УДК 004.272

А. О. Мельник, д. т. н., проф.;
І. Д. Яковлєва;
В. Ю. Ющенко, студ.

ПОБУДОВА ТА МАТРИЧНЕ ПОДАННЯ ПОТОКОВОГО ГРАФА АЛГОРИТМУ

Запропоновано підхід перетворення графічного подання графа алгоритму в потокову форму (ПГА), відображення ПГА в структурну матрицю, а також здійснення оберненої процедури відображення ПГА на основі структурної матриці.

Вступ

Граф алгоритму є основою для вивчення інформаційної структури алгоритмів [1, 2]. Але для перетворення його в дієвий інструмент, він має бути представлений у формі, зручній для проведення як теоретичних досліджень, так і практичних розрахунків.

Щоб оцінити обчислювальні і структурні характеристики алгоритму, його подають у вигляді потокового графа (ПГА) [1]. Побудова ПГА означає розподіл всіх його вершин по ярусах таким чином, що в *i*-му ярусі розміщені тільки функціональні оператори (ФО), які залежать від ФО попередніх ярусів і не залежать від ФО подальших ярусів. В межах одного ярусу між вершинами потокового графу немає зв'язків. Така форма подання алгоритму визначає ступінь паралелізму графа (максимальна кількість вершин на одному ярусі), а також мінімально можливий час обчислення даного алгоритму (кількість ярусів). Тому актуальним є питання автоматичного представлення графа алгоритму (ГА) у вигляді ПГА та «створення математичного апарату» [2] для його подальшого дослідження.

В роботі запропонований метод переходу від графічного подання ГА до ПГА, поданого у вигляді структурної матриці, а також обернена процедура графічного відображення ПГА із структурної матриці.

Огляд літератури

В роботі [3] запропоновано розміщувати опис графа в текстових файлах, структура яких описана в [4]. Проте, за потреби опрацювання ПГА, виникають певні проблеми, пов'язані з ускладненням графів зі збільшенням кількості операцій, а також необхідністю виділення великих об'ємів пам'яті для їх зберігання. Логічним виглядає подання ПГА матрицями.

В роботі [5] запропоновано заносити інформацію про ПГА в структурну матрицю, кількість стовпців якої дорівнює кількості дуг, по яких надходять операнди в найширшому ярусі, а кількість рядків дорівнює загальній кількості ярусів. Дуги в найширшому ярусі нумеруються від 1 до *n*, їх кількість на кожному ярусі є незмінною та дорівнює кількості вхідних та вихідних вершин, з яких (на які) надходять дані. Елементами такої матриці є номери ФО, які присвоєні кожній дузі, по якій на ФО надходять операнди. Структурна матриця зберігає обчислювальні і структурні характеристики ПГА такі як набір ФО, організацію з'єднань між ними, розподіл ФО по ярусах та інші. Ці характеристики об'єднані в множину $S(C, L, K, W)$, де $C = \{n_j\}$ — множина вхідних дуг ПГА, $j = \overline{1,n}$ — номер вхідної дуги ПГА, по яких надходять дані, n — кількість вхідних дуг ПГА; $L = \{l_i\}$, $i = \overline{1,l}$ — номер ярусу ПГА, l — кількість ярусів ПГА; $K = \{f_{ij}\} = \{0..k\}$ — множина функціональних операторів ПГА, $k \in N$, $i = \overline{1,l}$, $j = \overline{1,n}$ — номер функціонального оператора ПГА, $W = \{w_i, w\}$ — множина, яка визначає кількість ФО на кожному *i*-му ярусі і ширину ПГА — w. В цій же роботі [5] запропоновано заносити інформацію про ПГА в матрицю, кількість стовпців якої рівна кількості дуг, по яких надходять операнди в найширшому ярусі, кількість рядків рівна загальній кількості ярусів. Дуги в найширшому ярусі нумеруються від 1 до *n*, їх кількість на кожному ярусі є незмінною та дорівнює кількості вхідних та вихідних вершин, з яких (на які) надходять дані. Елементами такої матриці є номера ФО, які присвоєні кожній дузі, по якій на ФО над-

