УДК 004; 004.02;004.35;004.9

В. В. Грицик
Тернопільський національний технічний університет ім. І. Пулюя

БАЗОВІ СИСТЕМНІ СТРУКТУРИ СИНТЕЗУ СИСТОЛІЧНИХ СИСТЕМ ОПРАЦЮВАННЯ ДАНИХ У РЕАЛЬНОМУ ЧАСІ

© Грицик В. В., 2015

Досліджено базові системні структури синтезу складних систем комп'ютерного зору для реалізації інформаційно-аналітичних систем реального часу. Описано апаратно-орієнтований метод відеопотоків, метод визначення класів функцій, що дозволяють розпаралелювання, та метод синтезу систем паралельного опрацювання даних.

Ключові слова: розпаралелювання алгоритмів синтезу, обробка даних, єдина обчислювальна середовище, провайдер в реальному часі, вплив паралельних інформаційних технологій.

The basic system structures of synthesis of complex computer systems for the implementation of information-analytical systems in real time are studied in this paper. Namely hardware-based video streaming method, the method of determining the classes of functions that allow parallelization and the method of synthesis of parallel data processing are described.

Key words: paralleling synthesis algorithms, data processing, uniform computing environment, providing real-time, Impact of parallel information technologies.

Вступ. Загальна постановка проблеми

Сьогодні спостерігається тенденція до істотного збільшення обсягів відбору, передавання, зберігання, опрацювання інформації [1–4]. У цьому напрямі перспективних інформаційних технологій важливо зазначити постійно зростаючі вимоги до розроблення та дослідження нових інформаційних технологій та інформаційно-аналітичних систем для опрацювання інформації. Це потребує переосмислення ідей традиційних обчислювальних структур фон Неймана і забезпечення високопродуктивних обчислювальних процесів. Актуальність теми зумовлена постійно зростаючими вимогами до якості та кількості відеопотоків, які потрібно опрацьовувати за одиницю часу під час відеоспостереження, моніторингу, біологічних, медичних і космічних досліджень, виробничих процесів, телебачення, коли в реальному часі необхідно передавати і опрацьовувати великі об'єми відеоданих. На світовому ринку інтелектуальних інформаційних технологій представлено велику кількість програмно-апаратних засобів, призначених для цифрової обробки і аналізу зображень, а саме: адаптивні системи локалізації та моніторингу, системи класифікації, розпізнавання жестів, детектування наявності руху та аналізу області руху на предмет класифікації рухомого об'єкта. Щодо ЄС, то фінансування цієї тематики можна знайти у звітах 7-ї рамкової програми, що публікуються у таких журналах, як Research*EU (results supplements), зокрема, у звітах 2010–2011рр., в яких розглядаються такі задачі, як розпізнавання та проблеми образного мислення. Зокрема на цю тематику через програму FP7 з 2007 виділено 1.9 млрд. евро на 1200 проектів (у кожному з яких є щонайменше три учасники з різних країн). У програмі Горизонт 2020 ця сума збільшена.

Аналізуючи провідні IT-компанії світу (2015 р.), ми з'ясували, що Гугл, Еппл, ФБ застосовують нейронні мережі для розпізнавання звуків та пошуку об'єктів [5]. Компанії Цейс, Олімпус, Акконд продають не просто мікроскопи (чи якісну оптику), а комплексні системи з інтегрованими системами розпізнавання, класифікації й аналізу мікроскопічних об'єктів.

Мета роботи полягає у дослідженні суперечностей між необхідністю підвищення точності розпізнавання образів на складному полі уваги та зменшенням ресурсів (часових та апаратних) при

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

прийнятті рішень завдяки розробленню нових моделей комп'ютерного сприйняття, методів та архітектур опрацювання відеоданих, високопродуктивної інформаційно-аналітичної системи опрацювання відеопотоків у реальному часі.

Основні поняття системи опрацювання інформації

Розглянемо системи опрацювання інформації, за теоретико-множинним підходом [6, 7].

Означення 1. Системою опрацювання інформації є відношення на непустих множинах

$$S \subset \times\{V_i : i \in I\}\ , \tag{1}$$

де $\times$ – символ декартового добутку; I – множина індексів; V_i – об'єкт системи.

Розглянемо множини I_x та $I_y \in$ I, що утворюють I; $I_x \cap I_y = \varnothing$ і $I_x \cup I_y = I$

Означення 2. Множина $X = \times\{V_i : i \in I_x\}$ є вхідним об'єктом (вхід), а множина $Y = \times\{V_i : i \in I_y\}$ – вихідним об'єктом (вихід). За (1) система S визначається відношенням

$$S \subset X \times Y. \tag{2}$$

Це система "вхід-вихід". Розглянемо також систему S, яку можна визначити як $S \subset \{X, C_0, Y\}$, де C_0 – канал, що перетворює (опрацьовує) дані відповідним чином.

Область K і область S визначаються через $D(S)$ і $R(S)$ так:

$$D(S) = \{x : (\exists y)\,(x, y) \in S\};$$

$$R(S) = \{y : (\exists x)\,(x, y) \in S\} \tag{3}$$

Розглянемо [6] множину $\tilde{C}$ і функцію $R : (\tilde{C} \times x) \to y$ таку, що

$$(x,\ y) \in S \Leftrightarrow (\exists z)\ \left[R(\tilde{C},\ x) = y\right]. \tag{4}$$

Множина $\tilde{C}$ є множиною стану системи, а функція R – реакцією системи. Розглянемо довільні множини A і B; T – множина моментів часу; A^T і B^T – множина можливих відображень із T до A і B відповідно; $X \subset A^T$ і $Y \subset B^T$.

Означення 3. Загальною часовою системою S над X і Y є відношення над X і Y.

$$S \subset X \times Y.$$

При цьому множини A і B є алфавітом вхідних даних (вхід) і вихідних даних (вихід) системи S. Стан часової системи S в момент часу $t \in T$ позначимо $\tilde{C}_t$ і визначимо як функцію $b_t : \tilde{C}_t \times X_t \to Y$, що

$$(x_t, y_t) \in S_t \Leftrightarrow (\exists z)\left[b_t\left(\tilde{C}, x_t\right) = y_t\right],$$

$$S_t = S / T_t\,,\ T_t = [t' : t' \geq t].$$

Функція b_t визначає реакцію системи в момент часу t.

Означення 4. Система $S \subset X \times Y$ функціональна тоді і тільки тоді, коли

$$(x, y) \in S \wedge (x, y) \in S \Rightarrow y = y'.$$

Означення 5. Система $S \subset X \times Y$ взаємооднозначно функціональна тоді, коли S функціональна і

$$(x, y) \in S \wedge (x', y) \in S \Rightarrow x = x'.$$

Означення 6. Системи $S_1 \subset X_1 \times Y_1$, $S_2 \subset X_2 \times Y_2$, …, $S_n \subset X_n \times Y_n$ називатимемо інформаційно взаємонезалежними, якщо

$$X_1 \cap \{Y_1, Y_2, ..., Y_n\} = \varnothing;$$

$$X_2 \cap \{Y_1, Y_2, ..., Y_n\} = \varnothing; \tag{5}$$

$$X_n \cap \{Y_1, Y_2, ..., Y_{n-1}\} = \varnothing.$$

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

При виконанні умови (5) системи $S_1, S_2, ..., S_n$ допускають паралельне опрацювання інформації в часі. Розглянемо деяке поле *A* і нехай *X* і *Y*– лінійні алгебри над *A* і допустимо, що

$$s \in S \wedge s' \in S \Rightarrow s + s' \in S; \;\; s \in S \wedge a \in \mathrm{A} \Rightarrow a\ s \in S\,, \qquad (6)$$

де "+" операція додавання в X × Y, $a\ x$ – операція множення на скаляр.

Означення 7. Якщо елементи $s \in S$ задовольняють умови (6), то S є повною лінійною системою. Відповідь на означення умов лінійної системи дають теореми [6, 7]:

Теорема 1. Нехай *X* і *Y* – лінійні алгебри над одним і тим самим полем *A*. Система $S \subset X \times Y$ є лінійною в тому і тільки тому випадку, коли знайдеться така реакція $R : C \times X \to Y$, що:

1) $\widetilde{C}$ є лінійна алгебра над A;

2) є пара таких лінійних відображень $R_1 : \widetilde{C} \to Y$ і $R_2 : X \to Y$ і для всіх $(\tilde{c}, x) \in \widetilde{C} \times X$

$$R(\tilde{c}, x) = R_1(\tilde{c}) + R_2(x)\,.$$

Теорема 2. Нехай $S \subset X \times Y$ – лінійні системи і R – відображення, $R : \widetilde{C} = X \to Y$. Відображення R називається лінійною реакцією системи тоді і тільки тоді, коли:

1) $(x, y) \in S \Leftrightarrow (\exists t)\left[y = R(t, x)\right]$;

2) $\widetilde{C}$ є лінійна алгебра над полем *A* скалярів лінійних алгебр *X* і *Y*;

3) є два таких лінійних відображення $R_1 : \widetilde{C} \to Y$ і $R_2 : X \to Y$, що для довільних $(\tilde{c}, x) \in \widetilde{C} \times X$

$$R(\tilde{c}, x) = R_1(\tilde{c}) + R_2(x)\,.$$

При цьому $\widetilde{C}$ є лінійним об'єктом станів; $R_1 : \widetilde{C} \to Y$ – реакцією на стан; $R_2 : X \to Y$ – реакцією на вхід. Для лінійних систем є можливість зробити висновки стосовно поведінки системи для всього класу вхідних дій відповідно до того, як реагують лише деякі з них.

Відношення паралельного опрацювання інформації (даних) і розпаралелення

Розглянемо поняття відношення паралельного опрацювання інформації і розпаралелення [8].

Нехай задано відношення *R* на множині *X* ×*Y*.

Означення 8. Вважатимемо, що $S_1\ R_{\|}\ S_2$, якщо S_1 і S_2 – системи, що допускають паралельне опрацювання інформації; при цьому відношення $R_{\|}$ називається відношенням паралельного опрацювання інформації.

Теорема 3. Відношення $R_{\|}$ паралельного опрацювання інформації є відношенням еквівалентності. Справді, довільна система S_i реалізується паралельно сама собі (умова рефлективності): $S_i\ R_{\|}\ S_i$ для довільного $i \in I$. Далі маємо $S_{i_1}\ R_{\|}\ S_{i_2} \Rightarrow S_{i_2}\ R_{\|}\ S_{i_1}$ для $i_1, i_2 \in I$, що є умовою симетричності. І нарешті $S_{i_1}\ R_{\|}\ S_{i_2}$ і $S_{i_3}\ R_{\|}\ S_{i_2} \Rightarrow S_{i_1}\ R_{\|}\ S_{i_3}$ для $i_1, i_2, i_3 \in I$, що визначає умову транзитивності. Доходимо висновків:

1. Кожний клас еквівалентності відповідає паралельному опрацюванню інформації, визначеної множиною $\tilde{S}$ систем S_i, що реалізують паралельно з деякою *S*, тобто: $[S] = \left\{S_i / S\ R_{\|}\ S_i\right\}$.

2. Множина всіх класів еквівалентності по $R_{\|}$ утворює фактор-множину множини $\widetilde{S}$ по $R_{\|} : \widetilde{S} / R_{\|}$.

3. Для фактор-множини $\widetilde{S} / R_{\|}$ є система представників відношення $R_{\|}$.

4. Для всіх S_{i_1} і $S_{i_2} \in \tilde{S}$ або $\left[S_{i_1}\right] = \left[S_{i_2}\right]$, або $\left[S_{i_1}\right]$ і $\left[S_{i_2}\right]$ не перетинає.

Означення 5.9. Одночасно із реалізацією множин систем $[S] = \left\{S_i / S\ R_{\|}\ S_i\right\}$ назвемо паралельну операцію $\Pi[S]$.

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Маємо:

Теорема 4. Рефлексивне і транзитивне замикання R^* відношення $R_{\|}$ визначає паралельну операцію $\Pi_{[S]}$.

