УДК 681.62:655

В. В. Грицик

Опис алгоритмів паралельно-рекурсивної обробки даних в системах реального часу

(Представлено членом-кореспондентом НАН України В. В. Грициком)

Проаналізовано сучасний стан проблеми складних задач комп'ютерного зору для реалізації проблемно-орієнтованих або спеціалізованих систем паралельної і рекурсивної обробки даних реального часу. Показані можливості розпаралелювання та магістральної обробки даних. Методи і алгоритми можуть бути застосовані для створення програмних логічних інтегральних схем.

1. Класи алгоритмів для здійснення ефективної обробки даних. *Основні властивості алгоритмів.* При реалізації алгоритмів на багатопроцесорних системах паралельної обробки даних важливою є задача виділення класів алгоритмів, що допускають розпаралелювання обробки даних на заданому рівні. У зв'язку з цим нижче розглядається поняття алгоритму з точки зору обчислювальних функцій та досліджується можливість розпаралелювання процесів обчислення функцій відносно функцій їх заданого класу.

Теорія алгоритмів виникла у зв'язку з потребами теоретичної математики [1]. Важливою областю використання теорії алгоритмів є теорія і техніка синтезу та побудова високопродуктивних комп'ютерів, особливо у застосуванні найрізноманітніших задачах комп'ютерного зору [2, 3] для реалізації проблемно-орієнтованих і спеціалізованих систем паралельної обробки даних, інформаційно-аналітичних систем.

При визначенні поняття алгоритму встановлюють загальні істотні властивості алгоритмів, характерні для поняття алгоритму [1, 4–7].

1. Алгоритм — це процес послідовної побудови моделі величини, що є в дискретному часі, таким чином, що на початку моменту надається початкова кінцева система величини, а у кожний наступний момент система величин отримується за визначенням, згідно з законом (програмою), із систем величин, які мають у попередній момент часу (дискретність алгоритму).

2. Система величин, що отримують в деякий (не початковий) момент часу, однозначно визначається системою величин, одержаних у попередні моменти часу (детермінованість алгоритму).

3. Закон одержання наступної системи величин із попередніх повинен бути простим і локальним (елементарність кроків алгоритму).

4. Якщо спосіб отримання наступної величини із будь-якої величини не дає результату, то повинно бути вказано, що потрібно вважати результатом алгоритму (спрямованість алгоритму).

5. Початкова система величин може вибиратись із деякої потенційної нескінченної множини (масовість алгоритму).

Означення алгоритму, згідно з умовами 1–5, є нестроге і звичайно називається безпосереднім, або інтуїтивним поняттям алгоритму.

Якщо говорити неформально, алгоритм — це деяка детермінована процедура, що можна застосувати до довільного елемента деякого класу символічних входів і яка для кожного входу дає відповідно в кінці символічний вихід. В алгоритмічних проблемах важливим є визначення алгоритму для розв'язання такої задачі, де вхідними даними є цілочислові параметри x_1, x_2, x_3, $\ldots$, x_n, а вихідними — такі ж цілі числа y_1, y_2, y_3, $\ldots$, y_n. Таким чином, маємо числові функції. Це обмеження числовими функціями не є втратою узагальненості. При цьому числові функції, які мають значення за допомогою деякого алгоритму, називаються обчислювальними функціями. Зауважимо, що одну і ту ж функцію можна обчислити за допомогою декількох алгоритмів. Кожний алгоритм задає функцію, що визначає область застосування і відповідає кожному елементу цієї області застосування результату до цього алгоритму.

Інтуїтивно-обчислювальні функції. Інтуїтивне поняття алгоритму приводить до інтуїтивної обчислювальної функції. Відомо, що довільна функція не є інтуїтивно-обчислювальною. Можливі функції, що мають нечисленну множину, а множини інтуїтивно-обчислювальних функцій мають численну множину. Дійсно, кожній інтуїтивно-обчислювальній функції можна поставити у відповідність текст деякої мови із деякого скінченного алфавіту — українські букви, математичні символи тощо. Таких текстів можна здійснити численну множину, тому інтуїтивно-обчислювальних функцій також численна множина.