© А. О. Мельник, І. Д. Яковлєва, В. Ю. Ющенко, 2009

ходять операнди. Зважаючи на те, що більшість операцій є двомісними, а результат виконання операції часто необхідний для виконання більше як однієї наступної операції, тут використані ФО, які мають дві вхідні і дві вихідні дуги. Структурна матриця F розмірністю $l \times n$, елементами якої є номери ФО, присвоєні кожній дузі, по якій на ФО надходять операнди, має такий вигляд:

$$F = \begin{pmatrix} f_{11} & f_{12} & \cdots & f_{1n} \\ f_{21} & f_{22} & \cdots & f_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & f_{ij} & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ f_{l1} & f_{l2} & \cdots & f_{\ln} \end{pmatrix}, \quad (1)$$

де $\forall f_{ij} \in N$, $i = \overline{1,l}$, $j = \overline{1,n}$, l — кількість ярусів (рядків), n — загальна кількість дуг найширшого ярусу, по яких надходять операнди (стовпців).

Заповнюється структурна матриця F за правилом: на перетині i-го рядку і j-го стовпця ставиться номер відповідного ФО i-го ярусу, який виконує операцію над операндом, який надходить j-ю дугою ПГА $f_{ij} = k$, де k — належить множині натуральних чисел $k \in N$. Нумерація починається від 1 і призначається всім наступним ФО підряд в порядку зростання. Решта елементів такої матриці заповнюються нулями. Може бути два типи ФО: операційні, які передбачають виконання операцій, і перепускні, які передбачають передачу інформації з входу на вихід. Операційні ФО позначаються k, а перепускні — 0. Оскільки один ФО виконується над двома даними, які надходять різними дугами, то два елементи структурної матриці F i-го рядка приймають одне й те саме значення k. Якщо $f_{ij} = 0$, то в i-му ярусі над даними, які надходять j-тою дугою, не виконується жоден операційний ФО, а дані просто передаються на наступний ярус.

Постановка задачі

Оскільки для зберігання графа в комп'ютері і подальшого його опрацювання зручно використовувати матричний спосіб, а велике розповсюдження і наочність отримав графічний спосіб запису алгоритмів, то виникає необхідність розробки методу переходу між графічним і матричним представленням алгоритмів. Використання структурної матриці для опису ПГА є зручним для зберігання і дослідження. Тому постало питання розробки методу однозначного переходу від графічного подання ГА до ПГА із записом ПГА у вигляді структурної матриці, а також оберненої процедури відображення ПГА із структурної матриці для верифікації ГА.

1. Опис графа алгоритму

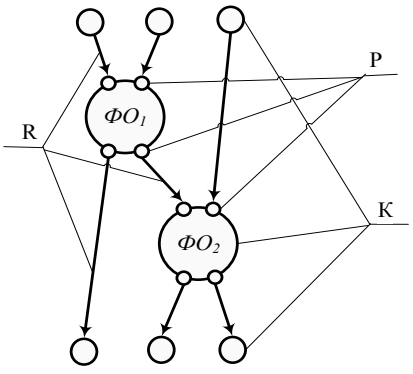
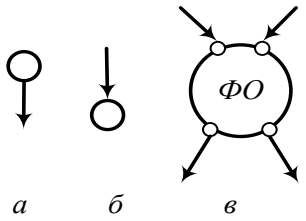


Рис. 1. Граф алгоритму $G(K, P, R)$; де K — множина вершин ГА, P — множина портів, R — множина дуг ГА

Для автоматичного зображення та дослідження графа алгоритму G необхідно точно описати множини вершин K та дуг R графа алгоритму. Опис вершин є досить тривіальною задачею, для якої добре розроблений математичний апарат. Значно складніше «подобраться к пониманию того, в какой форме должны описываться дуги» [2]. В роботі для опису дуг запропоновано логічно пов'язувати їх із вершинами на які вони надходять і з яких виходять. Для цього введена додаткова множина — порти вершин. Тоді граф описується трійкою $G(K, P, R)$ (рис. 1), де K — множина вершин ГА, P — множина портів, R — множина дуг ГА.

