ТЕМА

КЛАСИФІКАЦІЯ И ПРИНЦИПИ ОРГАНІЗАЦІЇ ПАРАЛЕЛЬНИХ КС

ПЛАН:

1. Принципи паралельної обробки завдань
2. Паралельні програми
3. Класифікації архітектур паралельних обчислювальних систем
4. Розширена класифікація Фліна

Висновки

1. ПРИНЦИПИ ПАРАЛЕЛЬНОЇ ОБРОБКИ ЗАВДАНЬ

Усі сучасні комп'ютерні **системи використовують елементи паралельної обробки інформації**: **Багатопроцесорність**, конвеєрна обробка.

- Усі сучасні комп'ютерні системи використовують розподілені обчислення: **Багатозадачність**, бази даних, файлові сервера
- Деякі завдання можна сьогодні вирішити тільки за допомогою паралельних і розподілених обчислень
  - отримання «надзвичайно» високої продуктивності;
  - отримання високої надійності і відмовостійкості;
  - Деякі ресурси розподілені по визначенню.

1. **Принципи паралельної обробки завдань.**

Паралельна обробка завдань

Наукові та промислові завдання, що вимагають паралельних обчислень:

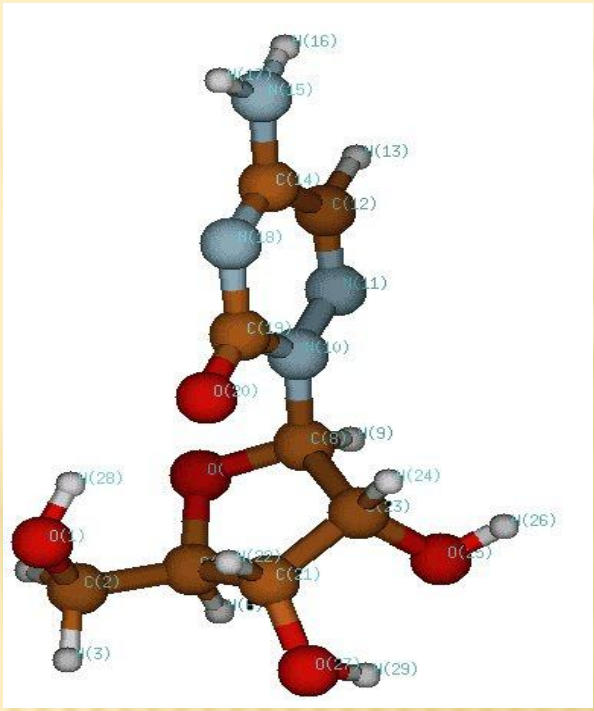
- Квантова фізика, хімія, молекулярна біологія;
- Мікроелектроніка;
- Статистичне моделювання (метод Монте-Карло);
- Ядерна фізика.

Один з основних факторів, який визначає розвиток обчислювальної техніки в цілому і обчислювальних систем зокрема, це висока ***продуктивність***.

Загальний метод збільшення продуктивності - організація паралельної обробки інформації, тобто одночасне вирішення завдань або суміщення в часі етапів розв'язання однієї задачі.

1. **Принципи паралельної обробки завдань.**

Приклад паралельної обробки завдань: Хімія



Є формула речовини (лікарський препарат), знайти, як ця речовина вступає в реакцію, наскільки вона стійка і як діє.

Властивості речовини визначаються типом атомів, положенням ядер і електронною конфігурацією.

Для знаходження електронної конфігурації необхідно вирішувати рівняння квантової фізики.

Кількість операцій, і обсяг оперативної пам'яті, необхідні для вирішення визначаються числом електронів молекули N.

Кількість операцій пропорційно N^4-N^7

Об'єм оперативної пам'яті пропорційний N^3- N^4

1. **Принципи паралельної обробки завдань.**

Оцінка часу і ресурсів

- Кількість атомів 29
- Кількість електронів N = 130
- Кількість базисних функцій 280
- Кількість операцій $N^5 \sim 10^{13}$
- Завдання необхідно вирішувати десятки / сотні разів ~ 10^{15}
- Час на процесорі продуктивністю 1 млрд.
- Операцій в секунду близько тижня
- Пам'яті майже 4 Гбайт
- Необхідно кілька процесорів

У всьому різноманітті способів організації паралельної обробки можна ***виділити три основні напрями:***

1) поєднання в часі різних етапів різних завдань;
2) одночасне рішення різних завдань або частин одного завдання;
3) конвеєрна обробка інформації.

Перший шлях - поєднання в часі етапів рішення різних задач - це мультипрограмна обробка інформації. Мультипрограмна обробка можлива навіть в однопроцесорній ЕОМ і широко використовується в сучасної СОД (системи обміну даними).

Другий шлях - одночасне рішення різних завдань або частин одного завдання - можливий тільки за наявності декількох оброблювальних пристроїв.

1. **Принципи паралельної обробки завдань.**

РІЗНОВИДИ ПАРАЛЕЛІЗМУ

Природний паралелізм незалежних завдань полягає в тому, що в систему надходить безперервний потік не пов'язаних між собою завдань, тобто рішення будь-якої задачі не залежить від результатів вирішення інших завдань**.**

Паралелізм незалежних гілок. Суть його полягає в тому, що при вирішенні великого завдання можуть бути виділені окремі незалежні частини-гілки програми, які при наявності декількох оброблювальних пристроїв можуть виконуватися паралельно і незалежно один від одного.

Паралелізм незалежних гілок

Двома незалежними гілками програми вважатимемо такі частини завдання, при виконанні яких виконуються наступні умови:

- ❖жодна з вхідних для гілки програми величин не являється вихідною величиною іншої програми (відсутність функціональних зв'язків);
- ❖для обох гілок програми не повинен робитися запис в одні і ті ж елементи пам'яті (відсутність зв'язку по використанню одних і тих же полів оперативної пам'яті);
- ❖умови виконання однієї гілки не залежать від результатів або ознак, отриманих при виконанні іншої гілки (незалежність по управлінню);
- ❖обидві гілки повинні виконуватися по різних блоках програми (програмна незалежність).

1. **Принципи паралельної обробки завдань.**

Ярусно-паралельна форма програми

Добре уявлення про паралелізм незалежних гілок дає ярусно-паралельна форма програми, приклад якої наведено на рис. 1.

Програма представлена у вигляді сукупності гілок, розташованих в декількох рівнях - ***ярусах.***

Для того щоб за допомогою декількох пристроїв, які обробляють інформацію, вирішити завдання, що має незалежні паралельні гілки, необхідна відповідна організація процесу, яка визначає шляхи вирішення завдання і виробляє необхідну інформацію про готовність кожної гілки.