Класи числових функцій. Проблеми щодо обчислювальних функцій. К. Гедель [8] вперше описав клас всіх рекурсивних функцій як клас числових функцій у деякій формальній системі. Потім А. Черч у 1936 р. [9] висловив гіпотезу про те, що клас рекурсивних функцій є тотожним класу всіх визначених функцій (Тези Черча). С. Кліні [10] ввів поняття функцій, які обчислюються через алгоритми, як частинно-рекурсивних (теза Кліні). Тому проблема щодо обчислювальних функцій, згідно з тезами Черча і Кліні, рівносильна поясненню її рекурсивності. Поняття рекурсивної функції є строгим [5, 7]. Але поняття обчислювальних функцій є вторинним порівняно з поняттям алгоритму. Проте Е. Пост [11, 12] і А. Тюрінг [13] описали у точних математичних термінах класи машин, на яких можна було б імітувати всі алгоритмічні процеси. При цьому виявилося, що класи функцій, обчислювальних на цих машинах, збігаються з класом всіх частково рекурсивних функцій; алгоритми, що реалізують на цих машинах, запропоновано розглядати як математичні "представники" взагалі всіх алгоритмів.

Проблема. Оцінка класів алгоритмів реалізації заданих задач в реальному часі τ. На сьогоднішній день актуальною проблемою є з'ясування можливостей реалізації алгоритму, який дав би можливість виконати задачі в реальному часі τ на проблемно-орієнтованій або спеціалізованій обчислювальній системі, наприклад на базі програмно-логічних інтегральних системах (ПЛІС). Особливо важливо реалізувати алгоритм для розпаралелювання процесу обробки даних. При цьому необхідно оцінити класи алгоритмів, які можуть ефективно реалізувати задачу в реальному часі τ. В роботах [3, 14] наводяться деякі можливості оцінки розпаралелювання алгоритмів. Нижче розглядається проблема оцінювання класів алгоритмів для реалізації їх розпаралелювання та аналізується можливість їх використання при розв'язанні задач комп'ютерного зору [3, 14] в реальному часі для найрізноманітніших областей знань.

2. Термальні представлення. *Числові функції. Функціональний алфавіт.* Розглянемо означення [3]. Нехай X, Y — деякі числові значення $x_1, \ldots, x_n \in X$, $y_1, \ldots, y_n \in Y$.

1. Числові функції S, O, I_n^m мають значення

$$
\begin{aligned}
&O(x_1, x_2, x_3, \ldots, x_n) = 0,\\
&S(x_1, x_2, x_3, \ldots, x_n) = x_k + 1,\\
&I_n^m(x_1, x_2, x_3, \ldots, x_n) = x_m,\\
&m = \overline{1, n}; \qquad n = 1, 2, \ldots .
\end{aligned} \tag{1}
$$

Ці функції назвемо найпростішими або, відповідно, функціями-константами, функціями слідування, функціями вибору. Функціональний алфавіт складається із символів трьох груп: 1) предметні символи a, b, x, y; 2) функціональні символи f^1, f^2, $\ldots$; букви з верхнім індексом n ($n \geqslant 1$) називаються n-місткі функціональні символи. Тоді говоритимемо, що слова особливого вигляду, які записані у цьому функціональному алфавіті, називаються термами. Якщо операція з X представлена деяким термом, записаним за допомогою функціональних символів f_1, $\ldots$, f_S, предметних символів a_1, $\ldots$, a_r і довільних предметних змінних, то вона називається термальною відносно даних операцій і елементів.

3. Розпаралелювання алгоритмів. Дослідимо обробки даних [3] для розв'язання задач реалізації алгоритмів в комп'ютерному зорі [2].