Вершини можуть бути вхідними (рис. 2а), вихідними (рис. 2б) і вершинами-ФО (рис. 2в). Даний параметр визначає тип вершин: $T_k = (0,1,2)$: 0 — якщо це є вхідна вершина, тобто на неї не надходить жодна дуга; 1 — якщо це є вихідна — кінцева вершина, тобто із неї не виходить жодна дуга; 2 — всі решта вершин.

Всі вершини отримують ідентифікатори (порядкові номери) в порядку створення. Ідентифікатори утворюють множину ідентифікаторів I_k, де, $\{I_k\}$ набуває значення із множини натуральних чисел і $k = \overline{1, N_f}$, N_f— кількість вершин ГА. Таким чином, вершини характеризуються параметрами $K(I_k, T_k)$, де I_k — множина ідентифікаторів; $T_k = (0, 1, 2)$ — тип вершин.



Вершина	
ідентифікатор вершини, *Ik*	тип вершини, *Tk*

г

Рис. 2. Типи вершин ГА: а — вхідна вершина (тип 0); б — вихідна вершина (тип 1); в — вершина-ФО (тип 2); г— опис вершини

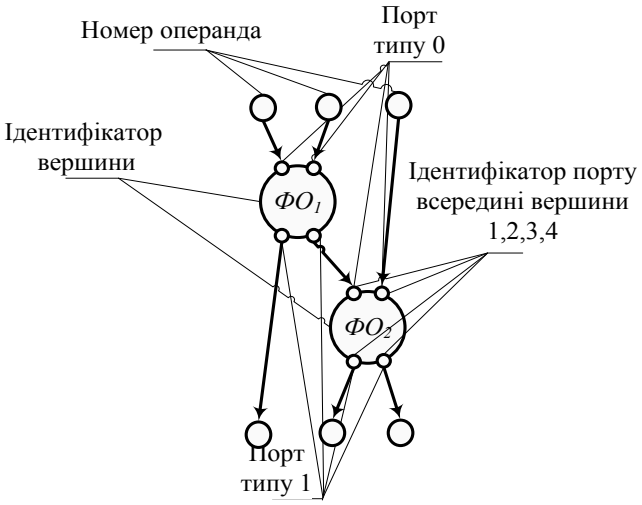
В свою чергу, вершини мають порти, кожен з яких характеризується множиною характеристик $P(I_1, C, I_2, T_p)$ (рис. 3). На рис 3. використані позначення:

1) I_1 — ідентифікатор порту. Ідентифікатор порту отримується в результаті конкатенації двох складових:
I_k — ідентифікатору вершини, якій належить даний порт;
I_p — порядковий номер порту всередині вершини $I_p = (1, 2, 3, 4)$;

2) j — номер операнда, який надходить в даний порт, $j = \overline{1, n} \in C$;

3) I_2 — ідентифікатор другого порту, який з'єднаний з даним портом дугою;

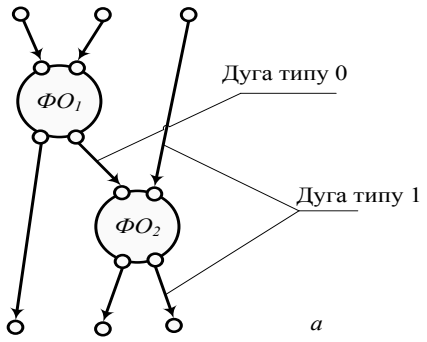
4) T_p — тип порту. Порти можуть бути двох типів, які відповідно позначаються. $T_p = (0, 1)$, порту привласнюється 0, якщо це вхідний порт і 1 — якщо вихідний.



Порт				
ідентифікатор порту, I_1		номер операнда, який надходить в даний порт, *j*	ідентифікатор порту, який з'єднаний з даним портом дугою, I_2	тип порту, *Tp*
ідентифікатор вершини, I*k*	порядковий номер порту всередині вершини, *Ip*			

Рис. 3. Опис портів

Дуги (рис. 4) бувають двох типів $T_r = (0, 1)$ і, відповідно, приймають два значення 0 і 1*:* 0 — якщо дуга з'єднує два ФО; і 1, якщо дуга з'єднує ФО з вхідною або вихідною вершиною. Дуга має такі параметри: 1) початок дуги (відповідає ідентифікатору вершини ГА, з якої виходить дуга), 2) кінець дуги (відповідає ідентифікатору вершини ГА, на яку надходить дуга), 3) початковий порт — ідентифікатор порту з якого виходить дуга, 4) кінцевий порт — ідентифікатор порту на який надходить дуга.



