Курс «Паралельні алгоритми: побудова та аналіз».

Лекція 9. ***Паралельні обчислювальні системи нетрадиційної архітектури*** **(*ч.* 2).**

План лекції.

1. Системи зі структурно-процедурною організацією обчислень.
2. Нейромережеві обчислювальні системи.
3. Обчислювальні машини потоку даних.

1. Останнім часом в обчислювальній техніці використовується паралельна пам'ять, яка дозволяє звертатися до багатьох даних одночасно. Але наявність паралельної пам'яті створює певні складності для програмування, оскільки виникають задачі розташування даних, перерозміщення даних під час виконання програми, мінімізація кількості пересилань даних тощо.

Паралельна пам'ять (багато модулів пам'яті) дозволяє зчитувати одночасно багато даних – по одному із кожного модуля. Найпростіше ця можливість реалізується у суперкомп'ютерах з розподіленою пам'яттю. Такими є кластерні системи, n-Cube, Connection Machine, система ПС-2000 та інші. Але в таких комп'ютерах одержання процесором даних із модуля пам'яті іншого процесора (міжпроцесорне пересилання) є досить тривалою операцією.

По-іншому організовано доступ до паралельної пам'яті у **суперкомп'ютерах зі структурно-процедурною організацією обчислень**. Такі комп'ютери почали розроблятися і розробляються у Науково-дослідному інституті мультипроцесорних обчислювальних систем при Таганрогському державному радіотехнічному університеті (м. Таганрог, Росія) спочатку під керівництвом академіка РАН А.В. Каляєва, а потім під керівництвом його сина, академіка РАН І.А. Каляєва. У суперкомп'ютерах цього типу час доступу кожного процесора до будь-якого модуля пам'яті є однаковим. При цьому кожен процесор може одночасно одержати два аргументи для бінарної операції. А самі процесори з допомогою комутатора можуть з'єднуватися у відповідності з будь-яким графом обчислень.

Для задач слабозв'язаних, тобто таких, що розбиваються на сімейства самостійних підзадач, що не передбачають частих обмінів даними, суперкомп'ютери з розподіленою пам'яттю є достатньо ефективними. Суперкомп'ютери зі структурно-процедурною організацією обчислень призначені для задач сильнозв'язаних. Але в першому і другому випадках розробка програмного забезпечення є дуже дороговартісною.

Суперкомп'ютер зі структурно-процедурною організацією обчислень має сегментовану сумісновикористовувану пам'ять і поле процесорів, з яких з допомогою комутатора може збиратися конвеєр будь-якої конфігурації (зокрема, декілька конвеєрів). Ця обчислювальна система складається із множини секторів пам'яті, множини елементарних процесорів і комутатора. Процесори не мають своєї локальної пам'яті. Для простоти подальшого викладу можна припустити, що всі процесори є однаковими і час виконання будь-якої операції в будь-якому проце-

сорі є однаковим і дорівнює часу засилання даного із будь-якої комірки будь-якого сектора пам’яті в процесор та дорівнює одиниці часу – одному тактові.

Кожен процесор має на вході буфер. Цей буфер дозволяє затримувати надходження даних в процесор. Такі дії бувають важливими при синхронізації конвеєрного обчислення. При цьому дозволяється в один момент часу звертатися до різних секторів пам’яті (для читання або запису).

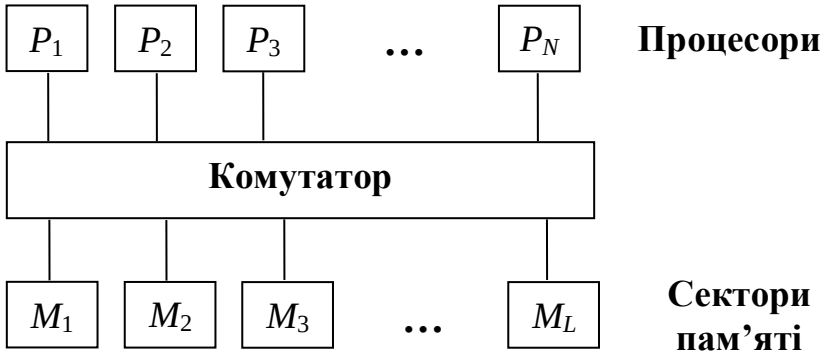


Рис. 1. Модель суперкомп’ютера зі структурно-процедурною організацією обчислень

Із кожного сектора пам’яті і із кожного процесора є вихід, що веде на вхід комутатора (див. рис. 1). До кожного сектора пам’яті і до кожного входу кожного процесора веде який-небудь вихід комутатора.