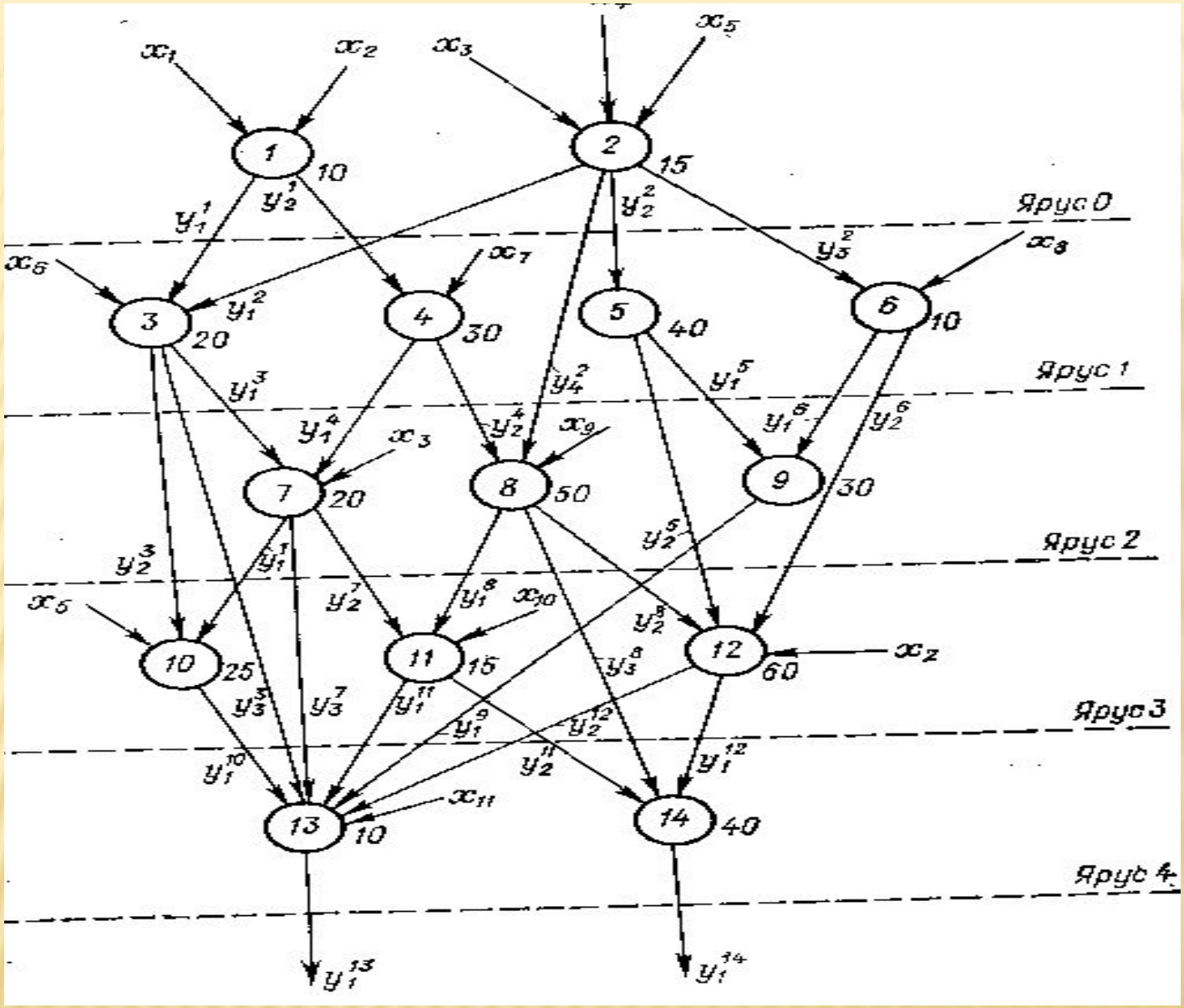


Рисунок 1 - Ярусно-паралельна форма програми

1. **Принципи паралельної обробки завдань.**

При вирішенні багатьох складних завдань лише програмування з виділенням незалежних гілок дозволяє істотно скоротити час рішення.

Добре піддаються паралельній обробці такого типу завдання матричної алгебри, лінійного програмування, спектральної обробки сигналів, прямі і зворотні перетворення Фур'є і ін.

Паралелізм об'єктів або даних має місце тоді, коли по одній і тій же (або майже по одній і тій же) програмі повинна оброблятися деяка сукупність даних, що надходять в систему одночасно.

Це можуть бути, наприклад, завдання обробки сигналів від радіолокаційної станції: всі сигнали обробляються по одній-і тій же програмі.

1. **Принципи паралельної обробки завдань.**

Конвеєрна обробка

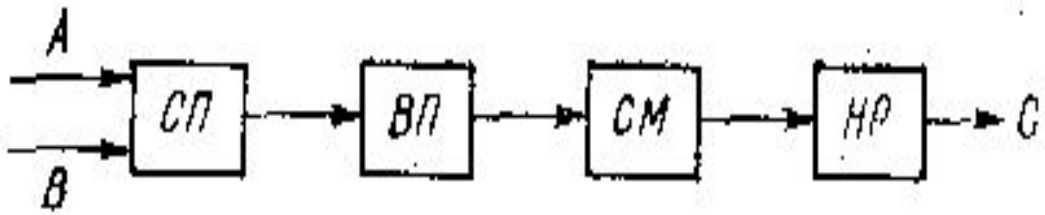
Третій шлях паралельної обробки інформації - конвеєрна обробка - може бути реалізований в системі і з одним процесором, розділеним на деяке число послідовно включених операційних блоків, кожен з яких спеціалізується на виконанні чітко визначеної частини операції.

Конвеєр операцій:

$$A + B = \left[a_i \cdot 2^x\right] + \left[b_i \cdot 2^y\right] = \left[c_i 2^{x/y}\right]$$

Конвеєрна обробка

а)



б)

Этап	1	2	3	4	5	6	...	i	...	n	n+1	n+2	n+3
СП	$a_1 b_1$	$a_2 b_2$	$a_3 b_3$	$a_4 b_4$	$a_5 b_5$	$a_6 b_6$	...	$a_i b_i$	...	$a_n b_n$			
ВП		$a_1 b_1$	$a_2 b_2$	$a_3 b_3$	$a_4 b_4$	$a_5 b_5$	...	$a_{i-1} b_{i-1}$	...	$a_{n-1} b_{n-1}$	$a_n b_n$		
СМ			$a_1 b_1$	$a_2 b_2$	$a_3 b_3$	$a_4 b_4$	...	$a_{i-2} b_{i-2}$	...	$a_{n-2} b_{n-2}$	$a_{n-1} b_{n-1}$	$a_n b_n$	
НР				c_1	c_2	c_3	...	c_{i-3}	...	c_{n-3}	c_{n-2}	c_{n-1}	c_n

Рисунок 2 – Структурна схема (а) та часова діаграма (б) конвеєра операцій

1. **Принципи паралельної обробки завдань.**

Приклад конвеєра команд

Этап	1	2	3	4	5	6	7	8
ФАК	K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_8
ВК		K_1	K_2	K_3	K_4	K_5	K_6	K_7
РКО			K_1	K_2	K_3	K_4	K_5	K_6
ФАО				K_1	K_2	K_3	K_4	K_5
ВО					K_1	K_2	K_3	K_4
АЛО						K_1	K_2	K_3