Дуга				
тип дуги, *Tr*	Початок дуги - ідентифікатор порту, I_1		Кінець дуги - ідентифікатор порту, I_2	
	ідентифікатор вершини, I1	порядковий номер порту всередині вершини, *Ip*	ідентифікатор вершини, I2	порядковий номер порту всередині вершини, *Ip*

б

Рис. 4. Опис дуг ГА: а — граф; б — таблиця

В загальному, для опису ГА необхідна інформація стосовно шести параметрів елементів ГА, яка в різних комбінаціях надає повну інформацію про структуру графа алгоритму (табл.).

Необхідні параметри елементів графа для опису ГА

Елементи графа	Параметри елементів графа					
	тип вершини, *Tk*	тип порту, *Tp*	тип дуги, *Tr*	ідентифікатор порту, Ip		номер операнда, який надходить в цей порт, *j*
				ідентифікатор вершини, *Ik*	порядковий номер порту всередині вершини, *Ip*	
вершина	+			+		
порт		+		+	+	+
дуга			+	+	+	

Примітка. Знаком + в таблиці позначено які параметри необхідні для опису кожного елементу графа.

Дана інформація про ГА є достатньою для його опису, зберігання і відображення, але не дає інформації про можливості його розпаралелювання.

2. Побудова ПГА та заповнення структурної матриці на основі графічно заданого ГА

Для забезпечення дослідження структурних характеристик із графічного подання графа алгоритму передбачено перетворення його в ПГА, відображення ПГА в структурну матрицю а також здійснення оберненої процедури відображення ПГА на основі структурної матриці.

В загальному, формальний опис ПГА такий: спочатку на основі послідовної програми будується граф алгоритму, потім вивчаються його паралельні форми і виконується їх формальний математичний опис. В цій роботі на основі аналізу ГА, представленого графічно і описаного, як показано вище, одночасно виконується побудова ярусу ПГА та заповнення рядка структурної матриці.

Для реалізації поставленої задачі розроблені і реалізовані такі алгоритми.

Алгоритм побудови та заповнення структурної матриці ПГА із заданою шириною на основі заданого графічно ГА

Для побудови ПГА із ГА введемо додаткову властивість вершини. $U_k = (1, 0)$ і, відповідно, приймає значення: 1 — якщо вершина «використана», тобто для неї готові операнди і вона розміщена в ярусі ПГА; і 0 — в інших випадках.

Крок 1. Встановлюємо $i = 1$, $U\_cnt = 0$, де i — поточний ярус ГА та рядок структурної матриці, U_cnt — лічильник кількості використаних вершин.

Крок 2. Встановлюємо $k = 1$, де k — поточна вершина.

Крок 3. Встановлюємо $l = 0$, де l — кількість елементів масиву T, $T = (T_l)$ — список вершин-кандидатів на поточний ярус ПГА та поточний рядок структурної матриці.

Крок 4. За *алгоритмом* 1 визначаємо готовність даних для вершини P_k. Якщо результат — *true*, то переходимо до кроку 5, інакше переходимо до кроку 6.

Крок 5. $l = l + 1$. Вершину k заносимо в масив вершин-кандидатів T_l.

Крок 6. $k = k + 1$. Якщо $k <= N_f$, то переходимо до кроку 4, інакше — до кроку 7.

Крок 7. Якщо $l > w$, то переходимо до кроку 8, інакше — до кроку 9 (w — ширина ПГА. Якщо вона не задана користувачем, то встановлюється максимальна, і для ФО, які мають два вхідні порти дорівнюватиме $w = n/2$)

Крок 8. Встановлюємо ширину ПГА: $l = w$.

Крок 9. Встановлюємо $k = 1$ для перегляду масиву T.

Крок 10. $j = T_l$; $U_j = true$. $k = k + 1$.