Справді, нехай потужність множини $\lfloor S_i / S\, R_{\|}\, S_i \rfloor$ дорівнює k. Тоді побудуємо таку послідовність $S_{i_1}\, R_{\|}\, S_{i_2}$, $S_{i_2}\, R_{\|}\, S_{i_3}$, $S_{i_3}\, R_{\|}\, S_{i_4}$, …, $S_{ik-1}\, R_{\|}\, S_{i_n}$, де $i_1,..., i_k \in I$. При цьому очевидно, що $S_{i_1}, S_{i_2}, S_{i_3},..., S_{i_n} \in \{S_i / S\, R_{\|}\, S_i\} = [S]$, тобто $S_{i_1}\, R_{\|}^*\, S_{i_n}$ і визначено $S_{i_1}\, R^*\, S_{i_n}$.

Тоді, використовуючи означення 9 для $\Pi_{[0]}$ на множині систем S_i, рефлексивне і транзитивне замикання R^* відношення $R_{\|}$ визначає паралельну операцію $\Pi_{[0]}$.

Нехай z – індекс відношення еквівалентності $R_{\|}$, визначеного на множині $\{S_i\} = \mathcal{S}$. І нехай $S_{i_1}^1, S_{i_2}^2, S_{i_3}^3,..., S_{i_S}^r$ – система представників відношення $R_{\|}$ для фактор-множини $\mathcal{S} / R_{\|}$.

Теорема 5. Рефлексивне і транзитивне замикання R^* відношення $R_{\|}$ визначає паралельну операцію $\Pi_{[S_{i_1}^1]}, \Pi_{[S_{i_2}^2]},..., \Pi_{[S_{i_z}^z]}$.

Означення 10. Розпаралелення опрацювання інформації в множині систем $\{S_i\}$ визначається як гомоморфізм $\{S_i\}$ на фактор-множині $\mathcal{S} / R_{\|}$.

Узагальнене магістральне (конвеєрне) опрацювання інформації

Важливого значення в теорії і практиці систем опрацювання інформації набувають магістральні методики опрацювання даних [8, 9]. Розрізняють такі основні рівні реалізації магістрального принципу опрацювання даних [8]: рівень пристроїв виконання елементарних операцій над бітами даних; рівень арифметико-логічних пристроїв і рівень керувальних пристроїв, які відповідають арифметико-магістральному, командно-магістральному і макромагістральному опрацюванню даних. За типом команд магістральні системи можна поділити на магістральні системи із звичайними командами і магістральні системи з векторними командами. За способом організації магістралі ці системи поділяють на статичні і динамічні. У випадку динамічних магістральних систем є можливість апаратурного і програмного переналаштування. Нижче розглянемо узагальнений підхід (метод) магістрального опрацювання даних.

Магістральний метод опрацювання даних дає змогу забезпечити виконання операцій в обчислювальних системах, пристроях у паралельному режимі або в режимі збігу [8]. Цей підхід можна успішно застосовувати для розроблення спецпроцесорів паралельного опрацювання даних, ПЛІС у системах фільтрації і розпізнавання образів. Цей метод опрацювання даних є важливим для налаштування однорідних обчислювальних середовищ. Він ґрунтується на тому, що одна й та сама операція повторно виконується над деякими змінними в однаковому потоці даних. Перевагу такого підходу при виконанні арифметичних операцій описано в роботі [8].

Нижче розглянемо метод у загальному випадку.

Нехай задано деяку функцію $f(x)$, процес обчислення якої можна подати у вигляді послідовного обчислення набору деяких елементарних функцій

$$f_1,\ f_2,\ f_3,...,\ f_n. \tag{7}$$

Час виконання $f_1,\ f_2,\ f_3,...,\ f_n$ відповідно дорівнює

$$t_1, t_2, t_3,..., t_n. \tag{8}$$

Позначимо f_i^z – виконання f_i залежно від z-го аргументу в одиничному потоці даних. Нехай для обчислення f маємо n обчислювальних пристроїв (процесорів). Із (7) випливає, що для обчислення значень $f(x)$ необхідно здійснити n кроків. Розглянемо обчислення $f_i\ (i = \overline{1, n})$ над деякими змінними в одиничному потоці даних у вигляді схеми:

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

$$
\begin{array}{lllllll}
\text{Крок 1} & f_1^1 & f_1^2 & f_1^3 \ldots & f_1^n \ldots & f_1^{S+n-1} \ldots & \\
\text{Крок 2} & & f_2^1 & f_2^2 \ldots & f_2^{n-1} \ldots & f_2^{S+n-2} \ldots & \\
\text{Крок 3} & & & f_3^1 \ldots & f_3^{n-2} \ldots & f_3^{S+n-3} \ldots & \\
\ldots\ldots\ldots\ldots & & & & & \ldots\ldots\ldots\ldots & \\
\text{Крок n} & & & & f_n^1 \ldots & f_S^n \ldots &
\end{array}
\tag{9}
$$

На кожному кроці обчислень використовується відповідний обчислювальний пристрій (процесор).

У цьому випадку необхідно $f_1, f_2, f_3, \ldots, f_n$ здійснювати так, щоб задовольняти схему (9). Це основна проблема теорії і практичної реалізації магістрального опрацювання даних, де можна розглядати дві задачі:

1. Чи для $f(x)$ взагалі є представлення (7), що задовольняє схему (9)?

2. Як найкраще вибрати $f_1, f_2, f_3, \ldots, f_n$, щоб задовольняти схему (9)?

Для функціонування цієї схеми в часі необхідно встановити тривалість кожного кроку. На основі (9) і розподілу часу виконання кожної f_i (8) отримуємо, що час реалізації t_i кожного кроку дорівнює

$$t = \max_i t_i \,. \tag{10}$$

Визначимо тепер час обчислення $f(x)$. Із (9) і (10) випливає, що перший результат f_n^1 отримано при $T = nt$. Кожний наступний результат обчислень $f_n^2, f_n^3, \ldots, f_S^n, \ldots$ отриманий через $T_0 = t$. Тому при обчисленні f_r^1 затримка в часі

$$T_{пр} = T - \sum_{i=1}^{r} t_i = nt - \sum_{i=1}^{n} t_i \tag{11}$$

Отже, при опрацюванні даних кожної наступної f_r^S, s = 1, 2, 3, … необхідно витратити менше часу :

$$T_в = \sum_{i=1}^{n} t_i - t = \sum_{i=1}^{n} t_i - \max_i t_i \tag{12}$$

Розглянемо випадок, коли

$$t_1 = t_2 = t_3 = \ldots = t_n = t_0 \,. \tag{13}$$

Тоді з врахуванням (13) рівності (10) ÷ (12) мають такий вигляд:

$$\left.
\begin{aligned}
& t = \max t_i = t_0 \\
& T_{пр} = nt_0 - \sum_{i=1}^{n} t_i = nt_0 - nt_0 = 0; \\
& T_в = \sum_{i=1}^{n} t_i - t_0 = nt_0 - t_0 = (n-1)t_0
\end{aligned}
\right\} \tag{14}$$

У загальному випадку розпаралелення алгоритмів опрацювання даних розглядається для множинного потоку даних і передбачає одночасно виконання на одному ярусі операції в декількох потоках даних. Паралельне опрацювання даних у випадку магістрального методу розглядається для одиничного потоку даних, і в цьому є певна відмінність від паралельного опрацювання даних у загальному випадку, коли розглядається множинний підхід потоків даних, наприклад, у матричних системах. У цьому сенсі магістральний підхід (метод) є частковим випадком розпаралелення алгоритмів опрацювання даних.У загальному випадку можна розглядати множинний потік даних для їх опрацювання у декількох магістралях. Зокрема така структура може бути характерною для багатофункціональних магістралей і векторного опрацювання даних. При

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

цьому якнайповніше проявляються функціональні можливості принципу магістрального опрацювання даних. Нехай задано функції $F_1, F_2, F_3, ..., F_n$, процес обчислення яких можна подати у вигляді набору деяких елементарних функцій.

$$F_1 \quad f_{1(1)} \quad f_{2(1)} \quad f_{3(1)} \cdots f_{k_1(1)};$$
$$F_2 \quad f_{1(2)} \quad f_{2(2)} \quad f_{3(2)} \cdots f_{k_2(2)};$$
$$F_3 \quad f_{1(3)} \quad f_{2(3)} \quad f_{3(3)} \cdots f_{k_3(3)};$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$F_n \quad f_{1(n)} \quad f_{2(n)} \quad f_{3(n)} \cdots f_{k_n(n)}.$$

Час виконання, відповідно, дорівнює

$$t_{1(1)} \quad t_{2(1)} \quad t_{3(1)} \cdots t_{k_1(1)};$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$t_{1(n)} \quad t_{2(n)} \quad t_{3(n)} \cdots t_{k_n(n)}.$$

Нехай для обчислення $F_1, F_2, F_3, ..., F_n$ маємо $\sum_{z=1}^{n} K_z$ обчислювальних пристроїв (процесорів). Для обчислення F_z необхідно реалізувати K_z кроків. Розглянемо обчислення $f_{i(z)}$ над деякими змінними в одиничному потоці даних у вигляді схеми:

$$\text{Крок 1} \quad \begin{array}{c} f^1_{1(1)} \quad f^2_{1(1)} \cdots f^{k_1}_{1(1)} \cdots \\ f^1_{1(2)} \quad f^2_{1(2)} \cdots f^{k_2}_{1(2)} \cdots \\ \ldots\ldots\ldots\ldots \\ f^1_{1(n)} \quad f^2_{1(n)} \cdots f^{k_n}_{1(n)} \cdots \end{array}$$

$$\text{Крок 2} \quad \begin{array}{c} f^1_{2(1)} \quad \cdots \quad f^{k_1-1}_{2(1)} \\ f^1_{2(2)} \quad \cdots \quad f^{k_2-1}_{2(2)} \\ \ldots\ldots\ldots\ldots\ldots \\ f^1_{2(n)} \quad \cdots \quad f^{k_n-1}_{2(n)} \end{array}$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$\text{Крок n} \quad f^1_{k_n(n)} \quad \cdots \quad f^{k_n}_{S_n(n)}$$

На кожному кроці обчислень реалізовано відповідні обчислювачі. Для цього випадку маємо:

$$t(F_z) = \max t_{i(z)} = t_{0(z)}$$

$$T_{пр(F_z)} = k_z - t_{0(z)} - \sum_{i=1}^{k_z} t_{i(z)} = k_z \cdot t_{0(z)} - k_z \cdot t_{0(z)} = 0$$

$$T_{в(F_z)} = \sum_{i=1}^{k_z} t_{i(z)} - t_{0(z)} = k_z \cdot t_{0(z)} - t_{0(z)} = (k_z - 1) t_{0(z)}.$$

Отже, магістральне опрацювання даних дає змогу здійснити паралельні обчислення множинного потоку команд за допомогою послідовного опрацювання великої кількості даних одиничного потоку даних через магістраль декількох спеціалізованих блоків опрацювання даних. У цьому сенсі магістральні системи опрацювання даних реалізують подальший розвиток концепції систем з переглядом вперед та з перекриттям операцій. Магістральні системи опрацювання даних є одними з високопродуктивних систем [8].