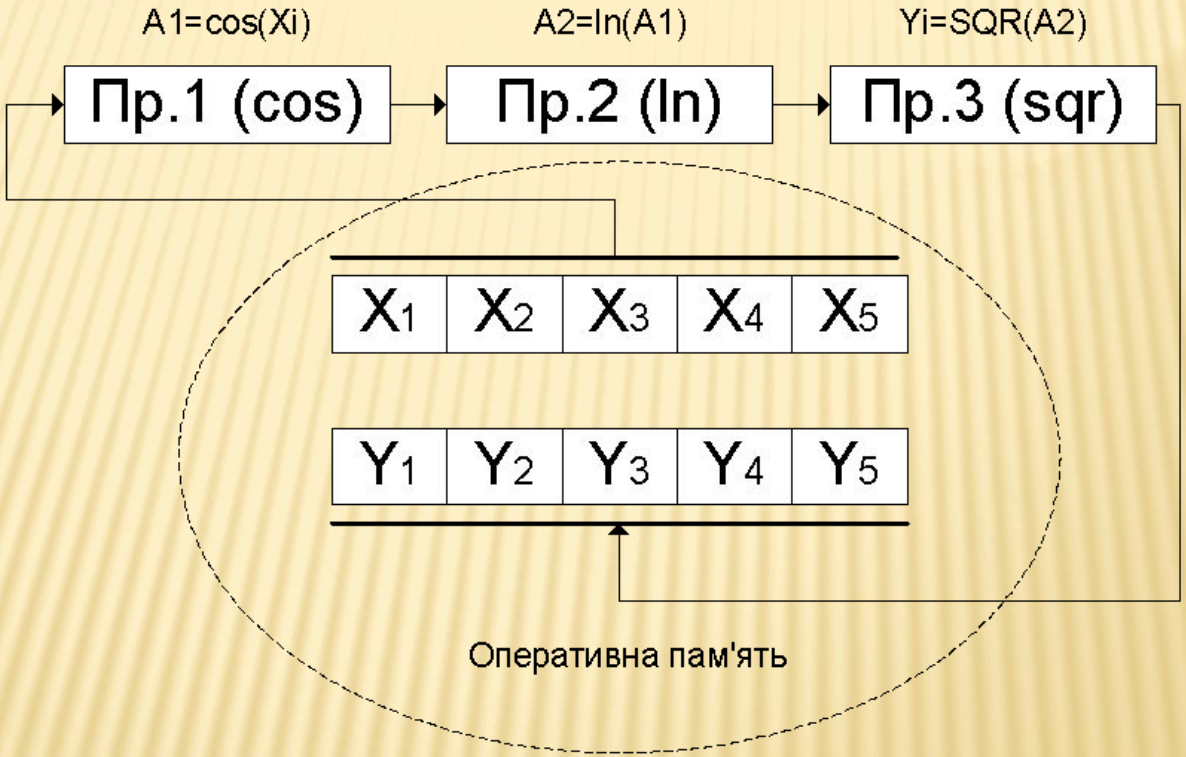
1. **Принципи паралельної обробки завдань.**

Приклад конвеєра команд

$$y = \sqrt{\ln(\sin(x))}$$

На вході 5 значень (X1, X2, …. X5)

В системі 3 процесора, кожен виконує свої дії



Результат с виходу одного процесора передаємо на вхід наступного.

1. **Принципи паралельної обробки завдань.**

В обчислювальних системах можна одночасно використовувати і конвеєр команд, і конвеєр арифметичних операцій, і навіть кілька паралельно працюючих конвеєрів команд і арифметичних операцій.

Стани конвеєра

На трьох процесорах результат виходить за 8 кроків замість 15 на одному процесорі

	Процесор 1 sin	Процесор 2 ln	Процесор 3 sqr	Y1	Y2	Y3	Y4	Y5
1	sin(X1)	-	-	-	-	-	-	-
2	sin(X2)	ln(sin(X1))	-	-	-	-	-	-
3	sin(X3)	ln(sin(X2))	sqr(ln(sin(X1))	Y1	-	-	-	-
4	sin(X4)	ln(sin(X3))	sqr(ln(sin(X2))	Y2	Y1	-	-	-
5	sin(X5)	ln(sin(X4))	sqr(ln(sin(X3))	Y3	Y2	Y1	-	-
6	sin(X5)	ln(sin(X4))	sqr(ln(sin(X3))	Y3	Y2	Y1	-	-
7	-	ln(sin(X5))	sqr(ln(sin(X4))	Y4	Y3	Y2	Y1	-
8	-	-	sqr(ln(sin(X5))	Y5	Y4	Y3	Y2	Y1

1. **Принципи паралельної обробки завдань.**

2. ПАРАЛЕЛЬНІ ПРОГРАМИ

Паралельні програми

На рубежі зміни століть парк обчислювальних засобів для вирішення прикладних завдань представлений великою різноманітністю обчислювальних систем.

Ці системи укомплектовані процесорами з тактовою частотою до декількох гігагерц, мережевою периферією для обміну даними з пропускною спроможністю до декількох гігабайт в секунду, локальними і мережевими операційними системами для управління рішенням завдань і підтримки взаємодіючих обчислювальних процесів.

Принципові фізичні обмеження обробки даних

Науково-дослідні роботи, роботи з проектування складних багатофункціональних технічних пристроїв, економічні дослідження, моделювання нестаціонарних динамічних процесів в фізичних системах і т. д. виставляють такі вимоги до швидкостей отримання рішень, які на одиночному надшвидкісний комп'ютері не можуть бути виконані за розумно-найближчий час.

До того ж, твердотільна електрична основа, яка використовується для схемотехнічних обчислювальних пристроїв, вносить принципові фізичні обмеження на подальше істотне збільшення швидкодії базових перемикаючих елементів.

Паралельне програмування

Виникла задача програмування безлічі процесів, які протікають одночасно і взаємодіють один з одним.

З'явилася потреба в розвитку мов високого рівня для запису програм процесів, які протікають паралельно, побудови операційних систем для їх компіляції, налагодження, виконання паралельних програм, документування результатів тощо.