Крок 11. Якщо $k <= l$, то переходимо до кроку 10, інакше — до кроку 12.

Крок 12. Загальна кількість використаних вершин $U\_cnt = U\_cnt + l$;

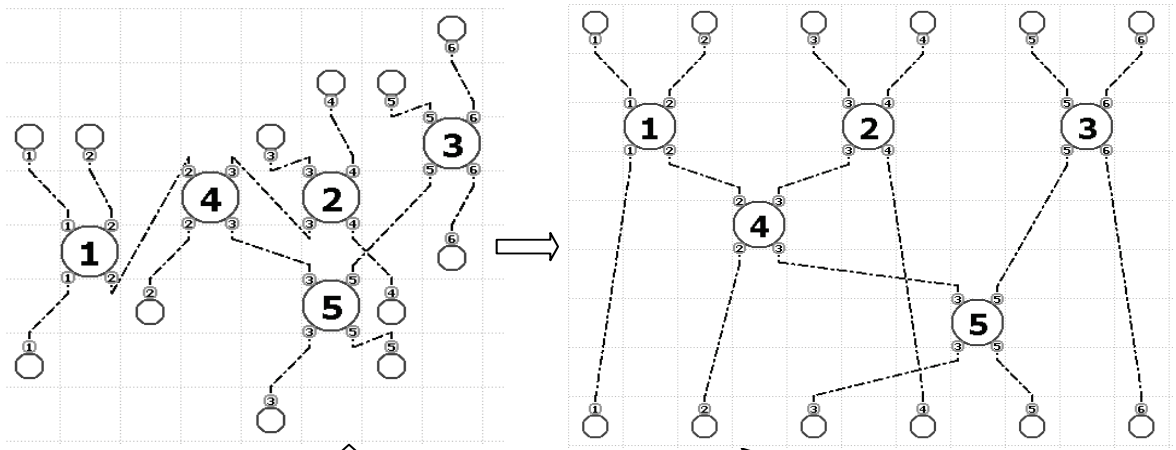
Крок 13. За *алгоритмом 2* із списку вершин-кандидатів T заповнюємо i-й рядок структурної матриці.

Крок 14. Якщо $U\_cnt = N_f$, то переходимо до кроку 15, інакше $i = i + 1$ та переходимо до кроку 2.

Крок 15. Структурна матриця заповнена, вона складається з i рядків.

Крок 16. Кінець алгоритму.

Результат виконання алгоритму для ГА, заданого графічно (рис. 5а) зображено на екранних формах програми (рис. 5): ПГА — рис. 5б; структурна матриця — рис. 5в.



$$F = \begin{pmatrix} 1 & 1 & 2 & 2 & 3 & 3 \\ 0 & 4 & 4 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 5 & 0 \end{pmatrix}$$

в

Рис. 5. Автоматична побудова ПГА із ГА, заданого графічно та заповнення структурної матриці:
а — ГА; б — ПГА; в — структурна матриця

Алгоритм 1. Алгоритм визначення готовності даних для поточної вершини

Даний алгоритм для побудови ярусу ПГА визначає, чи є в наявності всі необхідні вхідні операнди для виконання операції, яка задана вершиною ГА і, якщо, вона була поміщена в поточний ярус ПГА, то дана вершина позначається, як «використана». Також позначаються, як «використані» всі порти, що належать цій вершині. Кількість портів для кожної вершини — *m*. Результат виконання алгоритму — *true* або *false* (за замовчуванням).

Встановлюємо i = 1, res = true.

Крок 1. Якщо дана вершина використана, тобто *Uk = true*., то *res = false* і переходимо до кроку 6, інакше переходимо до кроку 3.

Крок 2. Розглядаємо *i*-й порт даної вершини. Якщо цей порт має тип 0 (вхідний порт), то переходимо до кроку 4, інакше — до кроку 5.

Крок 3. Якщо ідентифікатор порту, який з'єднаний з цим портом дугою, позначений як використаний або тип дуги для цього порту має тип 1 (дуга з'єднує ФО з вхідною або вихідною вершиною), то переходимо до кроку 5, інакше *res = false* і переходимо до кроку 6.