Оператор суперпозиції. Розглянемо оператор суперпозиції частинних функцій, де задані числові функції f^n і f_1^m, f_2^m, f_3^m, $\ldots$, f_n^m ($n > 0$, $m \geqslant 0$), утворимо з них функцію

$$
g(x_1, x_2, x_3, \ldots, x_m) = f(f_1(x_1, \ldots, x_m), f_2(x_1, \ldots, x_m), \ldots, f_n(x_1, \ldots, x_m)). \tag{2}
$$

Із (2) маємо, що функція g визначена для довільного кортежу $\langle x_1, x_2, x_3, \ldots, x_m \rangle$, для якого знайдено і f_1, f_2, f_3, $\ldots$, f_n, функція f визначена для кортежу $\langle f_1, f_2, f_3, \ldots, f_n \rangle$. Тоді встановимо, що функція g отримана за допомогою оператора суперпозиції S^{n+1} із функцій f, f_1, f_2, $\ldots$, f_n. Якщо визначити через F^n множину всіх часткових числових функцій від n змінних, то оператор S^{n+1} буде всюди визначений функцією із $F^n \times F^m \times F^m \times \times F^m \times \cdots \times F^m$ в F^m. Якщо часткові числові функції, які можна одержати за допомогою операторів суперпозиції $S^2, S^3, \ldots$ із часткових числових функцій $f_1^{n_1}$, $f_2^{n_2}, \ldots$ і функцій I_m^n, де $m, n = 1, 2, \ldots$, довільні, називаємо їх елементарними функціями відносно $f_1^{n_1}$, $f_2^{n_2}$, $\ldots$.

Термальне представлення функцій і розпаралелювання обчислень. Можна показати [3], що:

1) якщо функція $g^m(x_1, x_2, \ldots, x_m)$ отримана за допомогою оператора суперпозиції S^{n+1} із функцій f^n, f_1^m, f_2^m, $\ldots$, f_n^m і її термальне представлення таке:

$$
g^m(x_1, x_2, \ldots, x_m) = S^{n+1}(f^n, f_1^m, f_2^m, \ldots, f_n^m), \tag{3}
$$

то процес обчислення функцій $g^m(x_1, \ldots, x_m)$ можна розпаралелити на рівні функцій f_1^m, f_2^m, $\ldots$, f_n^m;

2) якщо функція $g^m(x_1, x_2, \ldots, x_m)$ допускає термальне представлення

$$
g^m(x_1, x_2, \ldots, x_m) = S^{n+1}(f^n, h_1^m(x_1, \ldots, x_m), \ldots, h_n^m(x_1, \ldots, x_m)), \tag{4}
$$

де

$$
\begin{aligned}
&h_i^m(x_1, \ldots, x_m) = S^{n_{ji}+1}(f_{ji}^{S_{ji}}; I_{q_{i_1}}^m(x_1, \ldots, x_m); I_{q_{i_2}}^m(x_1, \ldots, x_m), \ldots),\\
&1 \leqslant j_i \leqslant n, \qquad 1 \leqslant q_{i_1} \leqslant m, \qquad 1 \leqslant i \leqslant n,
\end{aligned} \tag{5}
$$

то процес обчислення функцій g^m можна розпаралелити на рівні функцій $f_{ji}^{S_{ji}}$.

З (5) можна зробити висновок: довільна функція $g^m(x_1, x_2, \ldots, x_m) \in \Phi_{io}$ — клас інтуїтивно-обчислювальних функцій типу $N^S \to N$ для довільних S. Таким чином, маємо широкий клас функцій розпаралелювання процесу обчислень відносно заданих функцій. Цей клас функцій має счисленну множину відносно елементарних числових функцій. Його можна розглядати як частинну алгебру

$$\eta = \langle \{f_{ji}, O, S, I_m^n\}, \{S^1, S^2, S^3, \ldots\} \rangle,$$

що викликана елементами f_{ji}, O, S, I_m^n.

4. Рекурсивне означення та алгоритми і розпаралелювання. Рекурсія — це спосіб обчислення функцій, де значення функції для довільних значень аргументів безпосередньо визначається значенням тієї ж функції для менших значень аргументів або значеннями більш простих функцій.