Весь комплекс підготовчих дій, викликаний розробкою і виконанням обчислювальних завдань в багатомашиному обчислювальному середовищі, з метою досягнення максимально швидкого вирішення громіздкого завдання отримав назву ***"паралельне програмування"***

Основні етапи підготовки паралельної програми

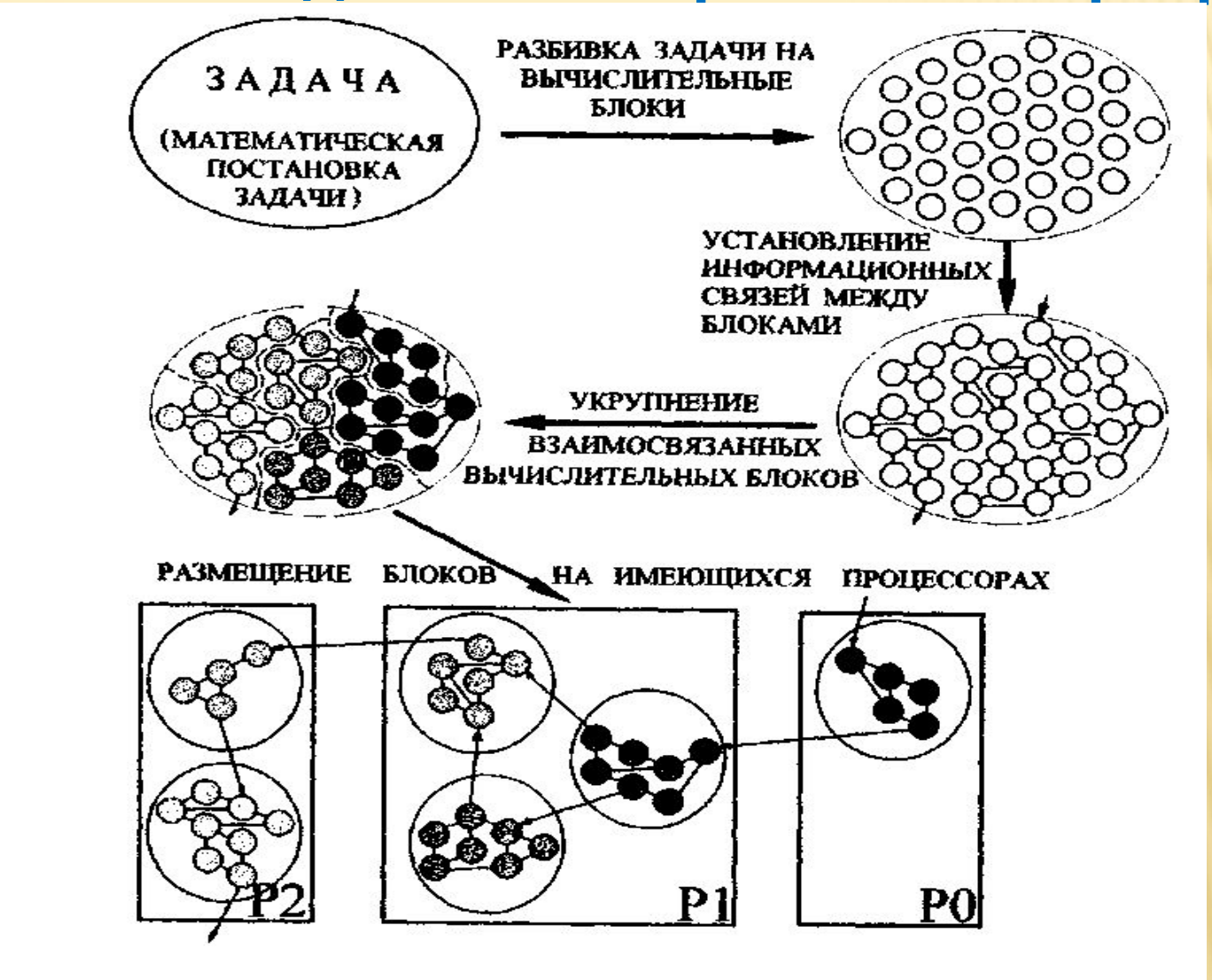


Рисунок 2 - Основні етапи підготовки паралельної програми

Розподілена обробка даних

У дисциплінах з програмування задач і розподіленої обробки даних виділяють три різних поняття: програма, процесор і процес. Ці поняття в літературі зазвичай пов'язують з розподіленим програмуванням і розподіленою обробкою даних.

Під ***системою розподіленого опрацювання даних*** розуміється така система, окремі компоненти якої одночасно функціонують на різних машинах, що володіють засобами обміну даними один з одним.

Розподілене програмування (distributed programming) - це сукупність мовних засобів і методів програмування систем розподіленої обробки даних в мережах ЕОМ і багатомашинних комплексах, обчислювальних кластерах.

3. КЛАСИФІКАЦІЇ АРХІТЕКТУР ПАРАЛЕЛЬНИХ ОБЧИСЛЮВАЛЬНИХ СИСТЕМ

Існують різні класифікації архітектур обчислювальних систем:

1. **Класифікація Флінна**: єдність або множинність потоків даних і команд.
1.1. Доповнення Ванга та Бріггса: конкретизація класів SISD, SIMD, MIMD.
2. **Класифікація Фенга**: дві прості чисельні характеристики паралелізму (послівний і порозрядного паралелізм).
3. **Класифікація Шора**: шість "типових архітектур" обчислювальних систем.
4. **Класифікація Хендлера**: кількісний опис паралелізму на трьох різних рівнях обробки даних (виконання програми, виконання команд, обробка бітів).
5. **Класифікація Хокні**: конкретизація класу MIMD.

Існують різні класифікації архітектур обчислювальних систем:

6. **Класифікація Шнайдера**: конкретизація класу SIMD (основна ідея - виділення етапів вибірки і безпосередньо виконання в потоках команд і даних).
7. **Класифікація Джонсона**: чотири класи MIMD-комп'ютерів (комп'ютери із загальною або розподіленою пам'яттю, програмовані за допомогою передачі повідомлень або поділюваних змінних).
8. **Класифікація Базу**: послідовність рішень, прийнятих на етапі проектування архітектури.
9. **Класифікація Крішнамарфі**: чотири якісні характеристики паралелізму (ступінь гранулярності паралелізму, спосіб реалізації, топологія і природа зв'язку процесором, спосіб управління процесорами).

Існують різні класифікації архітектур обчислювальних систем:

10. **Класифікація Скіллікорна**: опис архітектури комп'ютера як абстрактної структури, що складається з компонент 4 типів (процесор команд, процесор даних, ієрархія пам'яті, комутатор).
11. **Класифікація Дазгупти**: побудова схем архітектур з семи базових понять.
12. **Класифікація Дункана**.

Класифікація Флінна

У 1966 році М.Флінном (Flynn) був запропонований надзвичайно зручний підхід до класифікації архітектур обчислювальних систем.

В основу було положено поняття потоку, під яким розуміється послідовність елементів, команд або даних, які обробляються процесором. Відповідна система класифікації заснована на розгляді числа потоків інструкцій та потоків даних і описує чотири архітектурні класи:

SISD = Single Instruction Single Data
MISD = Multiple Instruction Single Data
SIMD = Single Instruction Multiple Data
MIMD = Multiple Instruction Multiple Data.

SISD (single instruction stream / single data stream)

SISD (single instruction stream / single data stream) - одиночний потік команд і одиночний потік даних. До цього класу належать послідовні комп'ютерні системи, які мають один центральний процесор, здатний обробляти тільки один потік послідовно виконуваних інструкцій.

В даний час практично всі високопродуктивний системи мають більше одного центрального процесора, однак, кожен з них виконують незв'язані потоки інструкцій, обробні незалежні потоки даних.