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Асоціативне опрацювання інформації

Одним із важливих підходів до реалізації задач, який дозволяє глибоке розпаралелення, є асоціативне опрацювання даних. Для низки задач комп'ютерного зору є можливість опрацьовувати дані в межах великої кількості даних, що швидко надходять, у реальному часі. Асоціативно паралельні процесори залежно від розв'язуваних задач можуть бути як універсальними обчислювальними системами (процесорами), так і проблемно-орієнтованими або спеціалізованими високопродуктивними обчислювальними системами. Функціональну схему і принцип асоціативного опрацювання даних можна розглядати так [8]. Нехай дані представлено у вигляді числової матриці H розмірності $m \times n$ елементів, що складені зі слів, кожне з яких є n розрядів. Тобто:

$$H = \begin{vmatrix} x_{11} & x_{12} & x_{13} \dots x_{1j} \dots x_{1n} \\ x_{21} & x_{22} & x_{23} \dots x_{2j} \dots x_{2n} \\ & \dots\dots\dots\dots\dots\dots & \\ x_{i1} & x_{i2} & x_{i3} \dots \; x_{ij} \dots \; x_{in} \\ & \dots\dots\dots\dots\dots\dots & \\ x_{m1} & x_{m2} & x_{m3} \dots x_{mj} \dots x_{mn} \end{vmatrix} \tag{15}$$

Асоціативне опрацювання даних ґрунтується на збереженні, а також паралельному відборі і записі даних за змістом або за деякими асоціативними ознаками. Як асоціативні ознаки можна вибрати деяке слово огляду або частину його (розряди), розміщені довільно. Розглянемо слово, що складається з асоціативних ознак

$$P_t = \left(p_{1t} \quad p_{2t} \dots p_{jt} \dots p_{nt} \right), \tag{16}$$

і слова-маски

$$M_t = \left(m_{1t} \quad m_{2t} \dots m_{jt} \dots m_{nt} \right). \tag{17}$$

Тоді вхідне слово опитування (запиту) формується зі слів (16) і (17):

$$Y_t = \left(y_{1t} \quad y_{2t} \dots y_{nt} \right);$$

$$y_{jt} = I\left(p_{jt}, m_{jt} \right) = \begin{cases} p_{jt} \text{ при } m_{jt} = 0 \\ 0 \quad \text{при } m_{jt} = 1 \end{cases} \tag{18}$$

Управління зчитуванням або записом здійснюється за допомогою слова

$$U = \left(u_1, u_2, \dots, u_i, \dots, u_m \right),$$

де $u_1 = f\left(x_{11} \quad x_{12} \dots x_{1n} \right)$

$$u_2 = f\left(x_{21} \quad x_{22} \dots x_{2n} \right) \tag{19}$$

$$u_i = f\left(x_{i1} \quad x_{i2} \dots x_{in} \right).$$

Пошук даних здійснюється згідно з рівністю або нерівністю. При цьому пошук, згідно з рівністю, визначається підмножиною слів, які збігаються зі вхідним словом опиту y_t, а пошук згідно з нерівністю – підмножина слів, відмінних від вхідного слова опиту y_t. В асоціативному паралельному процесорі з логікою збігу виконується функція рівнозначності вхідного слова опиту і відповідної інформації. При цьому елементи (19) слова u визначаються так:

$$U_i = \bigwedge_{j=1}^{n} \left(y_i \infty x_{ij} \right) \tag{20}$$

В асоціативному паралельним процесорам, за логікою незбігу, виконується функція нерівнозначності. Елементи (16) слова u визначаються так:

$$U_i = \bigvee_{j=1}^{n} \left(y_i \bar{\infty} x_{ij} \right) . \tag{21}$$

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

При опитуванні виконується перетворення:

$$H = \left\| x_{ij} \right\| \Rightarrow H' = \left\| x'_{ij} \right\| \Rightarrow u \,. \tag{22}$$

Із (19)–(22) випливає, що при $y_{jt} = 1$ у випадку логіки збігу стовпця H_j матриці H не змінює свого значення у матриці H', тобто $H_j = H'_j$: у випадку логіки незбігу стовпець H_j інвертує у випадку логіки збігу. При цьому, якщо на будь-який елемент вхідного слова накладено маску, то у випадку логіки збігу H_j перетворюється на H'_j, де H'_j= (111…1): у випадку логіки незбігу H'_j= (000…0).

Елементи слова u знаходять за допомогою кон'юнкції (20) елементів рядків H' за логіки збігу за допомогою диз'юнкції (21), елементи рядків H' – за логіки незбігу.

Отже, у результаті опиту виділяється деяка підмножина слів, яка, залежно від поставленого завдання, може зчитуватися, або можуть формуватися адреси слів, або визначається їх кількість у цій підмножині, або замість них записується нова інформація. Записувати інформацію можна за допомогою вхідного слова запису

$$S_i = \left(s_{1t} \quad s_{2t} \quad s_{3t} \ldots s_{jt} \ldots s_{nt} \right), \tag{23}$$

яке отримано зі слова асоціативних ознак

$$P'_t = \left(p'_{1t} \quad p'_{2t} \quad p'_{3t} \ldots p'_{jt} \ldots p'_{nt} \right) \tag{24}$$

і слова-маски

$$M'_t = \left(m'_{1t} \quad m'_{2t} \quad m'_{3t} \ldots m'_{jt} \ldots m'_{nt} \right) \tag{25}$$

Тоді з (23)–(25) маємо:

$$S_{jt} = I\left(p'_{jt}, m'_{jt} \right) = \begin{cases} p'_{jt} \text{ при } m'_{jt} = 0 \\ 0 \quad \text{при } m'_{jt} = 1 \end{cases}$$

Інформація записується у вибрану раніше підмножину слів або в одне слово, для яких u = 1, коли на вході є слово запису S_t і подано команду для виконання операції запису. Тоді довільний елемент x'_{ij} матриці H'', отриманої в результаті виконання мікрокоманд опиту і запису, є функції:

$$x''_{ij} = \bigwedge_k \tilde{x}_{ik} \,,$$

$$\tilde{x}_{ik} = \begin{cases} x_{ik} & \text{при} \quad y_k = 1 \\ \bar{x}_{ik} & \text{при} \quad y_k = 0 \end{cases}$$

Отже, під час виконання мікрокоманд опиту і запису реалізується операція Іі-НЕ як повний базис у системі булевих функцій. Тому, виконуючи певні послідовності мікрокоманд опиту і запису, можна реалізувати довільну суперпозицію базових функцій у межах заданого рядка, що означає виконати одночасно довільні логічні й арифметичні операції, однакові для всіх рядків. У роботі [8] запропоновано й проаналізовано алгоритм і технічні засоби для можливості виконання арифметичних і логічних операцій в асоціативному паралельному процесі опрацювання даних і розв'язання різних задач, а також описано метод паралельного опрацювання великої кількості даних у пам'яті з адресацією за вмістом із названої пам'яті з розподіленої логіки. За цим методом можна опрацьовувати числа паралельно за словами і за розрядами, а найефективніше застосовувати його у задачах, для яких характерно природно групувати дані і для яких опрацювання слів у межах кожної групи становить основну частину операції [8].

У роботі [8] для побудови асоціативних паралельних процесорів запропоновано метод однорідної мікроелектронної програмованої структури, що названа клітинною інтегральною решіткою. Ця решітка налаштовується (запрограмована) на реалізацію потрібної функції. Налаштовувати кожний елемент структури можна незалежно від останніх, а клітинна організація схем забезпечує глибоке розпаралелення алгоритмів опрацювання інформації і високу продуктивність.

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Найточніше структури процесора і структури алгоритму розв'язання кожної конкретної задачі забезпечує асоціативний паралельний процесор (АПП) з переналаштованою структурою. АПП такого напряму орієнтовані на реалізацію з опрацюванням масивів, де характерними особливостями задач є виконання групових і матричних операцій. Це найефективніший підхід до розв'язання задач [8] матричної алгебри (опрацювання даних у задачах комп'ютерного зору, цифрова фільтрація, попереднє опрацювання даних, опрацювання графічної інформації, дискретної техніки (технічної діагностики)).

Перерахуємо основні переваги асоціативного опрацювання даних.

Забезпечення високої продуктивності розв'язання задач завдяки структурному підходу, а не внаслідок застосування елементної бази швидкої дії.

Високою є регулярність або однорідність структури, що підвищує технологічність виготовлення АПП на основі ВІС, ПЛІС.

Групове опрацювання масивів інформації є характерним для глибокого розпаралелення алгоритмів, що визначає широкий клас задач, який можна реалізувати на АПП, ПЛІС.

Організація магістрального потоку й опрацювання інформації, що значно підвищує швидкодію.

Висока надійність і достовірність опрацювання даних у системах реального часу.

Отже, основні переваги асоціативного опрацювання даних дають змогу ефективно застосовувати асоціативні паралельні процесори для розв'язання задач найрізноманітніших класів у різних галузях науки, техніки, особливо важливих сьогодні, та вирішення проблем у галузях комп'ютерного зору в системах реального часу.

Класи алгоритмів, які допускають розпаралелення в системах паралельного опрацювання інформації. Алгоритми та обчислювальні функції

Поняття алгоритму є фундаментальним у теорії обчислень [10–13]. Для реалізації алгоритмів у системах паралельного опрацювання інформації є завдання визначити класи алгоритмів, що допускають розпаралелення опрацювання інформації на заданому рівні. Це дає можливість здійснити глибоке ієрархічне розпаралелення обчислень для розв'язання різних задач. У цьому напрямі розглядається поняття алгоритму з погляду обчислювальних функцій і досліджується можливість розпаралелення процесу обчислювання функцій відносно функцій із заданого класу. Враховуючи тези Черча і Кліні, можна припустити, що проблема обчислення функцій пов'язана з її рекурсивністю. Гедель [28] вперше описав клас усіх рекурсивних функцій як клас усіх числових функцій у деякій формальній системі. Згодом (1946) Черч висловив гіпотезу про те, що клас рекурсивних функцій є тотожним класу всіх означених функцій (теза Черча). Кліні [14, 15] ввів поняття частинно-рекурсивної функції, а також висловив гіпотезу про те, що всі частинні функції (тобто не обов'язково означені для всіх значень аргументів функції), при цьому обчислювані через алгоритми, є частинно-рекурсивними (теза Кліні). Власне поняття рекурсивної функції є строгим. Однак поняття обчислювальної функції є вторинним порівняно з поняттям алгоритму. З іншого боку, Пост [16] і Тюринг [17] чіткими і точними математичними термінами описали класи машин, на яких можна було імітувати всі алгоритмічні процеси. Виявилося, що клас функцій, які обчислювали на цих машинах, збігається з класом усіх частинно-рекурсивних функцій, і алгоритми, що реалізують на цих машинах, запропоновано розглядати як математичні "представники" цих алгоритмів. Є різні формальні способи опису алгоритмів [11, 14]. У деякому з формалізмів (машини Тюринга [17], граматика Хомського, алгоритми Маркова [12], лямбда-обчислювання, системи Поста [16] та ін.) можна промоделювати алгоритм, записаний у довільному із цих формалізмів. Тому всі перелічені формалізми є еквівалентними. При цьому дуже важливим є те, що довільною формальною мовою можна описати тільки зчислену множину алгоритмів.

Складність алгоритмів можна оцінювати за різними підходами. У роботі [11] окреслено два основні критерії для оцінювання того, наскільки добре працює алгоритм: 1 – кількість виконаних елементарних механічних операцій залежить від обсягу (величини) входу (критерій часової складності); 2 – обсяг допоміжної пам'яті, необхідної для зберігання проміжних результатів, що

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

виникають у процесі обчислення, є функцією від величини входу (критерій ємної складності). Тепер у задачах комп'ютерного зору також звертають увагу на проблему складності паралельних алгоритмів, які мають важливе значення у застосуванні для збільшення продуктивності опрацювання інформації за рахунок розпаралелення. Дослідженням класів алгоритмів (обчислювальних функцій або частинно-рекурсивних функцій), що допускають розпаралелення, можна пояснити ті шляхи, які необхідні для розв'язання низки задач комп'ютерного зору в забезпеченні опрацювання даних реального часу, а також різних галузей знань технічних задач комп'ютерного зору в умовах реального часу за допомогою спеціалізованих або проблемно-орієнтованих обчислювальних пристроїв паралельної дії, обчислювальних систем, структур, середовищ. У цьому напрямі важливим завданням є визначення класу обчислювальних функцій, для яких є можливість розпаралелення процесу обчислень. Також важливо отримати таку функцію, щоб була можливість оптимального розпаралелення. Ці функції дають можливість досліджувати структуру задач для програмного налаштування систем з'єднань між окремими модулями обчислювальної структури або системи для автоматичної реконфігурації відповідно до реалізації алгоритмом. Нехай задано множини *X* і *Y*. Неформально алгоритм – це деяка детермінована процедура, яку можна застосувати до довільного елемента деякого класу символьних вхідних і яка для кожного такого входу дає в кінці відповідний вихід [11]. В алгоритмічних проблемах важливо визначити алгоритми для розв'язання такої задачі, де входами є цілочислові параметри $x_i \in X$, а виходами – також цілі числа $y_j \in Y$. Отже, отримуємо числові функції. Обмеження числовими функціями не приводить до втрати узагальнення [11]. При цьому числові функції, значення яких можна визначити за допомогою деякого алгоритму, називаються обчислювальними. Одну й ту саму функцію можна обчислити за допомогою різних алгоритмів. Введемо такі означення:

Означення 11. Якщо деяким елементам множини *X* поставлено у відповідність однозначно визначені елементи множини *Y*, тоді говорять, що задана частинна функція з *X* в *Y*.