Комутатор дозволяє з’єднувати входи і виходи обчислювальної системи в будь-яких необхідних комбінаціях, допускаючи з’єднання одного входу з декількома виходами. Вважається, що час переналаштування комутатора є суттєво більшим за одиницю часу (один такт), тому слід мінімізувати кількість переналаштувань комутатора.

Розглянута обчислювальна система дозволяє з допомогою комутатора з’єднувати елементарні процесори в конвеєри різних конфігурацій і підключати до них сегменти пам’яті з даними.

Автоматичне розпаралелювання для суперЕОМ з сегментованою пам’яттю є проблемним із-за необхідності автоматичного розташування даних. Дані в сегментованій пам’яті можна розташовувати по-різному. Розташування повинно бути таким, щоб в будь-який момент часу роботи програми можна було б одночасно зчитувати всі необхідні дані. Якщо б сегментів пам’яті було так багато, що в кожному з них можна було розташувати одне дане, то такої проблеми не було б узагалі. А насправді, реально, в кожному сегменті доводиться розташовувати багато даних. Задача розташування даних полягає в тому, щоб ті дані, які виявляться в одному сегменті пам’яті, не потрібно було б одночасно обробляти. Крім цього, дані повинні бути розташовані так, щоб їх легко можна було знаходити. На даний час розроблена методика безконфліктного розташування даних і пропонуються алгоритми оптимального розташування даних для суперкомп’ютерів із структурно-процедурною організацією обчислень. Зауважимо, що тут слова «сектор» і «сегмент» пам’яті є синонімами.

2. Більшість розглядуваних нами раніше паралельних обчислювальних систем орієнтовані на розв'язання добре формалізованих задач, які зводяться до обчислень за формулами для заданих вхідних даних. Але існує широкий клас практично важливих, але погано формалізованих задач, наприклад, таких як

- розпізнавання образів;
- кластеризація даних;
- прогноз погоди;
- задачі діагностування тощо.

Нейромережеві обчислювальні системи або нейрокомп'ютери призначені для розв'язання саме таких задач.

Нейрокомп'ютери відрізняються від інших ЕОМ великими можливостями і в них принципово змінюється спосіб використання обчислювальної машини. Місце програмування займає навчання, тобто нейрокомп'ютер навчається розв'язувати задачі. Навчання – це коректування ваг зв'язків, унаслідок якого кожна вхідна дія призводить до формування відповідного вихідного сигналу. Після навчання нейронна мережа може застосовувати одержані навички до нових вхідних сигналів.

Поштовхом до розвитку нейрообчислень стали дослідження в біології. Нервова система людини і тварин складається із окремих клітин – нейронів. У мозку людини їх кількість досягає $10^{10} – 10^{12}$. Кожен нейрон зв'язаний з $10^3 – 10^4$ іншими нейронами і виконує порівняно прості дії. Час спрацювання нейрона – 2–5 мс. Сукупна робота всіх нейронів зумовлює складну роботу мозку, який в реальному часі розв'язує складні задачі.

Кожен біологічний нейрон має відростки нервових волокон двох типів – дендрити (по них приймаються імпульси) і єдиний аксон (по ньому нейрон передає імпульс). Аксон контактує з дендритами інших нейронів через спеціальні утворення – синапси, що впливають на силу імпульса (див. рис. 2).

