елімінувань. Секвенції елімінуємо за умовою наявності унітерма формули (*k(j).Na≠"#text"-?*), що дає елімінування

$$E_1 = \overbrace{S_1; S_2; (k(j).Na \ne "\#text")\text{-}?}.$$

Одержане елімінування із секвенцією елімінуємо за умовою (*(j≤r)-?*) завершення циклу *≺(j≤(r)Fore(k))*. Отримуємо формулу

$$E_2 = \overbrace{E_1; S_3; (j \le r)\text{-}?}^{} = \overbrace{S_1; S_2; (k(j).Na \ne "\#text")\text{-}?; S_3; (j \le r)\text{-}?}^{} =$$

$$\overbrace{\overbrace{p \in @Pat = E(k(j); D(k(j).Cn)}; c_j ; (k(j).Na \ne "\#text")\text{-}?; .; (j \le r)\text{-}?}$$

Підставивши цей вираз у передостанню формулу і записавши заголовок та операцію циклу, одержимо

$$pu\ D(k \in @xmln) = \overbrace{\prec(j \le (r)Fore(k)); . ; (r \ne 0)\text{-}?}$$

$$\overbrace{\left(\begin{array}{l} p \in @Pat = E(k(j).Cn); c_i ; k(j).Na \ne "text"\text{-}? ; . ; (j \le r)\text{ -}? \\ ; \\ D(k(j).Cn) , \end{array}\right.}$$

що і є моделлю абстрактного функційного унітерма *D()*.

Заголовок моделі абстрактного функційного унітерма *E()*, формування шляхів доступу до унітермів формул алгоритмів і збереження їх здійснюємо за загальним методом доступу *pu*, одним вхідним параметром *v* типу *xmln*, як видно з останньої формули, та одним вихідним параметром *w* стандартного типу *Str*. Отримуємо

$$pu\ \ (w \in @Str)E(v \in @xmln).$$

У зміст унітерма вводимо дії з формування початкового значення вихідної змінної *w=”/”+v.Na* і циклу для задання її повного значення, зумовленого вкладеннями

$$\begin{array}{l} \subset(v.Pn.Na \neq \text{”\#document”}) \\ \left(\begin{array}{l} w=\text{”/”}+v.Pn.Na+w \\ \ \ ; \\ v=v.Pn\ , \end{array}\right. \end{array}$$

де *Pn* – стандартна властивість вибору ключового слова *xml*-файла.

Остаточне формування стежки доступу до унітермів формул алгоритмів завершується дописуванням з лівого боку похилої риски до значення вихідної змінної *w=”/”+ w*.

Перед тим як записати сформовану стежку доступу у таблицю, необхідно перевірити, чи у *xml*-файлі ключовим словом вже є слово “*uniterm*”, яке підтверджене стандартною властивістю *Hcn*. Після цього виконують запис встановленої стежки до таблиці. Ці останні дії опишемо такою формулою

$$\left(\begin{array}{l} t_y=w;\ *;\ (!w.Cont(\text{“uniterm”})\&v.Hcn\text{-}? \\ \ \ ; \\ y=y+1, \end{array}\right)$$