Означення 12. Частинна функція з $X^{(n)}$ в *Y* називається частинною від *n* змінних до *n*-місних функцій з *X* в *Y*; частинна функція з $X^{(n)}$ до *X* називається *n*- місною частинною операцією на *X*; частинні функції із $N^{(n)}$ до N називаються *n*-місними числовими функціями, N – множина натуральних чисел.

У теорії алгоритмів важливе значення мають числові функції.

Означення 13. Числові функції S^1, O^n, I_m^n, які мають значення

$$O^n\left(x_1, x_2, x_3, ..., x_n\right) = 0$$
$$S^1\left(x_1, x_2, x_3, ..., x_n\right) = x_k + 1 \qquad (26)$$
$$I_m^n\left(x_1, x_2, x_3, ..., x_n\right) = x_m,\ m = \overline{1, n};$$

називаються найпростішими. Функції (1) інколи називаються, відповідно, функціями-константами, функціями наслідування, функціями вибору.

Розглянемо функціональний алфавіт, складений із символів трьох груп:

1. Предметні символи a, b, x, y;
2. Функціональні символи $f^1, g^2, f_0^1, f_1^1, ...;$ букви з верхнім індексом *n (n≥1)* називаються *n*-місними функціональними символами;
3. Символи лівої, правої дужок і коми – "(", ")", ",".

Означення 14. Слова особливого вигляду, що записуються у цьому функціональному алфавіті, називаються термами.

Алгоритми, що допускають розпаралелення опрацювання інформації

Розглянемо деякі числові функції $f^n, f_1^m, f_2^m, f_3^m, ..., f_n^m$ $\left(n > 0, m \geq 0\right)$. Утворимо з них функцію

$$g\left(x_1, x_2, x_3, ..., x_m\right) = f\ (f_1\ (x_1, ..., x_m)\ ,$$

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

$$f_2(x_1,...,x_m),...,f_n(x_1,...,x_m)) \quad (27)$$

Із (27) випливає, що функція g є визначена для кожного кортежу $\langle x_1, x_2, x_3,..., x_m \rangle$, для якого визначена і $f_1, f_2, f_3,..., f_n$. Функція f визначена для кортежу $\langle f_1, f_2, f_3,..., f_n \rangle$. Нехай функція g отримана за допомогою оператора суперпозиції S^{n+1} із функцій $f, f_1, f_2,..., f_n$.

Означення 15. Часткові числові функції, які можна отримати за допомогою операторів суперпозиції $S^2, S^3,\ldots$, з часткових числових функцій $f_1^{n_1}, f_2^{n_2},...$ і функцій I_m^n, де $m, n = 1, 2,\ldots$ довільні, називаємо елементарними функціями відносно $f_1^{n_1}, f_2^{n_2},...$

Нехай $\Phi_{i.o.}$ – клас інтуїтивно-обчислювальних функцій типу $N^S \to N$ для довільного S і $\Phi_{в.в.}$ – клас всюди визначених $N^S \to N$ для довільного S.

Тоді класи $\Phi_{i.o.}$ і $\Phi_{в.в.}$ замкнені відносно оператора суперпозиції.

Теорема 6. Нехай функція $g^m(x_1, x_2,..., x_m)$ отримана за допомогою оператора суперпозиції S^{n+1} з функцій

$$f^n, f_1^m, f_2^m,..., f_n^m$$

й її термальне представлення таке:

$$g^m(x_1, x_2,..., x_m) = \mathbf{S}^{n+1}(f^n, f_1^m, f_2^m,..., f_n^m), \quad (28)$$

тоді процес обчислення функцій $g^{(m)}(x_1, x_2, x_3,..., x_m)$ розпаралелене на рівні функцій $f_1^{(m)}, f_2^{(m)},..., f_n^{(m)}$.

Доведення. Для функцій $f_1^{(m)}, f_2^{(m)},..., f_n^{(m)}$ із (3) виконуються умови:

1. $f_i^{(m)}$ і $f_j^{(m)}$ інформаційно взаємонезалежні для $i, j = \overline{1, n}, \quad i \neq j$;
2. $In\ f_i^{(m)} \cap Out\ f_j^{(m)} = \varnothing.$ (29)

З (29) випливає, що процес обчислення функції $g^{(m)}$ розпаралелений на рівні $f_1^{(m)}, f_2^{(m)},..., f_n^{(m)}$.

Теорема 7. Якщо функція $g^{(m)}(x_1, x_2,..., x_m)$ допускає термальне представлення

$$g^{(m)}(x_1, x_2,..., x_m) =$$
$$\mathbf{S}^{n+1}(f^{(n)}, h^{(m)}(x_1,..., x_m),..., h_n^{(m)}(x_1,..., x_m)), \quad (30)$$

де $h_1^{(m)}(x_1,..., x_m) = \mathbf{S}^{S_{ji}+1}(f_{ji}^{S_{ji}}; I_{q_{i_1}}^{(m)}(x_1,..., x_m)$,

$$I_{q_{i_2}}^m(x_1,..., x_m),...I_{q_{i_{S_{ji}}}}^{(m)}(x_1,..., x_m) \quad (31)$$
$$j_i = \overline{1, n}, \quad q_{i1} = \overline{1, m}, \quad i = \overline{1, n},$$

тоді процес обчислення функції $g_{(S_{ji})}^{(m)}(x_1,..., x_m)$ розпаралелений на рівні функції f_{ji}.

Доведення можна здійснити аналогічно доведенню теореми 1, враховуючи представлення (30) і (31).

За теоремою 2 можна аналогічно розглянути доведення наступної теореми.

Теорема 8. Довільна функція $g^{(m)}(x_1, x_2,..., x_m) \in \Phi_{i.o.}$ допускає розпаралелення процесу обчислення на рівні f_{ji}.

Доведення теореми очевидне, і воно випливає з визначення функції з класу $\Phi_{i.o.}$ та (30) і теореми 7.

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Отже, визначено широкий клас обчислювальних функцій (алгоритмів), що допускають розпаралелення процесу обчислення відносно заданих функцій (на заданому рівні). Цей клас функцій є зчисленою множиною відносно елементарних числових функцій, і його можна визначити як часткову алгебру

$h = <\{ f_{ji}, \mathbf{O}, \mathbf{S}, \mathbf{I}_m^n \}, \quad \{\mathbf{S}^1, \mathbf{S}^2, \mathbf{S}^3, ...\} >$, що породжена елементами $f_{ji}, \mathbf{O}, \mathbf{S}, \mathbf{I}_m^n$.

Нижче розглянемо оператор примітивної рекурсії. У загальному випадку рекурсією називається такий спосіб задання функції, у якому значення визначеної функції для довільних значень аргументів безпосередньо визначаються значеннями аргументів або значеннями "простіших" функцій. Примітивна рекурсія є одним із найпростіших видів загальніших рекурсій. У теорії алгоритмів поняття рекурсії дуже важливе, оскільки рекурсивні означення можна розглядати як алгоритми; при цьому примітивно-рекурсивні функції можна отримати за допомогою такої формалізації [59]. Це дає можливість вивчити властивості різних алгоритмів і досліджувати їх внутрішню структуру [59]. Розглянемо числові часткові функції *g- n-* місна і *h-n+2-* місна.

Означення 16. *n +1*-місна часткова функція отримується з функцій *g* і *h* примітивної рекурсії, якщо для всіх натуральних значень $x_1, x_2, x_3, ..., x_n, y$ маємо (n ≥ 0)

$$f\ (x_1, x_2, x_3, ..., x_n, 0) = g\ (x_1, x_2, x_3, ..., x_n) \tag{32}$$

$$f\ (x_1, x_2, x_3, ..., x_n, y+1) = h(x_1, x_2, x_3, ..., x_n, y,$$
$$f\ (x_1, x_2, x_3, ..., x_n, y)). \tag{33}$$

У такому випадку функція *f* визначена через функції *g* і *h* за допомогою операції примітивної рекурсії (за схемою примітивної рекурсії або просто примітивної рекурсії). Із [18] відомо, що для довільних часткових функцій *g, n* - місна і *h, n +2* -місна (n ≥ 0) є одна і тільки одна часткова функція *f- n+1*-місна, яку можна отримати за допомогою оператора примітивної рекурсії

$$f = R\ (g, h),$$

де *R* – оператор примітивної рекурсії, визначений на множині *F* усіх часткових функцій.

Теорема 9. Якщо функція *f = R (g, h)* отримана за допомогою примітивної рекурсії із функцій *g* і *h*, тоді процес обчислення функції *f* не розпаралелений на рівні *g* і *h*.

Доведення. Справді, з означення (31), (32) і (33) випливає*:*

1. g і *h* інформаційно взаємозалежні.

2. $In\ g \cap Out\ h \neq \varnothing$.

Отже, процес обчислення функції *f* не розпаралелюється на рівні *g* і *h*.

Нехай задана система *F* часткових функцій $f_1, f_2, f_3...$ і найпростіші функції $\mathbf{S}, \mathbf{O}, \mathbf{I}_m^n$; η – часткова алгебра функцій, отриманих за допомогою оператора примітивної рекурсії.

Теорема 10. Для $f \in h$, отриманої кінцевою кількістю операцій примітивної рекурсії із *F*, процес обчислення *f* не розпаралелений на рівні $f_1, f_2, ..., \mathbf{S}, \mathbf{O}, \mathbf{I}_m^n$.

Доведення є аналогічним доведенню теореми 4.

Наслідок із теорем 4 і 5.

Є цілий клас функцій, отриманих за допомогою операторів примітивної рекурсії, обчислення яких не підлягає розпаралеленню на рівні певного класу функцій. Цей клас функцій можна розглядати як часткову алгебру:

$$h = <\{ f_i \}, \mathbf{R} >,$$

що породжена f_i.

Теорема 11. Якщо функція *f = R (g, h)* отримана за допомогою операції оператора примітивної рекурсії із функцій *g* і *h*, то процес обчислення функції *f* задовольняє схему магістрального опрацювання інформації при одиничному потоці даних.

Доведення. Розглянемо часткові функції *g-n*-місну і *h-n+2*-місну:

$$g(x_1, x_2, x_3, ..., x_{i-1}, x_{i+1}, ..., x_{n+1}),$$
$$h(x_i, ..., x_{i-1}, 0, x_{i+1}, ..., x_{n+1}, y)$$

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Тоді $n+1$-місна функція $g = R\,(g, h)$. Введемо такі позначення

$$
\begin{aligned}
&f_1^1 = g(x_1, x_2, x_3, ..., x_{i-1}, x_{i+1}, ..., x_{n+1}) \\
&f_2^1 = h(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_{n+1}, f_1^1), \\
&f_3^1 = h(x_1, ..., x_{i-1}, 1, x_{i+1}, ..., x_{n+1}, f_2^1), \\
&\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\
&f_{m+2}^1 = h(x_1, ..., x_{i-1}, m, x_{i+1}, ..., x_{n+1}, f_{m+1}^1).
\end{aligned}
\tag{34}
$$

Використовуючи (34), створимо магістральну схему опрацювання інформації для одиничного потоку даних для обчислень функцій

$$
\begin{aligned}
&\text{Крок 1} \quad f_1^1,\ f_1^2,\ f_1^3 \dots f_1^{m+2} \dots f_1^{s+m+1} \dots \\
&\text{Крок 2} \quad f_2^1,\ f_2^2 \dots f_2^{m+2} \dots f_2^{s+m} \dots \\
&\text{Крок 3} \quad f_3^1 \dots f_3^m \dots f_3^{s+m+1} \dots \\
&\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\
&\text{Крок m+2} \quad f_{m+2}^1 \dots f_3^{m+2} \dots
\end{aligned}
\tag{35}
$$

Із (35) випливає, що процес обчислення функцій f за допомогою оператора примітивної рекурсії R із функції g і h задовольняє схему магістрального опрацювання інформації для одиничного потоку даних.