Такий комп'ютер є набором SISD-машин, що працюють з незалежною безліччю даних.

stream)

Для збільшення швидкості обробки команд і швидкості виконання арифметичних операцій може застосовуватися конвеєрна обробка.

У разі векторних систем векторний потік даних слід розглядати як потік з одиночних неподільних векторів.

Прикладами комп'ютерів з архітектурою SISD являются біl q, Hewlett-Pack

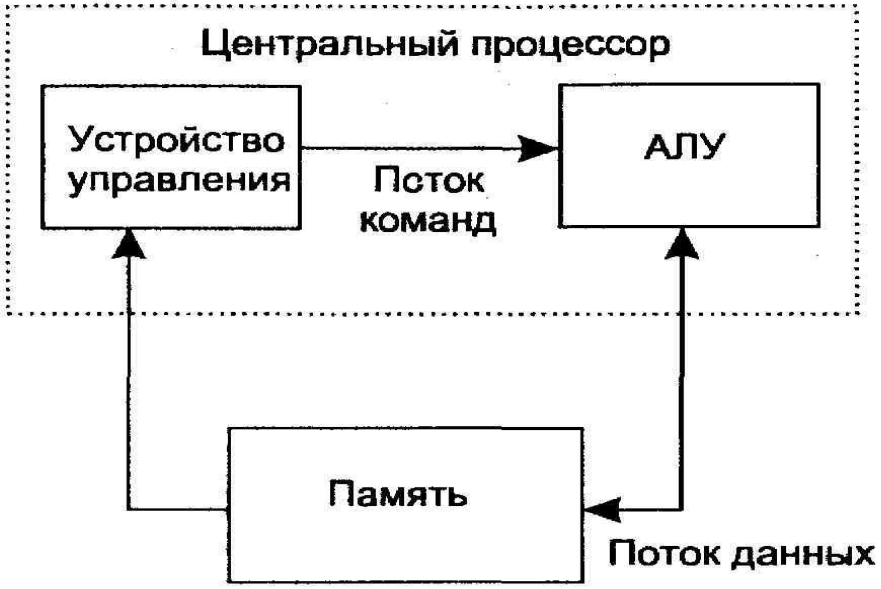


Рис. 1.1.

SIMD (single instruction stream / multiple data stream)

SIMD (single instruction stream / multiple data stream) - одиночний потік команд і множинний потік даних.

Ці системи зазвичай мають велику кількість процесорів, в межах від 1024 до 16384, які можуть виконувати одну і ту ж інструкцію щодо різних даних в жорсткій конфігурації.

Єдина інструкція паралельно виконується над багатьма елементами даних. Прикладами SIMD машин є системи CPP DAP, Gamma II і Quadrics Apemille. Іншим підкласом SIMD-систем є векторні комп'ютери.

Векторні комп'ютери маніпулюють масивами подібних даних подібно до того, як скалярні машини обробляють окремі елементи таких масивів. Це робиться за рахунок використання спеціально сконструйованих векторних центральних процесорів.

SIMD (single instruction stream / multiple data stream)

При роботі у векторному режимі векторні процесори обробляють дані практично паралельно, що робить їх в кілька разів швидшими, ніж при роботі в скалярному режимі.
В SIMD-комп'ютері управління виконується контролером, а «арифметика» віддана процесорним елементам (ПЕ).
Прикладами систем подібного типу є, наприклад, комп'ютери Hitachi S3600.