Примітивна рекурсія. Представлення функцій. Примітивна рекурсія є одним з найпростіших видів більш загальних рекурсій. В теорії алгоритмів [5–7] поняття рекурсії є дуже важливим, оскільки рекурсивне означення можна встановити як алгоритм. При цьому примітивно-рекурсивна функція — це досить широкий клас всюди визначених функцій, які можна одержати цією формалізацією, що дає можливість вивчати властивості різних алгоритмів з метою розпаралелювань і, відповідно, їх внутрішню структуру.

Розглянемо числові часткові функції $g - n$-містні і $h - n + 2$-містні.

Означення. $n + 1$-містка часткова функція f випливає з функції g і h примітивної рекурсії, якщо для всіх натуральних значень x_1, x_2, x_3, $\ldots$, x_n, y маємо ($n \geqslant 0$)

$$f(x_1, x_2, x_3, \ldots, x_n, 0) = g(x_1, x_2, x_3, \ldots, x_n); \tag{6}$$

$$f(x_1, x_2, x_3, \ldots, x_n, y + 1) = h(x_1, x_2, x_3, \ldots, x_n, y; f(x_1, x_2, x_3, \ldots, x_n, y)). \tag{7}$$

Функція f означена через функції g і h за допомогою операції примітивної рекурсії за схемою примітивної рекурсії або просто примітивної рекурсії, чи примітивно-рекурсивно. Функція f існує для довільних часткових функцій g і h, які задовольняють (6) і (7). Тоді

$$\begin{aligned}
&f(x_1, x_2, x_3, \ldots, x_n, 0) = g(x_1, x_2, x_3, \ldots, x_n);\\
&f(x_1, x_2, x_3, \ldots, x_n, 1) = h(x_1, x_2, x_3, \ldots, x_n, 0; g(x_1, x_2, x_3, \ldots, x_n));\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&f(x_1, x_2, x_3, \ldots, x_n, m + 1) = h(x_1, x_2, x_3, \ldots, x_n, m; f(x_1, x_2, x_3, \ldots, x_n, m)).
\end{aligned} \tag{8}$$

Розглянемо (8) у загальному випадку: якщо дві містких функції g (n-містка) і h ($n +$ $+ 2$)-містка, де $n \geqslant 0$)

$$\begin{aligned}
&f(x_1, \ldots, x_{i-1}, 0 x_{i+1}, \ldots, x_{n+1}) = g(x_1, x_2, x_3, \ldots, x_{i-1}, x_{i+1}, \ldots, x_{n+1});\\
&f(x_1, \ldots, x_{i-1}, y + 1, x_{i+1}, \ldots, x_{n+1}) =\\
&\qquad = h(x_1, \ldots, x_{i-1}; y, x_{i+1}, \ldots, x_{n+1}, f(x_1, \ldots, x_{i-1}, y, x_{i+1}, \ldots, x_{n+1})),
\end{aligned} \tag{9}$$

то (9) однозначно визначає функцію f.

Якщо f існує, то із (9) маємо

$$
\begin{aligned}
&f(x_1,\dots,x_{i-1},0,x_{i+1},x_{n+1}) = g(x_1,x_2,x_3,\dots,x_{i-1},x_{i+1},\dots,x_{n+1});\\
&f(x_1,\dots,x_{i-1},1,x_{i+1},\dots,x_{n+1}) =\\
&\qquad = h(x_1,\dots,x_{i-1},0,x_{i+1},\dots,x_{n+1},f(x_1,\dots,x_{i-1},0,x_{i+1},\dots,x_{n+1}));\\
&\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\\
&f(x_1,\dots,x_{i+1},m+1,x_{i+1},\dots,x_{n+1}) =\\
&\qquad = h(x_1,\dots,x_{i+1},m,x_{i+1},\dots,x_{n+1};f(x_1,\dots,x_{i-1},m,x_{i+1},\dots,x_{n+1})).
\end{aligned}
\tag{10}
$$

Функція f отримана із g і h за допомогою операторів примітивної рекурсії.