Наслідок. З'являється клас обчислювальних функцій алгоритмів, які отримуються за допомогою операторів примітивної рекурсії, для яких процес обчислення допускає магістральну схему опрацювання інформації. Цей клас обчислювальних функцій (алгоритмів) є алгеброю

$$h = <\{f_i\}, R>,$$

яка спричинена (викликана) елементами f_i.

Введемо такі означення:

Означення 20. Функція f є примітивно-рекурсивною, якщо її можна отримати скінченним числом операцій суперпозиції і примітивної рекурсії, враховуючи лише найпростіші функції S, O, I_m^n.

Означення 21. Часткова функція f називається примітивно-рекурсивною відносно системи часткових функцій σ, якщо f можна отримати з функцій системи σ і найпростіших функцій S, O, I_m^n скінченним числом операцій суперпозиції і примітивної рекурсії.

Отже, примітивно-рекурсивними (примітивно-рекурсивними відносно σ) є ті і тільки ті функції, які є значенням термів, що записані за допомогою предметних символів $R, S^2, S^3, ...$

Узагальненням теореми 6 є наступна теорема.

Теорема 12. Якщо функція f отримана за допомогою операторів примітивної рекурсії R з функцій $f_1, f_2, ..., f_n$ і її термальне представлення має вигляд

$$f = R(R(R(...(R(R(R(f_1), f_2), f_3), f_4), .., f_{n-2}), f_{n-1}), f_n),$$

то процес обчислення функції f задовольняє схему магістрального опрацювання інформації.

Таке доведення є аналогічним доведенню теореми 6.

Теорема 13. Якщо функція f отримана за допомогою операторів суперпозиції і примітивної рекурсії і термальне представлення має вигляд

$$
\begin{aligned}
&f = S^{n+1}(f_0, R(R(...(R(R(R(f_1^1, f_2^1), f_3^1), f_4^1), ... f_{m_1-1}^1), f_{m_1}^1), \\
&R(R(...(R(R(f_1^2, f_2^2), f_3^2), f_4^2), ..., f_{m_2-1}^2), f_{m_2}^2), \\
&\quad R(R(...(R(R(f_1^3, f_2^3), f_3^3), f_4^3), ..., f_{m_3-1}^3), f_{m_3}^3), \\
&\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\
&R(R(...(R(R(f_1^n, f_2^n), f_3^n), f_4^n), ..., f_{m_n-1}^n), f_{m_n}^n)).
\end{aligned}
\tag{36}
$$

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Тоді:

1. Процес обчислення функції f розпаралелений на рівні функцій $h^i = R(R(...(R(R(f_1^i), f_2^i), f_3^i), f_4^i),..., f_{m_i-1}^i), f_{m_i}^i), \ i = \overline{1,n}$;

2. Процес обчислення функцій h^i задовольняє схему магістрального опрацювання інформації.

Доведення можна провести, використовуючи теореми 1 і 7.

Теорема 14. Якщо функція f отримана за допомогою операторів примітивної рекурсії і суперпозиції та її термальне представлення має вигляд

$$\begin{aligned} f = R(&R(...R(R(S^{m_1+1}(f_0^1, f_1^1, f_2^1,..., f_{m_1}^1),\\ &S^{m_2+1}(f_0^2, f_1^2, f_2^2,..., f_{m_2}^2)),\\ &S^{m_3+1}(f_0^3, f_1^3, f_2^3,..., f_{m_3}^3)),\\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\ &S^{m_{n-1}+1}(f_0^{n-1}, f_1^{n-1}, f_2^{n-1},..., f_{m_{n-1}}^{n-1})),\\ &S^{m_n+1}(f_0^n, f_1^n, f_2^n,..., f_{m_n}^n)), \end{aligned} \tag{37}$$

тоді:

1. Процес обчислення функції f задовольняє схему магістрального опрацювання інформації;

2. Процес обчислення функції f розпаралелений на рівні функцій $f_S^i, i = \overline{1,n}, s = \overline{0,m_i}$.

Доведення можна проводити, використовуючи теореми 1 та 7.

Термальне представлення (36) і (37) є найбільш узагальненим і відбиває концепцію ієрархічності розпаралелення алгоритмів на різних рівнях. Термальне представлення (37) відбиває довільну деревоподібну гіллясту структуру алгоритму. Такому представленню відповідає структура, яка широко застосовується для розв'язання різних задач математичної фізики, наприклад, гіллястих дробів. За допомогою такого представлення можна легко описати різні ланцюгові фізичні процеси.

Основні висновки такі:

1. Визначено поняття алгоритму з погляду обчислювальних функцій і досліджено можливість розпаралелення процесу обчислення функцій відносно функцій із заданого класу.

2. Вивчено властивості різних термальних представлень обчислювальних функцій, отриманих за допомогою операторів суперпозиції та примітивної рекурсії відносно розпаралелення опрацювання інформації при їх обчисленні. Отримано термальні представлення для примітивно-рекурсивних функцій, процес обчислення яких розпаралелюється відносно функцій із заданого класу.

3. Вивчено класи алгоритмів, що допускають глибоке розпаралелення процесу опрацювання інформації. Визначено класи алгоритмів, що допускають магістральну схему опрацювання інформації. Їх можна ефективно реалізувати на систолічних структурах, однорідних середовищах.

4. Отримано термальні представлення, що є найбільш узагальненими і відповідають концепції ієрархічності розпаралелення алгоритмів на різних рівнях. Ці представлення властиві для структур алгоритмів, за допомогою яких можна описати довільні "ланцюгові" гіллясті фізичні та природні процеси, а також задач та алгоритмів опрацювання даних комп'ютерного зору.

Розпаралелення та налаштування в системах опрацювання даних комп'ютерного зору

Розпаралелення в системах опрацювання даних комп'ютерного зору

Тепер важливим є підхід до налаштування опрацювання даних для великої кількості однорідних середовищ. У напрямі реалізації задач великої розмірності важливим є дослідження з виявлення паралелізму алгоритму, а також необхідність розроблення принципово нових підходів конструктивних алгоритмів паралельної структури для розв'язання задач великих розмірностей у системах комп'ютерного зору для різних задач предметних галузей знань, практичного застосування. Незважаючи на значні досягнення, у різних задачах комп'ютерного зору є необхідність практичного опрацювання зображень у реальному часі.

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Важливим для розвитку теоретичних досліджень розпаралелення відносно структурних підходів, системних застосувань, методів реалізації задач комп'ютерного зору є дослідження, проведені у [19, 20, 8].

Довільну складну задачу можна розглядати для реалізації деякої системи, складеної із підсистем:

$$S_1 \subset X_1 \times Y_1, S_2 \subset X_2 \times Y_2, ..., S_m \subset X_m \times Y_m,$$

де $X = \{x_1, x_2, ..., x_m\}$ – множина вхідних даних; $Y = \{y_1, y_2, ..., y_m\}$ – множина результатів при реалізації $S_1, S_2, ..., S_m$.

Розглянемо модель, яка відтворює послідовно-паралельну організацію опрацювання інформації – ярусно-паралельну форму (ЯПФ). Для цього побудуємо функціональну граф-схему опрацювання даних: 1) поставимо у відповідність кожній підсистемі $S_i, i = \overline{1,m}$ деяку вершину графу; 2) вершина графу, відповідно до функціональної підсистеми S_j, поєднується з вершиною графу відповідно до функціональної підсистеми $S_j, j = \overline{1,m}$ лише тоді, якщо вихід (результат) y_i є одним із входів (аргументів) підсистеми S_j. Побудована таким чином граф-схема повністю відповідає внутрішній структурі задачі і визначає деяку систему $S = \{S_1, S_2, ..., S_m\}$ реалізації задачі.

Означення 22. Функціональна підсистема вважається інформаційно залежною від S_i, якщо S_j реалізує над виходом S_i.

Означення 23. Функціональна підсистема S_i є керівною відносно підсистеми S_j і $S_S(i, j, s = \overline{1,m})$, якщо вихід S_i визначає реалізацію S_j або S_s, а S_i безпосередньо передує S_j і S_s.

При реалізації функціональної граф-схеми у ній виділяються ділянки, всередині яких не вміститься керувальна підсистема.

Впорядкована ця функціональна граф-схема так.

Означення 24. Множини функціональних підсистем Ω_1 першого ярусу назвемо множиною тих і тільки тих підсистем, які є інформативно незалежними; множина функціональних підсистем Ω_2 другого ярусу є множиною тих і тільки тих підсистем, які інформативно залежні хоч би від однієї підсистеми першого ярусу і не є залежними від інших підсистем. Множина функціональних підсистем Ω_i-го ярусу незалежна від інших підсистем, що не належать ярусам з номерами меншими, ніж *i*.

Означення 25. Граф, відповідно впорядкований функціональними підсистемами згідно з ярусами, які визначені множинами функціональних підсистем $\Omega_1, \Omega_2, \Omega_3, ..., \Omega_i, ...,$ будемо називати ярусно-паралельним графом, а систему S, відповідно до цього графу, що названа ярусно-паралельною формою (ЯПФ).

Отже, розпаралелення процесу розв'язання задачі розбивається на відповідні етапи.

Перший етап паралельного розкладу – Ω_1: з множин $\{S\} \subset \{(X_r, Y_r)\}, r = \overline{1,l}$ визначається максимальне число інформативно взаємонезалежних підсистем $S_1^1, S_2^1, S_3^1, ..., S_{\delta_1}^1$. Число δ_1 визначає максимальну кількість процесорів, необхідних для реалізації підсистем $S_\rho^1, 1 \le \rho \le \delta_1$ на першому етапі паралельного розкладу Ω_1. Останні l-δ_1 підсистем є інформативно залежними від підсистем S_ρ^1. Другий етап розкладу – Ω_2: з множини $\{S_z / S_g^1\}$ визначаємо максимальне число δ_2 інформативно незалежних підсистем $S_1^2, S_2^2, S_3^2, ..., S_{\delta_1}^2, \delta_2 \ge \delta_1$. Число δ_2 визначає максимальну кількість елементів-процесорів, необхідних для реалізації підсистем $S_\rho^2, \delta \le \rho \le \delta_2$ на другому етапі паралельних розкладів Ω_2. Останні z-δ_1-δ_2 підсистем є інформативно залежними від підсистем S_ρ^2

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

і, можливо, від S_ρ^1 і т. д. Кожний етап паралельного розкладу – Ω_k: з множини $\left\{S_r / \bigcup_{v=1}^{k-1} S_\rho^v\right\}$ визначаємо максимальне число δ_k інформативно взаємонезалежних підсистем $S_1^k, S_2^k, S_3^k, ..., S_{\delta_k}^k, \delta_k \geq \delta_{k-1}$. Останні $l - \sum_{r=1}^{k-1} \delta_r$ підсистем є інформаційно залежними від підсистем S_ρ^{k-1} і, можливо, від S_ρ^{k-2}, S_ρ^{k-3},…, S_ρ^1. Цей процес розпаралелення продовжується до тих пір, коли $\left\{S_r / \bigcup_{v}^{N-1} S_\rho^v\right\}$, де N – номер ярусу паралельного розкладу Ω_N, що є множиною інформативно взаємонезалежних підсистем. При цьому $\delta_N = 1 - \sum_{r=1}^{N-1} \delta_r$ визначає кількість необхідних процесорів на останньому етапі Ω_N і $\delta_N \leq \delta_{N-1}$. Максимальна кількість процесорів, необхідних для розв'язання відповідної задачі опрацювання інформації, $\delta_{max} = max\,\delta_k$.