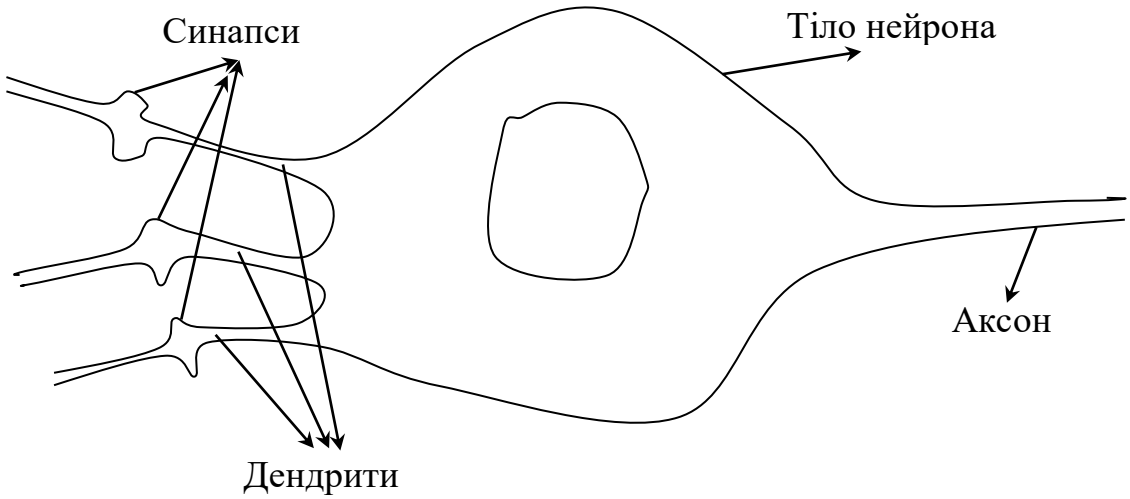


Рис. 2. Біологічний нейрон

При проходженні синапса сила імпульса змінюється відповідно до ваги синапса. Імпульси, що поступили в нейрон одночасно по декількох дендритах, суму-

ються. Якщо сумарний імпульс перевищує деякий поріг, то нейрон збуджується, формує власний імпульс і передає його далі по аксону. Важливо зазначити, що ваги синапсів можуть змінюватися в часі, а отже і змінюється поведінка відповідного нейрона.

Легко побудувати математичну модель описаного процесу (штучний нейрон). На рис. 3 зображено модель штучного нейрона з трьома входами, до того ж синапси цих дендритів мають ваги w_1, w_2, w_3.

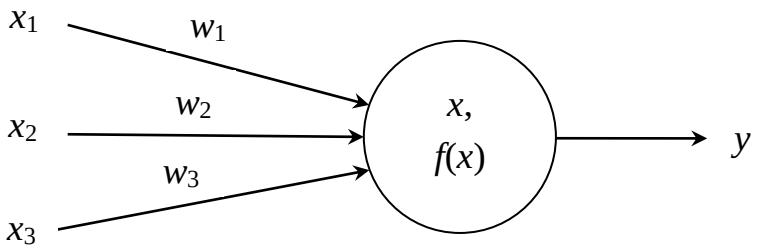


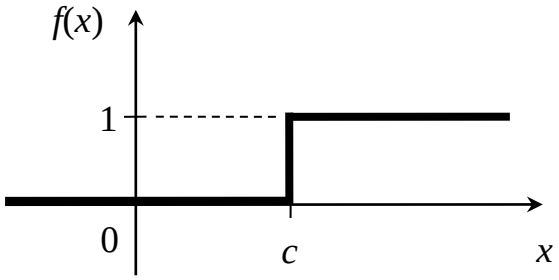
Рис. 3. Штучний (математичний) нейрон

Нехай до синапсів поступають імпульси сили x_1, x_2, x_3 відповідно, тоді після проходження синапсів і дендритів до нейрона поступають імпульси w_1x_1, w_2x_2, w_3x_3. Нейрон перетворює отриманий сумарний імпульс $x = w_1x_1 + w_2x_2 + w_3x_3$ у відповідності з деякою функцією активації нейрона $f(x)$. Сила вихідного імпульсу дорівнює $y = f(x) = f(w_1x_1 + w_2x_2 + w_3x_3)$. Отже, нейрон повністю описується своїми вагами w_k $(k = \overline{1,3})$ і функцією активації $f(x)$.

Найбільш поширеними є такі функції активації:

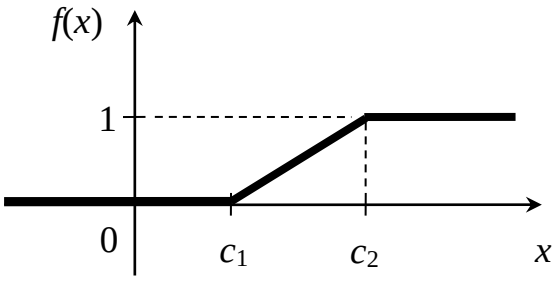
1) порогова функція (сходинка):

$$f(x) = \begin{cases} 0, & x < c; \\ 1, & x \ge c; \end{cases} \qquad c = const;$$



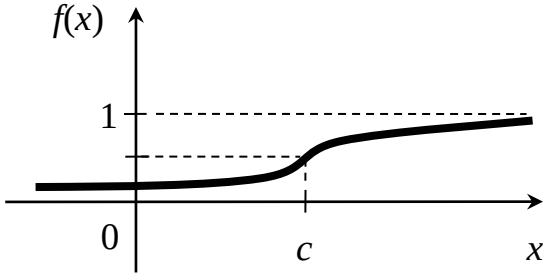
2) лінійна функція:

$$f(x) = \begin{cases} 0, & x < c_1; \\ kx + b, & c_1 \le x < c_2; \\ 1, & x \ge c_2; \end{cases} \qquad k, b, c_1, c_2 = const;$$