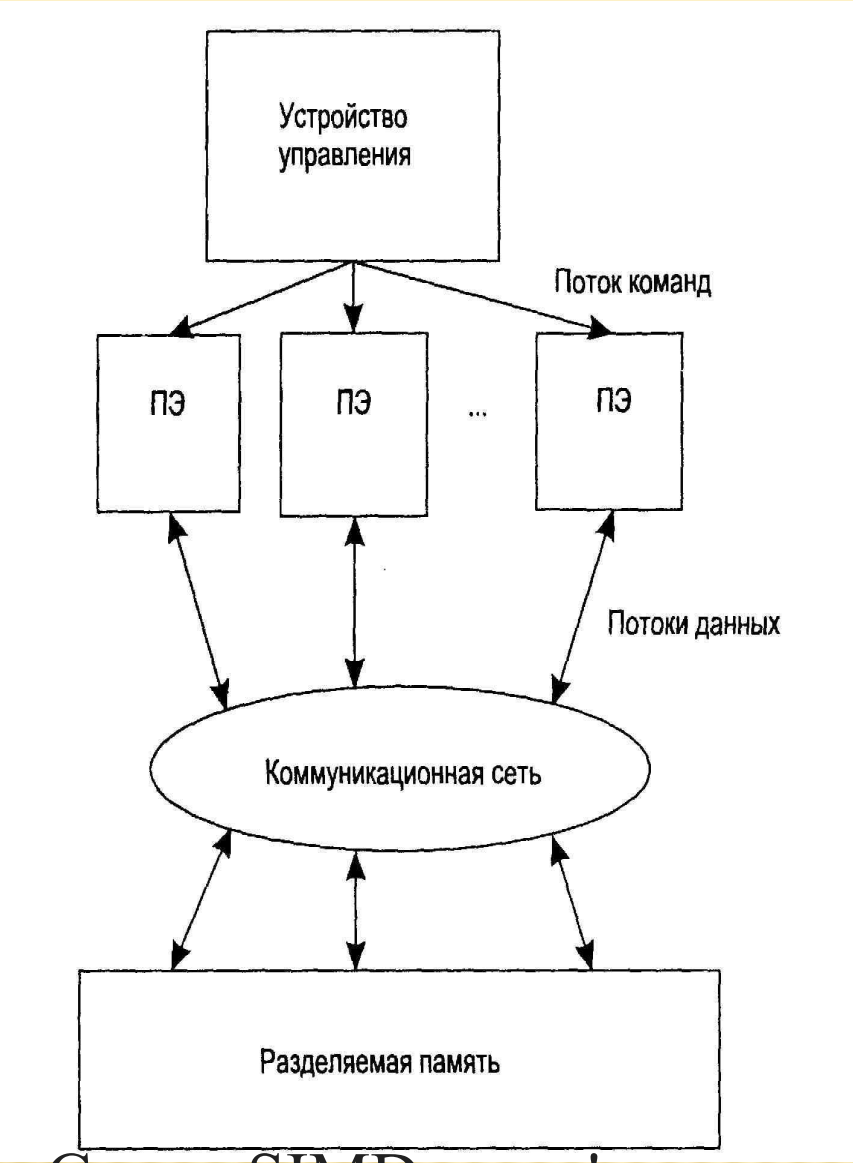


Схема SIMD-комп'ютера з роздільною пам’яттю

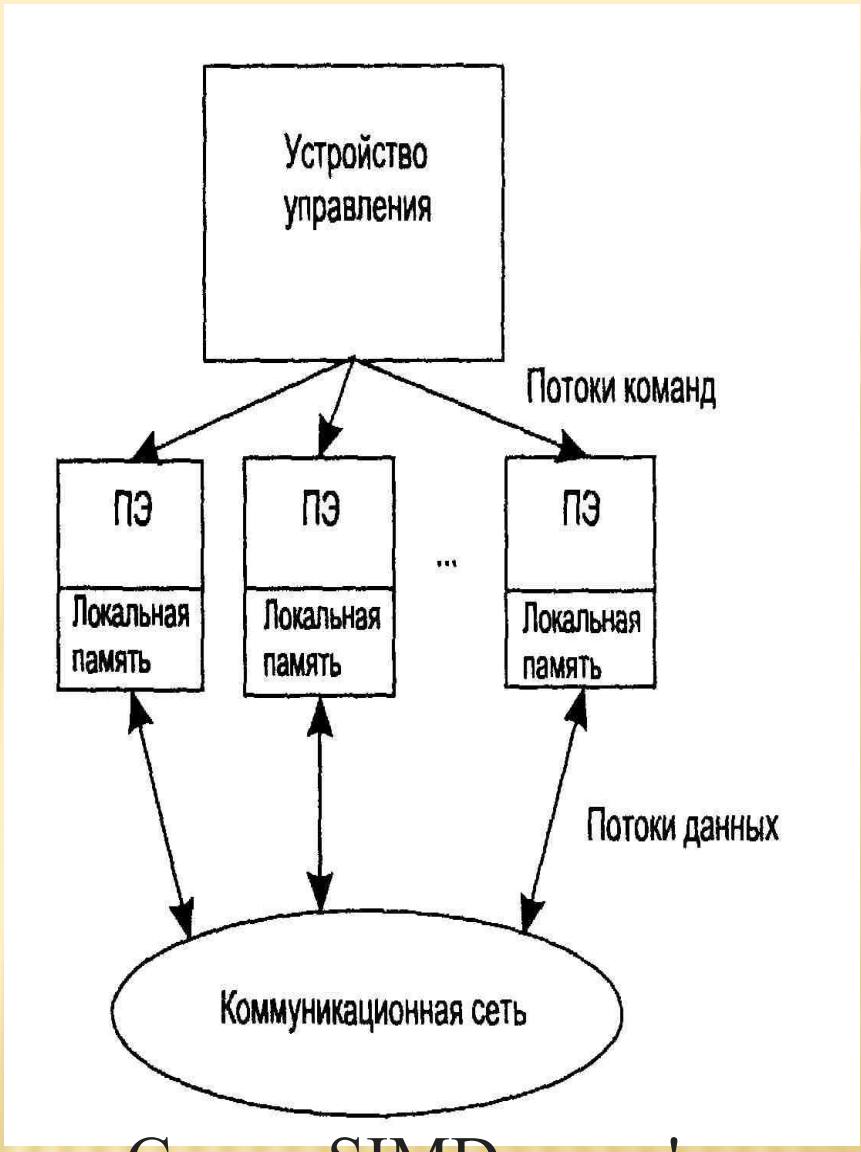


Схема SIMD-комп'ютера з розподільною пам'яттю

3. Класифікації архітектур паралельних обчислювальних систем

MISD (multiple instruction stream / single data stream)

MISD (multiple instruction stream / single data stream) - множинний потік команд і одиночний потік даних. Теоретично в цьому типі машин безліч інструкцій повинно виконуватися над єдиним потоком даних. До сих пір жодної реальної машини, що потрапляє в даний клас, не було створено. Як аналог роботи такої системи, мабуть, можна розглядати роботу банку.

З будь-якого терміналу можна подати команду і щось зробити з наявним банком даних. Оскільки база даних одна, а команд багато, то ми маємо справу з множинним потоком команд і одиночним потоком

MIMD (multiple instruction stream / multiple data stream)

MIMD (multiple instruction stream / multiple data stream) - множинний потік команд і множинний потік даних.

Ці машини паралельно виконують кілька потоків інструкцій над різними потоками даних.

На відміну від багатопроцесорних SISD-машин, згаданих вище, команди і дані пов'язані, тому що вони представляють різні частини одного і того ж виконуваного завдання.

Наприклад, MIMD-системи можуть паралельно виконувати безліч підзадач, з метою скорочення часу виконання основного завдання.