У загальному випадку розпаралелення опрацювання даних у системах реального часу зводиться до побудови з функціонального графу підсистеми S_i ярусно-паралельного графу. Основним недоліком цього підходу є відсутність правил формування підсистем. У роботі [47] запропоновано підхід до розпаралелення, де розглядається деяка глобальна характеристика внутрішньої синхронної функціональної схеми.

Нехай деяка задача реалізується за допомогою системи $S \subset (X, C, Y)$, де C – канал, у якому перетворюються дані. Канал С задається у загальному випадку за допомогою матриці ймовірностей $\|P_{ij}\|$, де P_{ij} – ймовірність, що характеризує перетворення вхідної множини X в Y. Підсистеми $S_r \subset (X_r, C_r, Y_r)$ можна інтерпретувати так: а) обчислювальний пристрій виконує елементарні мікрооперації; б) обчислювальний пристрій здійснює складні функціональні залежності $f_1, ..., f_n$; обчислювальні середовища й ін. Якщо характеристики каналів $C_1^k, ..., C_{\delta_k}^k$ у системах $S_1^k, ..., S_{\delta_k}^k$ однакові, то системи S_k називаємо однорідними. Такі однорідні системи мають значний практичний інтерес для розв'язання задач опрацювання даних у реальному часі [21–27] і значні переваги порівняно з неоднорідними у вартості технічної реалізації на основі ВІС, у компактності побудови, в забезпеченні високої надійності. Тому потрібно здійснювати такі алгоритми опрацювання даних, які дають можливість реалізовувати задачі на однорідних системах паралельного опрацювання даних.

Синтез складних систем паралельного опрацювання даних та їх налаштування

Розглянемо можливість розпаралелення опрацювання інформації при синтезі складних систем опрацювання даних на рівні ярусно-паралельних структур алгоритму, застосування магістральних методів реалізації процесу опрацювання даних. Такий підхід дає можливість налаштування системи на реалізацію задач у заданому режимі опрацювання даних, що надходять. У цьому підході важливим є вибір основних операторів, що дозволяють налаштовувати системи. Розглянемо систему

$$S_i \subset X_i \times Y_i \,, \tag{38}$$

і нехай $X_i = \times\left\{X_{ij} : j \in I_{X_i}\right\}$, $Y_i = \times\left\{Y_{ij} : j \in I_{Y_i}\right\}$.

Позначимо через Z_{X_i} декартовий добуток компонентних множин X_i, які можуть використовуватися для реалізації з'єднань систем; а через $\overline{Z}_{X_i}$ – сімейство всіх компонентів множин X_i і позначимо через

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

$$\overline{X}_i^* = \left\{ X_{ij} : X_{ij} \in \overline{X}_i \wedge X_{ij} \notin \overline{Z}_{X_i} \right\},$$

де $\overline{X}_i$ – сімейство компонентних множин X_i;

$$X_i^* = \times \left\{ X_{ij} : X_{ij} \in \overline{X}_i \wedge X_{ij} \notin Z_{X_i} \right\} = \times \left\{ X_{ij} \cdot X_{ij} \in \overline{X}_i^* \right\}.$$

Так отримуємо:

$$X_i = X_i^* \times Z_{X_i} . \tag{39}$$

Аналогічно

$$Y_i = Y_i^* \times Z_{Y_i} . \tag{40}$$

Із (38) і (40) можна синтезувати множину з'єднаних систем

$$S_{iz} \subset \left(X_i^* \times Z_{X_i} \right) \times \left(Y_i^* Z_{Y_i} \right) . \tag{41}$$

Системи S_i і $S_{i\bar{z}}$ не однакові; система $S_{i\bar{z}}$ визначає можливість з'єднання (синтезування) систем. Клас синтезованих систем із (41) визначимо так

$$\overline{S}_z = \left\{ S_{iz} : S_{iz} \subset \left(X_i^* \times Z_{X_i} \right) \times \left(Y_i^* \times Z_{Y_i} \right) \right\}.$$

У цьому класі систем знайдемо основні параметри синтезу систем.

I. Каскадний синтез (з'єднання):

Нехай $S_1 \subset X_1 \times \left(Y_1^* \times Z_{X_1} \right)$, $S_2 \subset (X_2^* \times Z_Y) \times Y_2$.

Введемо операцію ○: $\overline{S}_Z \times \overline{S}_Z \Rightarrow \overline{S}_Z$ таку, що $S_1 \circ S_2 = S_3$, де

$S_3 \subset (X_1 \times X_2^*) \times (Y_1^* \times Y_2)$, $Z_{X_1} = Z_{Y_2} = Z$

і $((X_1, X_2), (Y_1, Y_2) \in S_3 \Leftrightarrow (\exists z)((X_1 (Y_1, Z)) \in S_1 \wedge ((X_2, Z), Y_2) \in S_2),$.

Операцію ○ визначимо як каскадний синтез або каскадну операцію.

II. Паралельний синтез (з'єднання):

Нехай $S_1 \subset \left(X_1^* \times Z_{X_1} \right) \times Y_1$, $S_2 \subset (X_1^* \times Z_{X_2}) \times Y_2$.

Введемо операцію +: $\overline{S}_Z \times \overline{S}_Z \Rightarrow \overline{S}_Z$ таку, що $S_1 + S_2 = S_3$, де

$S_3 \subset (X_1^* \times X_2^* \times Z) \times (Y_1 \times Y_2)$, $Z_{X_1} = Z_{X_2} = Z$

і $((X_1, X_2, Z), (Y_1, Y_2)) \in S_3 \Leftrightarrow ((X_1, Z) Y_1) \in S_1 \wedge ((X_2, Z), Y_2) \in S_2$.

Операцію + назвемо паралельним синтезом (з'єднання) або паралельною операцією.

III. Замикання зворотного зв'язку (організація операції циклів).

Нехай F – відображення F: $\overline{S}_Z \Rightarrow \overline{S}_Z$ таке, що $F(S_1) = S_2$,

де $S_1 \subset (X^* \times Z_X) \times (Y^* \times Z_Y)$, а $S_2 \subset X^* \times Y^*$, $Z_X = Z_Y = Z$

і $(X, Y) \in S_2 \Leftrightarrow (\exists z)(((X, Z), (Y, Z)) \in S_1).$

Відображення F називається замиканням зворотного зв'язку або операцією замикання оберненого зв'язку. Отже, введено три основні операції синтезу систем, які практично вичерпують можливості організації операцій складних систем опрацювання даних.

Подаємо основні властивості синтезу операцій опрацювання даних.

1. Якщо операцію $(S_1 \circ S_2) \circ S_3$ визначено, то справедлива рівність
   $(S_1 \circ S_2) \circ S_3 = S_1 (S_2 \circ S_3)$.
2. $S_1 \circ S_2 \neq S_2 \circ S_1$.
3. Якщо операції $(S_1 + S_2) + S_3$ і $S_1 + (S_2 + S_3)$ визначено, тоді справедлива рівність
4. $(S_1 + S_2) + S_3 = S_1 + (S_2 + S_3)$. $S_1 + S_2 = S_2 + S_1$.
5. В операції ○ немає одиничного елемента.
6. Роль одиничного елемента для операції + відіграє пуста система .

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

7. F($S_1 \circ S_2$) = F($S_2 \circ S_1$), якщо обидві частини цієї рівності мають сенс.

8. Якщо системи $S_1 \subset X_1 \times (Y_1 \times Z)$ і $S_2 \subset (X_2 \times Z) \times Y_2$ не попереджують, тоді не по-переджує і система $S_3 = S_1 + S_2$.

9. Якщо системи $S_1 \subset (X_1 \times Z) \times Y$ і $S_2 \subset (X_2 \times Z) \times Y$ не попереджують, тоді не буде попереджувати і система $S_3 = S_1 + S_2$.

10. Якщо системи S_1 і S_2 лінійні, тоді і системи $S_1 \circ S_2$, $S_1 + S_2$ і F(S_1) будуть лінійними, якщо вони визначені.

11. Якщо системи S_1 і S_2 функціональні, тоді функціональними є системи $S_1 \circ S_2$ і $S_1 + S_2$ за умовою, що вони визначені; каскадне і паралельне з'єднання зберігають властивості взаємооднозначної функціональності; операції замикання зворотного зв'язку у загальному випадку функціональності не зберігають.

12. Нехай $S \subset (X + Z) \times (Y \times Z)$ є функціональними і

$$S(X) = \{Z : (\exists y)((x,z,y,z) \in S)\},$$

$$S(x,y) = \{Z : (\exists z')((x,z,y,z') \in S)\}.$$

Система F (S) є функціональна в тому і тільки тому випадку, коли для кожного $x \in X$ $(\exists y)(S(x) \subset S(x,y))$.

Тепер дослідимо можливості розпаралелення опрацювання інформації для різних операцій синтезу систем на рівні заданих систем S_i. Операція замикання оберненого зв'язку дозволяє опрацьовувати дані лише на рівні однієї системи S_i. Тому важливо дослідити можливість розпаралелення опрацювання даних при побудові різних систем за допомогою каскадного і паралельного поєднання (з'єднання) операцій на рівні заданих систем S_i.

Теорема 15. Нехай задано системи опрацювання даних

$$S_1 \subset X_1 \times (Y_1^* \times Z_{X_1}), S_2 \subset (X_2^* \times Z_{Y_2}) \times Y_2$$

і визначено каскадні з'єднання цих систем за допомогою каскадної операції

$$S_1 \circ S_2 = S = (X_1 \times X_2^*) \times (Y_1^* \times Y_2),$$

тоді система *S* допускає магістральне опрацювання даних на рівні S_1 і S_2.

Доведення. Нехай системи S_1, S_2, S реалізують деякі функції f_1, f_2, f (рис. 8). Побудуємо магістральну схему опрацювання даних одиничного потоку для визначення функції *f*.

Час →

Крок 1 $\quad f_1^1 \quad f_1^2 \quad f_1^3 \quad \ldots$

Крок 2 $\quad f_2^1 \quad f_2^2 \quad \ldots$ (42)

де f_i^r – функція f_i залежно від *r* – аргументу в одиничному потоці даних. Аналогічно побудуємо магістральну схему функціонування системи *S* при одиничному потоці даних на входах S_1 і S_2. Позначимо через S_i^r – систему S_i в момент *r* для визначення f_i^r.

Крок 1 $S_1^1 \quad S_1^2 \quad S_1^3 \quad \ldots$

Крок 2 $S_2^1 \quad S_2^2 \quad \ldots$

Аналогічні результати можна отримати при мультимагістральних системах опрацювання даних.

Теорема 16. Нехай задані системи опрацювання даних

$$S_1 \subset X_1 \times (Y_1^* \times Z_{X_1}), S_2 \subset (X_2^* \times Z_{Y_2}) \times (Y_2^* \times Z_{Y_2}),$$

$$S_3 \subset (X_3^* \times Z_{Y_3}) \times (Y_3^* \times Z_{X_3}), \ldots, S_n \subset (X_n^* \times Z_{Y_n}) \times Y_n.$$

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Каскадне з’єднання цих систем визначаємо за допомогою каскадного з’єднання операцій

$$S_1 \circ S_2 \circ S_3 \circ \ldots \circ S_n = S \subset (X_1 \times X_2^* \times X_3^* \times \ldots \times X_n^*) \times$$

$$\times (Y_1^* \times Y_2^* \times Y_3^* \times \ldots \times Y_{n-1}^* \times Y_n);$$

тоді система *S* допускає магістральне опрацювання даних на рівні S_1, S_2, S_3, …, S_n.

Рис. 1. Мультимагістральне опрацювання даних

Нехай системи S_1, S_2, S_3, …, S_n, S реалізують функції $f_1, f_2, f_3, \ldots, f_n, f$. На рис. 9 зображено систему *S* опрацювання даних для визначення *f*. Побудуємо магістральну схему опрацювання даних для одиничного потоку для визначення *f*.

Час →

Крок 1 f_1^1 f_1^2 f_1^3 ... f_1^n ...

Крок 2 f_2^1 f_2^2 ... f_2^{n-1} ...

Крок 3 f_3^1 ... f_3^{n-2} ...

…….

Крок n ..... f_n^1 ...