Крок 4. $i = i + 1$. Якщо $i > m$, то переходимо до кроку 6, інакше — до кроку 3.

Крок 5. Якщо *res = true*, то для даної вершини є в наявності всі вхідні дані і вона може бути розміщена в ярусі ПГА, інакше — ні. Кінець алгоритму.

Заповнюється структурна матриця *F* за правилом: на перетині *i*-го рядку і *j*-го стовпця ставиться номер відповідного ФО *i*-го ярусу, який виконує операцію над операндом, який надходить *j*-ю дугою ПГА $f_{ij} = k$, де *k* — належить множині натуральних чисел $k \in N$. Заповнення рядка структурної матриці відбувається за таким алгоритмом.

Алгоритм 2. Заповнення рядка структурної матриці відповідно до списку вершин в поточному ярусі

Масив з *n* вершин, які складають поточний ярус графа. В кожної вершини є *m* портів. Позначимо номер *k*-ї вершини як n_k, а *j*-й елемент рядка структурної матриці — f_j.

Крок 1. Встановлюємо $k = 1$, $z = 1, z = \overline{1, m}$, z — лічильник портів.

Крок 2. Розглядаємо z-й порт k-ї вершини. Якщо номер операнда, який надходить в даний порт — j, тоді $f_j = n_k$ і переходимо до кроку 3.

Крок 3. Перехід на наступний порт вершини: $z = z + 1$. Якщо $z > m$, то перейти до кроку 4, інакше — до кроку 2.

Крок 4. Перехід на наступну вершину: $k = k + 1$. Якщо $k > w$, то перейти до кроку 5, інакше — до кроку 2.

Крок 5. Рядок структурної матриці заповнено. Кінець алгоритму.

3. Відображення ПГА на основі структурної матриці

Відображення ПГА здійснюється на основі однократного аналізу структурної матриці. При зображенні ПГА передбачено відображення вершин, з яких надходять вхідні дані на ФО та на які надходить результат виконання всіх операцій. Кількість вхідних, вихідних вершин і дуг в одному ярусі однакова і дорівнює кількості стовпців структурної матриці — n. Кількість вершин, які відповідають ФО, дорівнює загальній кількості значущих елементів ПГА. Кожна k-та вершина, яка відповідає ФО, відображається в i-му ярусі, якщо між елементами i-го рядка структурної матриці виконується умова $f_{ij} = f_{iz} = k$. Одна вхідна дуга, яка надходить до k-го ФО зображується від j-ї дуги попереднього ярусу (або j-ї вхідної вершини, якщо це перший ярус) до k-го ФО, друга дуга — від z-ї дуги попереднього ярусу (або z-ї вхідної вершини) до k-го ФО. Оскільки кожен ФО повертає результат виконання на ті самі номери дуг, по яких були отримані операнди, то вихідні дуги для k-го ФО будуть тими самими, як і вхідні, і визначаються по другому індексу елементів матриці, які задовольняють умові $f_{ij} = f_{iz} = k$. Якщо елемент структурної матриці приймає нульове значення $f_{ij} = 0$, то j-та дуга не з'єднується з жодною вершиною i-го ярусу, а просто продовжується до j-ї дуги наступного ярусу. Така процедура повторюється до тих пір поки не будуть перебрані всі елементи f_{ij} структурної матриці F. Дуги ФО останнього ярусу з'єднуються за тими ж правилами з вихідними вершинами. Таке відображення структурної матриці у ПГА дає можливість проводити верифікацію структурних матриць, отриманих із послідовних програм і забезпечувати графічне подання алгоритмів.

Алгоритм відображення структурної матриці в ПГА

Введемо позначення: F — структурна матриця розмірністю $n \times m$; D — вектор використання вхідних даних вершинами ФО довжиною m; кожний елемент D [j] цього вектора містить номер вершини, яка останньою виконувала операцію, які отримані j-ї вхідної вершини (на початку роботи вектор містить номера вхідних вершин); K — множина, яка складається з номерів вершин, які були графічно зображені в процесі роботи.

Крок 1. Присвоїти $i = 0$ і перейти до кроку 2.

Крок 2. $i = i + 1$. D [i] = 0 і перейти до кроку 3.