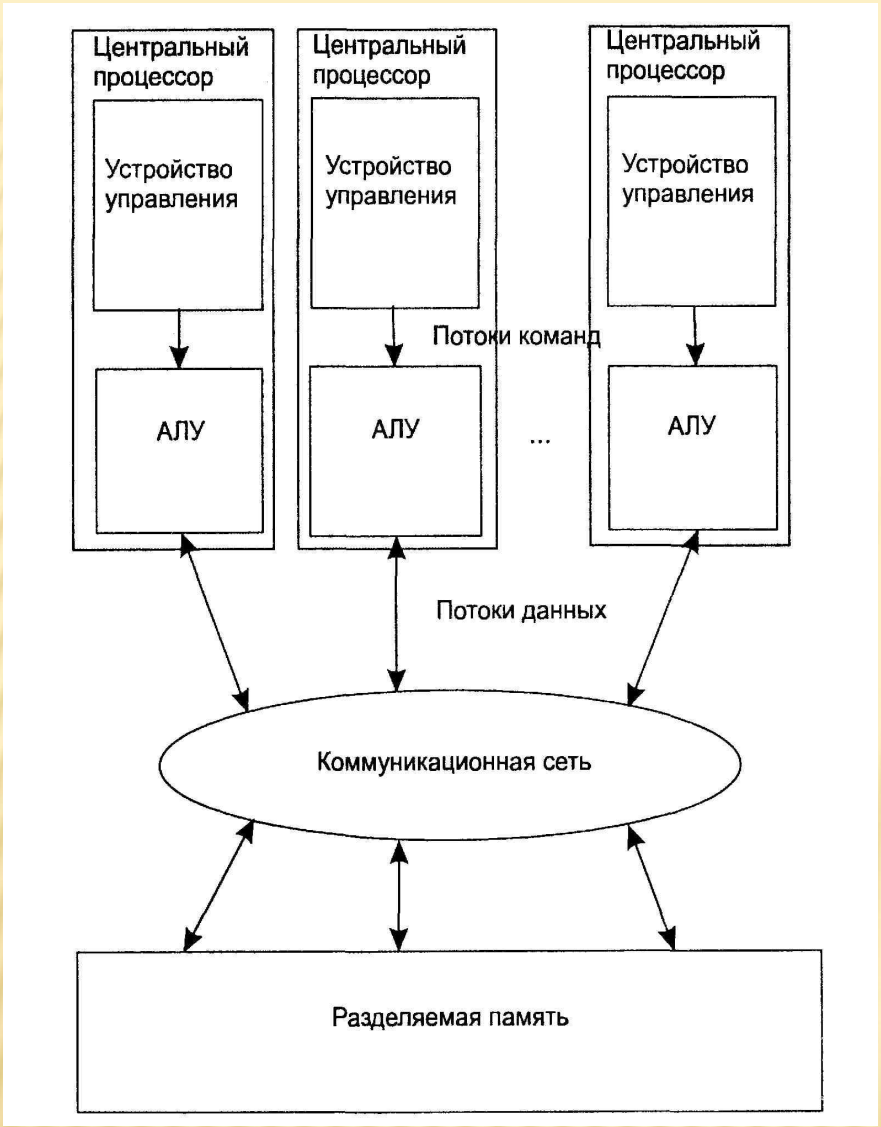


Схема MIMD-комп'ютера з роздільною пам’яттю

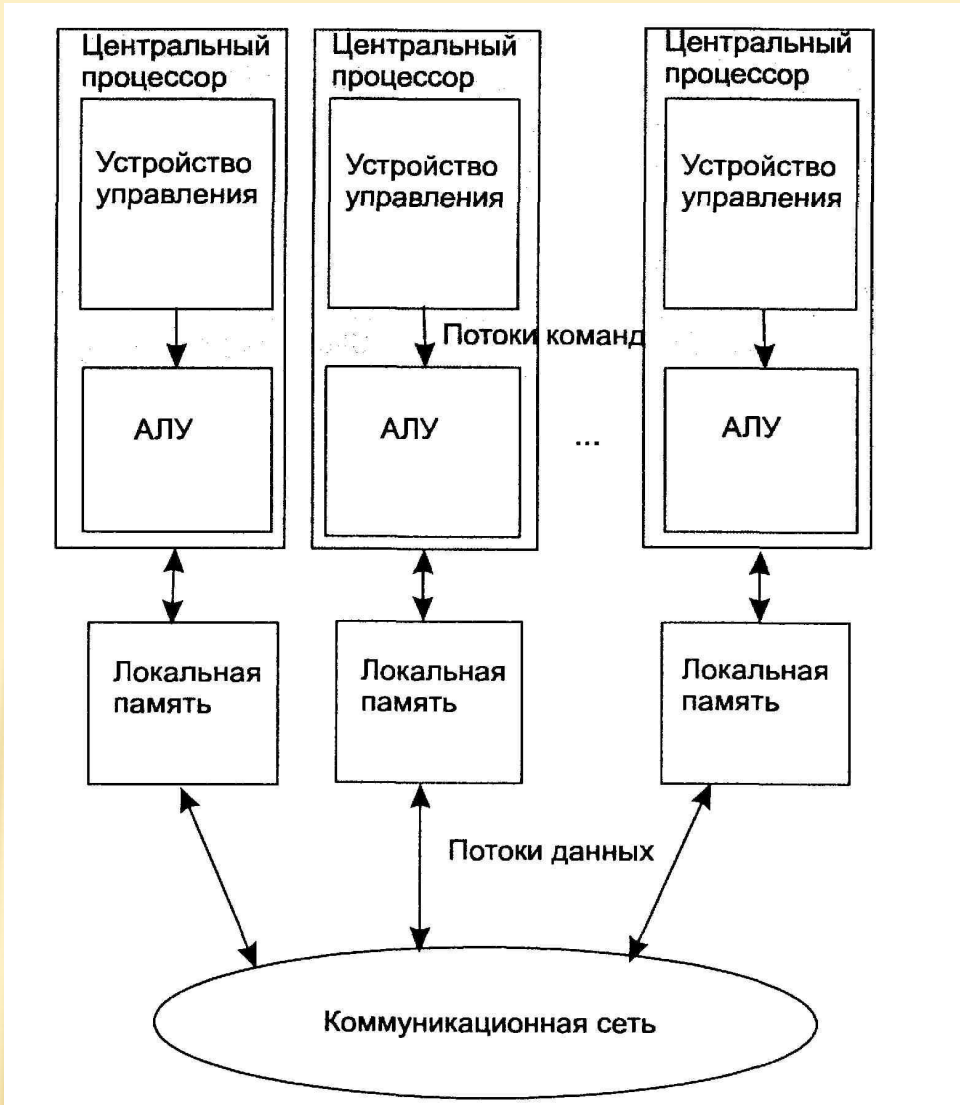


Схема MIMD-комп'ютера з розподіленою пам'яттю

3. Класифікації архітектур паралельних обчислювальних систем

Класифікація націнок систем в залежності від їх розмірів

Межпроцессорное расстояние	Процессоры расположены	Пример
0,1 м	На одной плате	Потоковая вычислительная машина
1 м	В одной системе	Мультикомпьютер
10 м	В одной комнате	Локальная сеть
100 м	В одном здании	
1 км	На одной территории студенческого городка	
10 км	В одном городе	Региональная сеть
100 км	В одной стране	Глобальная сеть
1000 км	На одном континенте	
10 000 км	На одной планете	Интернет

Рисунок - Класифікація багатопроцесорних систем за розміром

3. Класифікації архітектур паралельних обчислювальних систем

4. РОЗШИРЕНА КЛАСИФІКАЦІЯ ФЛІНА

На сьогоднішній день, хорошої класифікації комп'ютерів паралельної дії до сих пір не існує. Найчастіше використовується класифікація Флінна (Flynn), але навіть вона є дуже грубим наближенням.

Машини SIMD розпалися на дві підгрупи. *До першої підгрупи* входять численні суперкомп'ютери та інші машини, які оперують векторами, виконуючи одну і ту ж операцію над кожним елементом вектора. *В другу підгрупу* - машини типу ILLIAC IV, в яких головний блок управління посилає команди декільком незалежним АЛУ.У нашій класифікації категорія MIMD розпалася на мультипроцесори (машини з пам'яттю спільного використання) і мультикомп’ютери (машини з передачею повідомлень). *Існує три типи мультипроцесорів*. Вони відрізняються один від одного за способом реалізації пам'яті спільного використання.