Аналогічно можна побудувати магістральну схему функціонування *S* для одиничного потоку даних на входах S_1, S_2, S_3, …, S_n.

Час →

Крок 1 S_1^1 S_1^2 S_1^3 ... S_1^n ...

Крок 2 S_2^1 S_2^2 ... S_2^{n-1} ...

Крок 3 S_3^1 ... S_3^{n-2} ...

…….

Крок n S_n^1.

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Рис. 2. Мультимагістральна система опрацювання даних

Теорема 17. Нехай задані системи опрацювання даних $S_1 \subset (X_1^* \times Z_{X_1}) \times Y_1$, $S_2 \subset (X_2^* \times Z_{X_2}) \times Y_2$ і визначено паралельне з'єднання (синтез) цих систем за допомогою паралельного з'єднання операції

$$S_1 + S_2 = S = (X_1^* \times X_2^* \times Z) \times (Y_1 \times Y_2).$$

Тоді система S допускає розпаралелення опрацювання даних на рівні S_1 і S_2.

Доведення. Якщо за умови побудови синтезу *S*

$$X_1^* \cap Y_2 = \wedge X_2^*, \quad Z_{X_2} \cap Y_1 =$$

системи S_1 і S_2, згідно з означенням, є інформаційно незалежними, тоді система *S* допускає розпаралелення опрацювання даних на рівні S_1 і S_2. На рис. 10 представлено систему *S* опрацювання даних.

Рис. 3. Система синтезу S паралельних систем S_1 і S_2

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Теорема 18. Нехай задані системи опрацювання даних $S_1 \subset (X_1^* \times Z_{X_1}) \times Y_1$, $S_2 \subset (X_2^* \times Z_{X_2}) \times Y_2$, $S_3 \subset (X_3^* \times Z_{X_3}) \times Y_3, ..., S_n \subset (X_n^* \times Z_{X_n}) \times Y_n$ і визначено синтез (паралельне з'єднання) цих систем за допомогою паралельного з'єднання операцій

$$S_1 + S_2 + S_3 + ... + S_n = S = (X_1^* \times X_2^* \times X_3^* \times ... \times X_n^* \times Z) \times \times (Y_1 \times Y_2 \times Y_3 \times ... \times Y_n).$$

Тоді система S допускає розпаралелення опрацювання даних на рівні S_1, S_2, S_3, …, S_n.

Доведення. Оскільки за умовою побудови S

$$X_1^*, Z_{X_1} \cap Y_2, Y_3, ..., Y_n = \varnothing \wedge X_2^*, Z_{X_2} \cap Y_1, Y_3, ... Y_n = \varnothing \wedge ... \wedge X_n^*,$$

$$Z_{X_n} \cap Y_1, Y_2, Y_3, ..., Y_{n-1} = \varnothing,$$

То, згідно з означенням, системи S_1, S_2,…, S_n інформаційно взаємонезалежні. У цьому випадку система S допускає розпаралелення опрацювання даних на рівні S_1, S_2,…, S_n. На рис. 11 представлено систему синтезу S паралельного опрацювання даних.

Теорема 19. Нехай задані системи опрацювання даних

$$S_1 \subset (X_1^* \times Z'_{X_1}) \times (Y_1^* \times Z_{X_1}), \quad S_2 \subset (X_2^* \times Z_{Y_2}) \times Y_2, \quad S_3 \subset (X_3^* \times Z_{X_3}) \times Y_3$$

і паралельне каскадне з'єднання (синтез) цих систем за допомогою операцій

$$(S_1 \circ S_2) + S_3 = S_{12} + S_3 = S \subset (X_1^* \times X_2^* \times X_3^* \times Z) \times (Y_1^* \times Y_2 \times Y_3).$$

Тоді система S допускає магістральне опрацювання даних на рівні S_1 і S_2 і розпаралелення опрацювання даних на рівні S_{12} і S_3.

Рис. 4. Система синтезу S паралельного опрацювання даних S_1, S_2,…, S_n

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Доведення. Справді, згідно з теоремою 1, підсистема $S_{12} = S_1 \circ S_2$ допускає магістральне опрацювання даних для одиничного потоку на рівні S_1 і S_2, а за теоремою 3 система S = $S_{12} + S_3$ допускає розпаралелення опрацювання даних на рівні S_{12} і S_3. На рис. 12 представлено систему *S* опрацювання даних.

Теорема 20. Нехай задані системи опрацювання даних $S_1 \subset (X_1^* \times Z_{X_1}) \times Y_1$,

$$S_2 \subset (X_2^* + Z_{X_2}) \times (Y_2^* \times Z_{X_2}), \quad S_3 \subset (X_3^* \times Z_{Y_3}) \times Y_3$$

і паралельне каскадне з'єднання цих систем за допомогою операцій

$$S_1 + (S_2 \circ S_3) = S_1 + S_{23} = S \subset (X_1^* \times X_2^* \times X_3^* \times Z) \times (Y_1 \times Y_2^* \times Y_3),$$

тоді система *S* допускає магістральне опрацювання даних на рівні S_2 і S_3 і розпаралелення опрацювання даних на рівні S_1 і S_{23}. Доведення цієї теореми аналогічне доведенню теореми 5, оскільки $S_1 + (S_2 \circ S_3) = (S_2 \circ S_3) + S_1$.

Рис. 5. Синтез паралельного і магістрального опрацювання даних

Теорема 21. Нехай задані системи опрацювання даних

$$S_1 \subset (X_1^* \times Z_{X_1}) \times (Y_1^* \times Z'_{Y_1}), \quad S_2 \subset (Z_{X_2} \times X_2^*) \times (Y_2^* \times Z''_{Y_2}), \quad S_3 \subset (X_3^* \times Z'_{Y_3} \times Z''_{Y_3}) \times Y_3$$

і паралельне каскадне з'єднання цих систем за допомогою операцій

$$(S_1 + S_2) \circ S_3 = S_{12} \circ S_3 = S \subset (X_1^* \times X_2^* \times X_3^* \times Z) \times (Y_1^* \times Y_2^* \times Y_3),$$

тоді система *S* допускає розпаралелення опрацювання даних на рівні S_1 і S_2 і магістральне опрацювання даних на рівні S_{12} і S_3.

Доведення. Згідно з теоремою 3 підсистеми $S_{12} = S_1 + S_2$ допускають розпаралелення опрацювання даних на рівні S_1 і S_2, а згідно з теоремою 1 система S допускає магістральне опрацювання даних для одиночного потоку на рівні S_{12} і S_3. На рис. 13 представлено систему S.

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Теорема 22. Нехай задані системи

$$S_1 \subset (X_1^* \times Z_{X_1}) \times (Y_1^* \times Z_{Y_1}),\ S_2 \subset (X_2^* \times Z_{X_2}) \times (Y_2^* \times Z_{Y_2})$$

і визначене каскадне з'єднання, що охоплене оберненим зв'язком за допомогою каскадного з'єднання операції та операції замикання оберненого зв'язку

$$\mathrm{F}\,(S_1 \circ S_2) = S \subset (X_1^* \times X_2^*) \times (Y_1^* \times Y_2^*), \tag{43}$$

тоді система S допускає розпаралелення опрацювання даних на рівні S_1 і S_2.

Доведення. За умовою:

$$X_1^* \cap Y_2^* = \varnothing \ \wedge\ X_2^* \cap Y_2^* = \varnothing \tag{44}$$

З (43) і (44) маємо, що система *S* допускає розпаралелення опрацювання даних на рівні S_1 і S_2. На рис. 14 зображено систему *S*.

Рис. 6. Синтез каскадного з'єднання та операції замикання оберненого зв'язку

Рис. 7. Синтез, що охоплений оберненим зв'язком за допомогою каскадних операцій замикання оберненого зв'язку

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Теорема 23. Нехай задано системи

$S_1 \subset (X_1^* \times Z_{X_1}) \times (Y_1^* \times Z_{Y_2}),$

$S_2 \subset (X_2^* \times Z'_{X_2} \times Z''_{X_2}) \times (Y_2^* \times Z'_{Y_2} \times Z''_{Y_2}), ..., S_{n-1} \subset (X_{n-1}^* \times Z'_{X_{n-1}} \times Z''_{X_{n-1}}) \times (Y_{n-1}^* \times Z'_{Y_{n-1}} \times Z''_{Y_{n-1}})$

і визначено каскадне з'єднання, що охоплено оберненим зв'язком за допомогою каскадних операцій замикання оберненого зв'язку

$$F(S_1 \circ S_2 \circ \ldots \circ S_n) = S \subset (X_1^* \times X_2^* \times ... \times X_n^*) \times (Y_1^* \times Y_2^* \times ... \times Y_n^*).$$

Тоді система S допускає розпаралелення (45) опрацювання даних на рівні S_1, S_2,…, S_n.

Доведення. За умовою

$$X_1^* \cap Y_2^* \cap ... \cap Y_n^* = \varnothing \wedge X_2^* \cap Y_1^* \cap Y_3^* \cap ... \cap Y_n^* = \varnothing$$

$$X_n^* \cap Y_1^* \cap ... \cap Y_{n-1}^* = \varnothing \qquad (45)$$

З (45) випливає, що система S допускає розпаралелення опрацювання даних на рівні S_1, S_2,…, S_n. На рис. 15 представлено систему S, отриману за допомогою операцій каскадних з'єднань замикання оберненим зв'язком.

Функціональний опис налаштування системи опрацювання даних

Для функціонального опису складних систем опрацювання даних, отриманих у результаті налаштування (програмного або апаратного) з'єднання між окремими підсистемами на розв'язання задачі або клас задач, використано теореми 15–23. При цьому використаємо термальний запис функціонування систем опрацювання даних.

Асинхронне і синхронне паралельне опрацювання даних

Обчислювальну систему, спеціалізовані, проблемно-орієнтовані системи, отримані в результаті налаштування з'єднань між окремими підсистемами (теорема 3) на асинхронне або синхронне паралельне опрацювання даних для розв'язання задачі (класу задач), функціонально можна описати за допомогою оператора суперпозиції

$$f = S^3(f_0, f_1, f_2), \qquad (45)$$

де S – оператор суперпозиції; f_0 – функція (операція) з'єднань (f_0 = "+").

У загальному випадку налаштування синтезу (з'єднань) окремих підсистем обчислювальних систем, середовищ, які реалізують функції $f_0, f_1, f_2, ..., f_n$ на проведення паралельного опрацювання даних (теореми 3, 4) можна функціонально описати за допомогою термального запису

$$f = S^{n+1}(f_0, f_1, f_2, ..., f_n).$$

Магістральне (конвеєрне) опрацювання даних

Нехай задано дві підсистеми S_1 і S_2, які реалізують f_1 і f_2, і проведено налаштування з'єднань на магістральне опрацювання даних (теорема 1). У цьому випадку з'єднання між окремими підсистемами обчислювальної системи можна описати за допомогою термального представлення

$$f = R(f_1, f_2), \qquad (46)$$

де R – оператор примітивної рекурсії.

У загальному випадку, коли задано n – підсистем S_1, S_2,…, S_n, які реалізують функції $f_1, f_2, ..., f_n$, і проведено налаштування з'єднань на магістральне опрацювання даних (теорема 1), функціональний опис цієї системи можна отримати за допомогою термального представлення

$$f = R(f_{n-1}, f_n) = R(R(f_{n-2}, f_{n-1}), f_n = R(R(R(f_{n-3}, f_{n-2}), f_{n-1}), f_n) = ... =$$

$$= R(R(R(...(R(R(R(f_1, f_2), f_3), f_4), ..., f_{n-2}), f_{n-1}), f_n). \qquad (47)$$

Розглянемо деякі узагальнені модифікації магістрального і паралельного опрацювання даних у загальному випадку системи типу "множина потоків даних – множина потоків команд".