Крок 3. Якщо $i < m$, то перейти до кроку 2, інакше перейти до кроку 4.

Крок 4. Присвоїти $i = 1$, $j = 1$ і перейти до кроку 5.

Крок 5. Якщо F [i, j] = 0, то перейти на крок 9, інакше — до кроку 6.

Крок 6. Якщо F [i, j] належить множині K (тобто вершина зображена), то перейти до кроку 8, інакше — до кроку 7.

Крок 7. Добавити F [i, j] в множину K. Відобразити вершину з номером F [i, j] і перейти до кроку 8.

Крок 8. Відобразити дугу, що з'єднує порт I_p вершини D [j] з портом I_p вершини F [i, j]. D [j] = F [i, j] (I_p — визначається функцією, яка повертає перший невикористаний порт для даної вершини: для вершини D [j] — порт типу 1, а для F [i, j] — 0); і перейти до кроку 9.

Крок 9. $j = j + 1$. Якщо $j <= m$, то перейти до кроку 5, інакше — до кроку 10.

Крок 10. $j = 1$, $i = i + 1$. Якщо $i <= n$ перейти до кроку 5, інакше — до кроку 11.

Крок 11. Кінець алгоритму.

ПГА (рис. 5б), структурна матриця якого зображена на рис. 5в, повністю співпадає з ПГА, отриманим із ГА, поданого графічно (рис. 5а).

Висновки

1. Запропоновано метод та алгоритми формального опису за допомогою структурної матриці потокових графів інваріантних до зсуву алгоритмів на основі аналізу графічного подання ГА. Метод дозволяє одразу переходити від ГА до формального опису ПГА без його побудови або паралельно із побудовою.

2. Розроблені і реалізовані алгоритми:

— Побудови ПГА та заповнення структурної матриці на основі графічно заданого ГА, які визначають розподіл всіх вершин графа алгоритму по ярусах таким чином, що в *i*-му ярусі розміщені тільки ФО, які залежать від ФО попередніх ярусів і не залежать від ФО наступних ярусів; в межах одного ярусу між вершинами потокового графу немає зв'язків.

— Відображення ПГА на основі структурної матриці, яке здійснюється на основі однократного аналізу структурної матриці.

3. Описані алгоритми виконані як програмний модуль і реалізовані мовою С++ та є частиною проекту, яка стосується застосування структурної матриці для САПР спеціалізованих процесорів.

СПИСОК ЛІТЕРАТУРИ

1. Мельник А. О. Спеціалізовані комп'ютерні системи реального часу / А. О. Мельник — Львів : НУ «Львівська політехніка», 2002. — 60 с.

2. Воеводин В. В. Вычислительная математика и структура алгоритмов / В. В. Воеводин. — М. : Изд-во МГУ, 2006. — 112 с.

3. Сальников А. Н. Прототип системы разработки приложений и автоматического распараллеливания программ для гетерогенных многопроцессорных систем / А. Н. Сальников, А. Н. Сазонов, М. В. Карев // Вопросы Атомной Науки и Техники /Математическое моделирование физических процессов. — ВНИИЭФ, 2003. — № 1. — С. 61—68.

4. Сальников А. Н. Некоторые технические аспекты инструментальной системы для динамической балансировки загрузки процессоров и каналов связи / А. Н. Сальников // Програм. системы и инструменты: Темат. сб. тр. фак. вычисл. математики и кибернетики МГУ. — 2002. — № 3. — С. 152—164.

5. Мельник А. О. Подання потокового графа алгоритму структурною матрицею / А. О. Мельник, І. Д. Яковлєва // Вісник Хмельницького національного університету. — 2008. — № 4. — С. 124—129.

Рекомендована кафедрою комп'ютерних систем управління

Надійшла до редакції 21.10.08
Рекомендована до друку 20.11.08

Мельник Анатолій Олексійович — завідувач кафедри електронно-обчислювальних машин.

Національний університет «Львівська політехніка»;

Яковлєва Інна Дмитрівна — асистентка кафедри комп'ютерних систем та мереж; ***Ющенко Василь Юрієвич*** — студент.

Чернівецький національний університет імені Юрія Федьковича