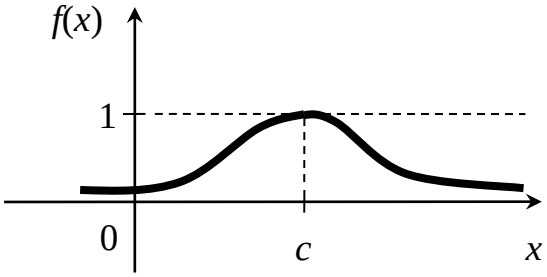
3) сигмоїдальна функція:

$f(x) = (1 + e^{-k(x-c)})^{-1}; \quad k, c = const;$



4) гаусова функція:

$f(x) = e^{-k(x-c)^2}; \quad k, c = const.$



Нейронною мережею будемо називати структуру, що складається із зв'язаних між собою нейронів. Нейронні мережі можуть мати різні архітектури. Залежно від вигляду графа міжнейронних зв'язків такі мережі поділяються на:

- ациклічні мережі (див. рис. 4);
- циклічні мережі (див. рис. 5).

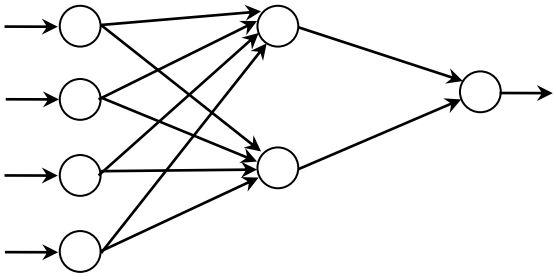


Рис. 4

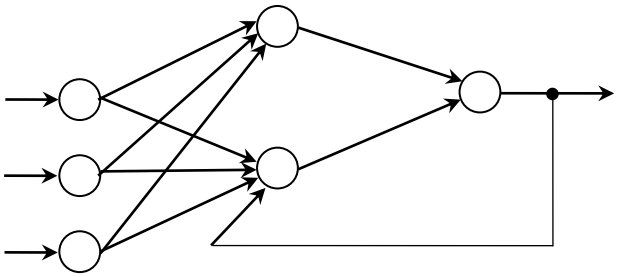


Рис. 5

Досить поширеними є багатошарові нейронні мережі. Крім вхідного і вихідного шарів у таких мережах є один або декілька проміжних (прихованих) шарів. Якщо нейрони кожного шару мають одну функцію активації, то таку **нейронну мережу** будемо називати **однорідною**.

Приклад багатошарової нейронної мережі з **проективними зв'язками** (зв'язки між нейронами сусідніх шарів) подано на рис. 6.

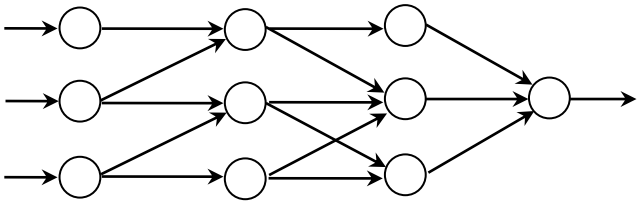


Рис. 6

Приклад багатошарової нейронної мережі з **проективними та латеральними зв'язками** (латеральні зв'язки – це зв'язки між нейронами одного шару) подано на рис. 7.

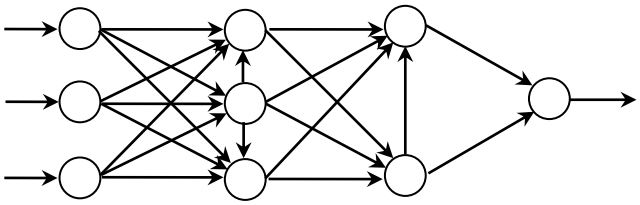


Рис. 7

Нейрокомп'ютером називають обчислювальну систему, архітектура якої орієнтована на виконання операцій, заданих структурою нейронної мережі. Нейрокомп'ютери є доволі дорогими засобами обчислень і масово не випускаються. Однак, апаратна реалізація нейронних мереж може бути виконана на нейрочипах (мікросхеми, що містять фрагменти мережі), на НВІС-пластинах або оптоелектронним способом. Спочатку для реалізації нейромережевих методів використовували наявні послідовні комп'ютери, а зараз – сучасні суперкомп'ютери, оскільки нейронні мережі володіють значними резервами розпаралелювання обробки інформації.