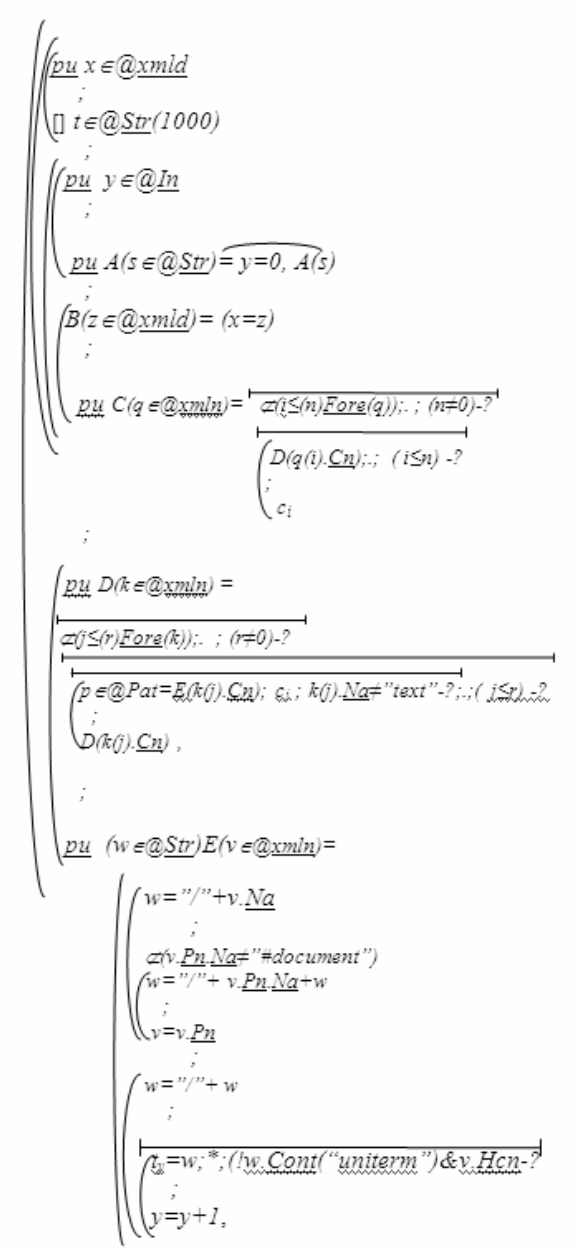
де *w.Cont(“uniterm”)* – стандартний функційний унітерм порівняння значення параметра зі значенням змінної , *!* – оператор зміни на протилежне значення логічної змінної.

З отриманих останніх формул утворюємо загальний опис моделі функційного унітерма формування і запису шляхів доступу до унітермів формул алгоритмів

$$\begin{array}{l} pu\ \ (w \in @Str)E(v \in @xmln)= \\ \left(\begin{array}{l} \left(\begin{array}{l} w=\text{”/”}+v.Na \\ \ \ ; \\ \subset(v.Pn.Na \neq \text{”\#document”}) \\ \left(\begin{array}{l} w=\text{”/”}+v.Pn.Na+w \\ \ \ ; \\ v=v.Pn \end{array}\right. \\ \ \ ; \\ w=\text{”/”}+w \\ \ \ ; \\ \left(\begin{array}{l} t_y=w;\ *;\ (!w.Cont(\text{“uniterm”})\&v.Hcn\text{-}? \\ \ \ ; \\ y=y+1, \end{array}\right) \end{array}\right) \end{array}\right) \end{array}$$

3. Модель абстрактної підсистеми

Підставивши у формулу (1) замість абстрактних функційних унітермів їхні моделі, отримуємо модель абстрактної підсистеми встановлення шляхів доступу до унітермів формул алгоритмів, яка описується такою формулою:



Висновки

Декомпозиція підсистеми на абстрактні функційні унітерми зменшує складність проектування підсистеми.

Створення абстрактних функційних унітермів зменшує складність розроблення їхніх моделей.

Синтезована модель абстрактної підсистеми описує процеси пошуку доступу до унітермів формул алгоритмів, які подані *xml*-файлами, утвореними комп'ютерною системою "Генкод".

Визначені шляхи доступу до унітермів формул алгоритмів можна використовувати для комп'ютерної мінімізації формул алгоритмів.

1. Owsiak W., Owsiak A., Owsiak J. Teoria algorytmów abstrakcyjnych i modelowanie matematyczne systemów informacyjnych. – Opole: Politechnika opolska, 2005. – 275 s. 2. Owsiak W., Owsiak A. Rozszerzenie algebry algorytmów / Pomiary, automatyka, kontrola. – № 2, 2010. – S. 184–188. 3. Kleene S.C. Origins of recursive function theory. Annals of the Theory of Computing, vol. 3, No. 1, Jan. 1981, pp. 52–67. 4. Turing A.M. On computable numbers, with an application to the Entscheidungsproblem // Proceedings of London Mathematical Society, series 2, vol. 42 (1936-1937), pp.