Обробка даних при обчисленні функцій, одержаних за допомогою операторів примітивної рекурсії. Маємо [3]:

1) якщо функція f отримана за допомогою операторів примітивної рекурсії із функцій g і $h - f = R(g,h)$, то процес обчислення функцій f не розпаралелений на рівнях g і h;

2) для функції $f \in \Re$, що отримана скінченним числом операцій примітивної рекурсії із F, процес обчислення f не розпаралелений на рівні f_1, $f_2,\dots,S$, O, I_m^n.

Із 1 і 2 випливає наслідок: існує цілий клас функцій, одержаних за допомогою операторів примітивної рекурсії, обчислення яких не підлягає розпаралелюванню на рівні певного підкласу функцій. Цей клас можна розглядати як часткову алгебру $\eta = \langle\{f_i\},R\rangle$ із елементами f_i.

Магістральна обробка даних. Якщо функція f отримана за допомогою оператора примітивної рекурсії із функцій g і $h - f = R(g,h)$, то процес обчислення функцій f задовольняє схему магістральної обробки даних [3].

Таким чином, у роботі запропоновано методи опису алгоритму на основі операторів суперпозиції та рекурсії. Наводяться різні можливості термального представлення алгоритмів обробки даних для реалізації паралельних та магістральних систем обробки даних і застосування у задачах комп'ютерного зору.

1. *Марков А. А.* Теория алгоритмов // Тр. Мат. ин-та им. В. А. Стеклова. – 1961. – № 38. – С. 176–189; № 42. – С. 376–378.
2. *Форсайт Девид, Понс Жанс.* Компьютерное зрение. Современный подход. – Москва; Санкт-Петербург; Киев: ИД "Вильямс", 2004. – 930 с.
3. *Грицик В. В., Грицик В. В.* Розпаралелювання і налаштування алгоритмів обробки даних для реалізації інформаційно-аналітичних системах. – Львів: Вид. Держ. наук.-досл. ін-ту інформ. інфраструктури НАН України, 2008. – 64 с.
4. *Колмогоров А. Н., Успенский В. А.* К определению алгоритма // Усп. мат. наук. – 1958. – № 4. – С. 3–28.
5. *Мальцев А. И.* Алгоритмы и рекурсивные функции. – Москва: Наука, 1965. – 391 с.
6. *Петер Р.* Рекурсивные функции / Под ред. А. Н. Колмогорова. – Москва: Изд-во иностр. лит., 1954. – 264 с.
7. *Роджерс Х.* Теория рекурсивных функций и эффективная вычислимость. – Москва: Мир, 1972. – 624 с.
8. *Gödel* K. Über formal unenfaschaldbare Sätze der Principia Mathematica und verwandfer System I // Monafsch. Math. und Phys. – 1931. – No 38. – S. 173–198.
9. *Church A.* The calculi of lambda conversion // Ann. Math. Stud. – 1941. – No 6. – P. 54–63.
10. *Kleene S. C.* General recursive functions of natural numbers // Math. Ann. – 1936. – **112**. – P. 727–742.
11. *Post E. L.* Finite combinatory process-formulation // J. Symbolic Logic. – 1936. – No 1. – P. 103–105.

12. *Post E. L.* Formal reduction of the general combinatorial decision problem // Amer. J. Math. – 1943. – **65**. – P. 197–215.
13. *Turing A. M.* On computable numbers, with an application to the Entscheidungs problem // Proc. London Math. Soc. Ser. 2. – 1937. – **42**. – P. 230–265.
14. *Параллельная* обработка информации. В 5-ти т. Т. 1. Распараллеливание алгоритмов обработки информации. – Киев: Наук. думка, 1985. – 273 с.

Державний науково-дослідний інститут інформаційної інфраструктури НАН України, Львів

Надійшло до редакції 03.10.2008

V. V. Hrytsyk

The description of algorithms of parallel-recursive data processing in real-time systems

The analysis of the modern state of problems of computer vision is done for the implementation of specialized systems with parallel and recursive data processing in real-time. The capabilities of the paralleling processing of data are shown. The method can be used for the design of logic integrated circuits.