3. Обчислювальні машини потоку даних. За типом керування паралельні обчислювальні системи можна розділити на традиційні обчислювальні машини з

програмно-логічним керуванням та обчислювальні машини потоку даних (Data-flow). Більшість сучасних обчислювальних систем є системами першого типу.

У машинах потоку даних обчислення ініціюються не черговими командами програми, а готовністю до обробки необхідних даних. Операндам кожної команди надаються мітки готовності до обробки – теги:

<КОП> <Адреса результату> <Теги> <Операнди>,

де КОП – код операції. Про готовність операнда сигналізує стан «1» відповідного тегу. Якщо не всі теги операції встановлені в стан «1», то команда знаходиться в стані «очікування». При встановленні всіх тегів у стан «1» команда переводиться в стан «готова до виконання». На кожному такті готові до виконання команди розподіляються комутатором по процесорах (див. рис. 8).

Потенційно такий підхід дозволяє досягти високого ступеня паралелізму. Ефективність потокових машин в більшості визначається програмуванням, від якого вимагається формулювання задачі в термінах паралельних і незалежних операцій.

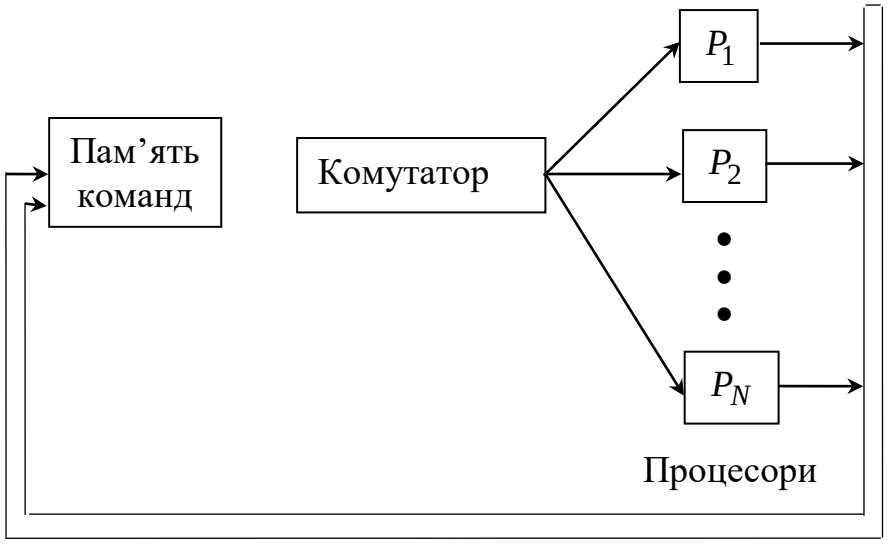


Рис. 8

Ідеологія обчислень, керованих потоком даних, була розроблена в 60-х роках минулого століття Карпом (Karp R.M.) і Міллером (Miller R.E.). На початку 70-х років Денніс, а пізніше і інші почали розробляти відповідні комп’ютерні архітектури (Dennis J.B.).

У Dataflow-архітектурах відсутнє поняття «послідовність інструкцій», немає лічильника команд, відсутня навіть адресована пам’ять у звичному сенсі. Програма в потоковій системі – це не набір команд, а орієнтований граф, інколи він називається графом потоків даних. Цей граф складається із вершин, які відображають операції, та ребер або дуг, які показують потоки даних між тими вершинами графа, які вони з’єднують. Операція у вершині виконується лише тоді, коли по дугах в неї поступила вся необхідна інформація. Зазвичай така операція потребує одного або двох операндів, а для умовних операцій необхідна наявність вхідного логічного значення. Операція формує один, два або один із двох можливих результа-

тів. Отже, у кожної вершини може бути від одної до трьох вхідних дуг і одна або дві, що виходять з неї. Після активації вершини і виконання операції результат передається по дузі до очікуваної вершини. Процес повторюється, поки не будуть активовані всі вершини і отриманий кінцевий результат. Одночасно можуть бути активованими декілька вершин, при цьому паралелізм в обчислювальній моделі виявляється автоматично.

Відомо багато проектів потокових обчислювальних машин, наприклад, такі:

Tagged Token, Monsoon (США);
Sigma, EMS, EMS-4 (Японія).

ЕОМ з архітектурою потоку даних зараз промислово не випускаються. З іншого боку, елементи потокових машин знайшли застосування в сучасних суперскалярних процесорах (мікропроцесори Intel Pentium Pro, HP PA-8000) і процесорах з довгим командним словом (VLIW).