230. 5. Kleene S.C. Turing's analysis of computability, and major applications of it. In Rolf Herken, Editor, The universal Turing machine: A half-century story, Oxford University Press, 1988, pp. 17-54. 6. Post E.L Finite Combinatory Processes – Formulation 1 // Journal of Symbolic Logic, 1, pp. 103-105, 1936. Reprinted in The Undecidable, pp. 289ff. 7. De Mol L. Closing the circle: an analysis of Emil Post's early work. The Bulletin of Symbolic Logic, Vol. 12, Issue 02, June 2006, pp. 267 — 289. 8. Колмогоров А.Н. О понятии алгоритма // УМН. – Т. 8, Вып. 4(56), 1953. С. 175-176; translated into English in Uspensky V.A., Semenov A.L.: Algorithms: Main Ideas and Applications, Kluwer, 1993. 9. Колмогоров А.Н., Успенский В.А. О дефиниции алгоритма // УМН. – Т 13, вып. 4, 1958. – С. 3-28; translated into English in AMS Translations 29 (1963), pp. 217-245. 10. Gurevich Y. Kolmogorov machines and related issues. In G. Rozenberg and A. Salomaa, Editors, Current Trends in Theoretical Computer Science, World Scientific, 1993, pp. 225-234; originally in Bull. EATCS 35 (1988). 11. Schönhage A. Universelle Turing Speicherung. In J. Dörr and G. Hotz, Editors, Automatentheorie und Formale Sprachen, Bibliogr. Institut, Mannheim, 1970, pp. 369-383. 12. Schönhage A. Storage modification machines // SIAM Journal on Computing, 9 (1980), pp. 490-508. 13. Марков А.А. Теория алгоритмов // Труды МИАН. – Т.38, 1951. – С. 176–189; translated into English in American Mathematical Society Translations, 1960, series 2, 15, pp. 1–14. 14. Markov A.A., Nagorny N.M. The Theory of Algorithms (Mathematics and its Applications). Springer, 2001. 15. Church A. An unsolvable problem of elementary number theory // American Journal of Mathematics, vol. 58 (1936), pp. 345–363. 16. Constable R.L., Smith S.F. Computational foundations of basic recursive function theory // Theoretical Computer Science, 121, pp. 89-112, Dec. 1993. 17. Sieg W. Step by recursive step: Church's analysis of effective calculability // The Bulletin of Symbolic Logic, 3:2 (1997), pp. 154–180. 18. Aho A.V, Hopcroft J.E, Ullman J.D. The design and analysis of computer algorithms. Addison-Wesley Publishing Company, 1974. 19. Криницкий А.Н. Алгоритмы вокруг нас. – М.: Мир, 1988; also translated to Spanish (Algoritmos a nuestro alrededor). 20. Овсяк О. Класи інформаційної системи генерування коду / О. Овсяк // Вісник Тернопільського державного технічного університету: "Тернопільський національний технічний університет імені Івана Пулюя". – № 1, 2010. – С. 171–176. 21. Овсяк В.К. Методи підвищення ефективності математичного моделювання алгоритмів інформаційно-технологічних систем: автореф. дис. на здобуття наук. ступеня док. тех. наук: спец. 01.13.02 "Математичне моделювання в наукових дослідженнях" / Овсяк В.К. – Львів, 1996. – 45 с. 22. Бритковський В.М. Моделювання редактора формул секвенційних алгоритмів: автореф. дис. на здобуття наук. ступеня канд. тех. наук: спец. 01.05.02 "Математичне моделювання та обчислювальні методи" / Бритковський В.М. – Львів, 2003. – 18 с. 23. Василюк А.С. Підвищення ефективності математичного і програмного забезпечення редактора формул алгоритмів: автореф. дис. на здобуття наук. ступеня канд. тех. наук: спец. 01.05.02 "Математичне та програмне забезпечення обчислювальних машин і систем" / Василюк А.С. – Львів, 2008. – 20 с. 24. Назаркевич М.А. Синтез, моделі та моделювання алгоритмів мікропроцесорної системи керування електроприводом друкарських машин: автореф. дис. на здобуття наук. ступеня канд. тех. наук: спец. 01.05.02 "Математичне моделювання та обчислювальні методи" / Назаркевич М.А. – Львів, 2000. – 19 с. 25. Овсяк О.В. Моделювання транслятора структур даних електромеханічних схем друкарських машин: автореф. дис. на здобуття наук. ступеня канд. тех. наук: спец. 01.05.02 "Математичне моделювання та обчислювальні методи" / Овсяк О.В. – Львів, 2002. – 18 с.