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Нехай проведено налаштування з'єднань підсистем на опрацювання даних. Тоді функціональний опис отриманої системи здійснюється за допомогою термального представлення

$$f = S^3(f_0, f_3, R(f_1, f_2)) \tag{48}$$

У загальному випадку функціональний опис налаштування (синтезу) з'єднань підсистем на опрацювання даних у системі "множинний потік даних – множинний потік команд" (мультимагістральне опрацювання даних) отримується за допомогою термального представлення

$$f = S^{n+k+1}(f_0, f_1, ..., f_k, R(R(...(R(R(f_1^1, f_2^1), f_3^1), f_4^1), ..., f_{m_1-1}^1), f_{m_1}^1),$$
$$R(R(...(R(R(f_1^2, f_2^2), f_3^2), f_4^2), ..., f_{m_2-1}^2), f_{m_2}^2), \quad R(R(...(R(R(f_1^3, f_2^3), f_3^3), \tag{49}$$

З (49) легко отримати деякі частинні випадки мультимагістрального опрацювання даних.

При $n = 1$ маємо

$$f = S^{k+2}(f_0, f_1, ..., f_k, R(R(...(R(R(f_1^1, f_2^1), f_3^1), f_4^1), ..., f_{m_1-1}^1), f_{m_1}^1). \tag{50}$$

Система, що реалізує (50), є системою з однією магістраллю і k-підсистем, що працюють паралельно. При $k = 0$ маємо

$$f = S^{n+1}(f_0, R(R(...(R(R(f_1^1, f_2^1), f_3^1), f_4^1), ..., f_{m_1-1}^1), f_{m_1}^1), R(R(...(R(R(f_1^2, f_2^2),$$
$$f_3^2), f_4^2), ..., f_{m_2-1}^2), f_{m_2}^2), R(R(...(R(R(f_1^3, f_2^3), f_3^3), f_4^3), ..., f_{m_3-1}^3), f_{m_3}^3), ...$$
$$R(R(...(R(R(f_1^n, f_2^n), f_3^n), f_4^n), ..., f_{m_n-1}^n), f_{m_n}^n). \tag{51}$$

Система, що реалізує (51), є мультимагістральною системою з n різними за довжиною взаємонезалежними і такими, що працюють паралельно, магістралями. При n = 0 маємо

$$f = S^{k+1}(f_0, f_1, f_2, ..., f_k). \tag{52}$$

Система, що реалізує (8), збігається з системою, що реалізує (1) для k=n.

При $m_1 = m_2 = m_3 = ... = m_n = m$ маємо:

$$f = S^{n+k+1}(f_0, f_1, f_2, f_3, ..., f_k, R(R(...(R(R(f_1^1, f_2^1), f_3^1), f_4^1), ..., f_{m-1}^1), f_m^1),$$
$$R(R(...(R(R(f_1^2, f_2^2), f_3^2), f_4^2), ..., f_{m-1}^2), f_m^2),$$
$$R(R(...(R(R(f_1^3, f_2^3), f_3^3), f_4^3), ..., f_{m-1}^3), f_m^3),$$
$$R(R(...(R(R(f_1^n, f_2^n), f_3^n), f_4^n), ..., f_{m_n-1}^n), f_m^n). \tag{53}$$

Система, що реалізує (53), є мультимагістральною системою з n однаковими за довжиною взаємонезалежними і такими, що працюють паралельно, магістралями.

Отже, із (49) випливає низка практичних важливих випадків систем мультимагістрального опрацювання даних. При цьому є можливість синтезувати (будувати) як завгодно складні підсистеми, з'єднані в магістральне опрацювання даних; магістралі підсистем є інформативно взаємонезалежними.

Розглянемо найзагальніший випадок налаштування та синтезу підсистем мультимагістрального опрацювання даних.

Нехай задано підсистеми S_1, S_2, S_3 і проведено налаштування (синтез) на магістральне опрацювання даних (теореми 6, 7). Функціональний опис налаштування систем можна отримати за допомогою термального представлення

$$f = R(S^3(f_0, f_1, f_2), f_3). \tag{54}$$

У загальному випадку налаштування з'єднань на магістральне опрацювання даних (46) функціональний опис має вигляд

$$\begin{aligned} f = R(R(...R(R(&S^{n_1+1}(f_0^1, f_1^1, f_1^2, ..., f_{n_1}^1),\\ &S^{n_2+1}(f_0^2, f_1^2, f_2^2, ..., f_{n_2}^2)),\\ &S^{n_3+1}(f_0^3, f_1^3, f_2^3, ..., f_{n_3}^3)),\\ &\dots\dots\dots\dots\dots\dots\dots\dots\\ &S^{n_{k-1}+1}(f_0^{k-1}, f_1^{k-1}, f_2^{k-1}, ..., f_{n_{k-1}}^{k-1})),\\ &S^{n_k+1}(f_0^k, f_1^k, f_2^k, ..., f_{n_k}^k)). \end{aligned} \tag{55}$$

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

Систему, яка реалізує (55), назвемо узагальненою мультимагістральною системою. Функціональний опис (55) враховує найрізноманітніші інформаційні зв'язки між процесорами магістральної системи. З (55) випливають різні практичні важливі випадки.

При $n_1 = n_2 = n_3 = ... = n_k = n$ маємо

$$\begin{aligned} f = R(R(...R(R(S^{n+1}(f_0^1, f_1^1, f_2^1,..., f_n^1), \\ S^{n+1}(f_0^2, f_1^2, f_2^2,..., f_n^2)), \quad S^{n+1}(f_0^3, f_1^3, f_2^3,..., f_n^3)),... \\ ... S^{n+1}(f_0^{k-1}, f_1^{k-1}, f_2^{k-1},..., f_n^{k-1})), \quad S^{n+1}(f_0^k, f_1^k, f_2^k,..., f_n^k)). \end{aligned} \tag{56}$$

Якщо інформаційні зв'язки є між процесорами, які обчислюють f_i^z і f_i^{z+1} $\forall_i = \overline{1, n_k}$ і $\forall_z = \overline{1, k}$, тоді (56) мають обчислювальну систему, синтезовану і налаштовану для n взаємонезалежної магістралі, де кожний процесор рекурсивно визначає f_i^z. Для $n_1 = n_2 = n_3 = ... = n_k = 1$ з (55) маємо одну магістральну лінію опрацювання даних. Використовуючи (55), можна налаштувати для синтезу на реалізацію довільних гіллястих обчислювальних процесів.

Висновки та перспективи подальших наукових розвідок

У роботі досліджено алгоритм розпаралелення для систем паралельного опрацювання даних. Запропоновано підхід і синтез для ярусно-паралельної форми.

1. Запропоновано системний підхід до опрацювання даних комп'ютерного зору.
2. Досліджено і запропоновано метод синтезу налаштування систем паралельного опрацювання даних. Для цього визначено основні операції і каскадне з'єднання операцій ("○"), паралельне з'єднання операцій ("+") і операція замикання оберненого зв'язку ("F"), які покладено в основу синтезу і реалізації мови опису складних систем.
3. Визначено класи систем і запропоновано конструктивні алгоритми для синтезу паралельного та магістрального опрацювання даних. Наведено низку прикладів.
4. Використовуючи запропонований метод термального запису функціонування систем опрацювання даних, показано можливість реалізації проблемно-орієнтованих і спеціалізованих структур реального часу для складних задач штучного інтелекту.
5. Досліджено та запропоновано метод функціонального опису та синтезу налаштування найбільш загальних систем опрацювання даних для можливостей реалізації складних систем типу наноструктур.

*1. EU research: from FP7 to Horizon 2020//research*eu focus magazine №15. – 2015. – P. 13. 2. Why The EU is betting big on 5G. 3.Грицик В. В. Інформаційні технології і системи у застосуванні комп'ютерного зору / Грицик В. В. ; ДНДІІІ. – Львів : Сполом,2009. – 115 с. 11. 4. Опрацювання відеозображень біологічних клітин людини для різних умов апоптозу. Результати практичних експериментів апоптичних зображень клітин людини у рамках реалізації інформаційно-аналітичних систем : дисертація / Мін-во освіти і науки України, Харківський нац. ун-т радіоелектроніки. – Харків, 2013. – 354 с. – Додаток 1 (27 с.). 5. http://research.google.com/pubs/pub38131.html. 6. Месарович М. Общая теория систем : математические основы / М. Месарович, И. Такахара. – М. : Мир, 1978. 7. Месарович М. Теория иерархических многоуровневых систем / М. Месарович, Д. Мако, И. Такахара; пер. с англ. – М. : Мир, 1973. – 344 с. 8. Грицик В. В. Розпаралелювання алгоритмів обробки даних для реалізації інформаційно-аналітичних систем / В. В. Грицик, В. В. Грицик (мол.); ДНДІІІ. – Львів : Сполом, 2007. – 87 с. 9. Самофалов К. Г. Основы построения конвейерных ЭВМ / К. Г. Самофалов, Г. М. Луцкий. – К. : Вища школа, 1981. – 224 с. 10. Клини С. Введение в метаматематику / С. Клини. – М. : Изд-во иностр. лит., 1957. – 526 с. 11. Мальцев А. И. Алгоритмы и рекурсивные функции / А. И. Мальцев. – М. : Наука, 1965. – 391 с. 12. Марков А. А. Теория алгоритмов / А. А. Марков // Труды математического института им. В. А. Стеклова. – 1951. – № 38. – С. 376–378. 13. Минский М. Вычисления и автоматы /*

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua

М. Минский. – М. : Мир, 1971. – 364 с. 14. Клини С. Введение в метаматематику / С. Клини. – М. : Изд-во иностр. лит., 1957. – 526 с. 15. Nieniewski M. Morfologia matematuczna w przetwarzaniu obrazów / Mariusz Nie-niewski. – Warszawa : Akademicka oficyna wydawnicza PLJ, 1998. – 315 p. 16 Prszelaskowski A. Kompresja danych / Artur Prszelaskowski. – Warszawa : Wyd. BTC, 2005. – 260 p. 18. Виткус Р. Ю. Адаптивные линейные фильтры для обработки изображений / Р. Ю. Виткус, Л. П. Ярославский // Адаптивные методы обработки изображений / под ред. : В. И. Сифорова, Л. П. Ярославского. – М. : Наука, 1988. – С. 6–35. 19. Грицик В. В. Паралельні алгоритми обробки даних в реальному часі / В. В. Грицик // Системні технології, математичне та програмне забезпечення інтелектуальних систем : Регіональний міжвузівський зб. наук. праць. – Дніпропетровськ, 2007. – Вип. 6 (53). – С. 84–87. 20. Грицик В. В. Оцінка якості передавання і комп'ютерна обробка даних образів / В. В. Грицик // Доповіді НАН України. – 2008. – № 9 : Інформатика та кібернетика.– С. 43–48. 21. Вальковский В. А. Лекции по параллельному программированию. Параллелизм в ЭВМ и языках программирования : уч. пособие / В. А. Вальковский. – Новосибирск : Изд-во Новосиб. у-та, 1982. – 88 с. 22. Вальковский В. А. Распараллеливание циклов общего вида методов пирамид / В. А. Вальковский // Кибернетика. – 1983. – № 3. – С. 41–49. 23. Вальковский В. А. Автоматическое построение параллельных программ. Распараллеливание выражений и циклов / В. А. Вальковский, В. Е. Котов. – Новосибирск, 1979. – 40 с. – (Препр. / Вычисл. центр Сиб. АН СССР ; № 146). 24. Вальковский В. А. Элементы параллельного программирования / В. А. Вальковский, В. Е. Котов, А. Т. Марчук, Н. Н. Миренков. – М. : Радио и связь, 1983. – С. 28–36. 25. Васильев В. И. Распознающие системы / В. И. Васильев. – К. : Наук. думка, 1969. – 292 с. 26. Гельфанд И. М. О континуальных моделях управляющих систем / И. М. Гельфанд, М. Л. Цетлин // Доклады АН УССР. – 1960. – № 6. – С. 1242–1245.27. Глушков В. М. Два универсальных критерия эффективности вычислительных машин / В. М. Глушков // Доклады АН УССР. – 1960. – № 4. – С. 477–481. 28. Яцимірський М. М. Швидкі алгоритми ортогональних перетворень / М. М. Яцимірський. – Львів: Академічний експрес, 1997. – 219 с.

Lviv Polytechnic National University Institutional Repository http://ena.lp.edu.ua