4. Розширена класифікація Фліна

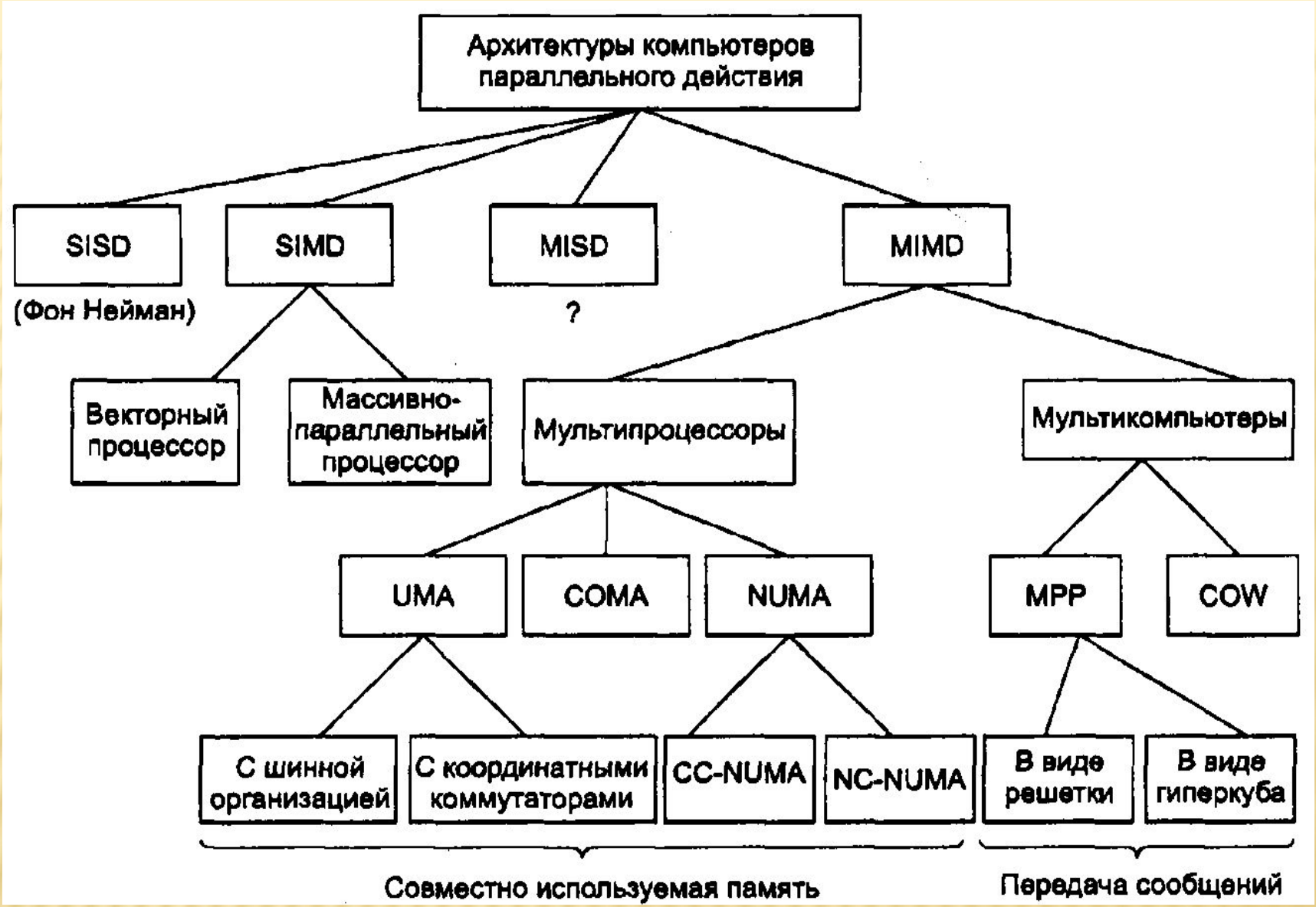


Рисунок - Класифікація комп'ютерів паралельної дії

4. Розширена класифікація Фліна

UMA (Uniform Memory Access - архітектура з однорідним доступом до пам'яті), кожне слово пам'яті можна вважати з тією ж швидкістю, що і будь-яке інше слово пам'яті.

NUMA (NonUniform Memory Access - архітектура з неоднорідним доступом до пам'яті) і **COMA** (Cache Only Memory Access - архітектура з доступом тільки до кеш-пам'яті).

В машинах UMA кожен процесор має один і той же час доступу до будь-якого модулю пам'яті.

Така однорідність робить продуктивність передбачуваною, а цей фактор дуже важливий для написання ефективної програми.

4. Розширена класифікація Фліна

Мультипроцесори NUMA і COMA

Мультипроцесор NUMA, навпаки, не володіє цією властивістю.

Зазвичай є такий модуль пам'яті, який розташований близько до кожного процесору, і доступ до цього модулю пам'яті відбувається набагато швидше, ніж до інших.

З точки зору продуктивності дуже важливо, куди поміщаються програма і дані.

Машини COMA теж з неоднорідним доступом, але з іншої причини.

Мультикомп’ютери

У другу підкатегорію машин MIMD потрапляють мультикомп’ютери, які на відміну від мультипроцесорів не мають пам'яті спільного використання на архітектурному рівні.

Іншими словами, операційна система в процесорі мультикомп’ютера не може отримати доступ до пам'яті, що відноситься до іншого процесору, просто шляхом виконання команди LOAD.

Йому доводиться відправляти повідомлення і чекати відповіді. Саме здатність операційної системи зчитувати слово з віддаленого модуля пам'яті за допомогою команди LOAD відрізняє мультипроцесори від мультикомп’ютерів.

Так як мультикомп’ютери не мають прямого доступу до віддалених модулів пам'яті, вони іноді називаються машинами **NORMA** (NO Remote Memory Access - без доступу до віддалених модулів пам'яті).
Мультикомп’ютери можна розділити на дві категорії:
Перша категорія містить процесори MPP (Massively Parallel Processors - процесори з масовим паралелізмом) - дорогі суперкомп'ютери, які складаються з великої кількості процесорів, пов'язаних високошвидкісною комунікаційною мережею. Як приклади можна назвати Cray T3E і IBM SP / 2.
Друга категорія мультикомп’ютерів включає робочі станції, які зв'язуються з допомогою вже існуючої технології з'єднання. Ці примітивні машини називаються NOW (Network of Workstations - мережа робочих станцій) і COW (Cluster ofWorkstattions - кластер робочих станцій)

4. Розширена класифікація Фліна

Способи побудови міжз’єднань в мультикомп’ютерах і мультипроцесорах.

У мультикомп’ютерах вузли обробки даних об'єднуються за допомогою різних мережевих технологій.

У мультипроцесорах для цих цілей використовуються високошвидкісні вузли зв'язку, що отримали назву комутаційні середовища.

Найпростішими комутаційними середовищами є шини і комутатори.

Для зменшення витрат комутаційних елементів застосовуються структури з більш складною топологією: багатовимірні куби, тори, дерева.

4. Розширена класифікація Фліна

Апаратні засоби паралельних обчислень

Для паралельних обчислень потрібно кілька процесорів або комп'ютерів Кілька процесорів / комп'ютерів завжди в сумі дорожче, ніж один процесор / комп'ютер

Необхідне забезпечення високошвидкісних каналів зв'язку між процесорами / комп'ютерами

Зі збільшенням кількості і складності устаткування часто зменшується його надійність

Приклади паралельних систем

Кластер Київського національного Університету імені Тараса Шевченка

Приклади паралельних систем

Кластери Інституту кібернетики НАН України імені Глушкова

Приклади паралельних систем

Китайський суперкомп'ютер: Tianhe-2 розроблений в 2013р. Найпотужніший у світі.

Тяньхе-2 складається з 16 тисяч вузлів, кожен з яких включає 2 процесора Intel Xeon E5-2692 на архітектурі Ivy Bridge з 12 ядрами кожен (частота 2,2 ГГц) і 3 спеціалізованих співпроцесора Intel Xeon Phi 31S1P (на архітектурі Intel MIC, по 57 ядер на прискорювач, частота 1,1 ГГц, пасивне охолодження). 33,86 Пфлопс

На кожному вузлі встановлено 64 ГБ (16 модулів) оперативної пам'яті типу DDR3 ECC і додатково по 8 ГБ GDDR5 в кожному Xeon Phi (всього 88 ГБ).

ВИСНОВОК

1. Загальний метод збільшення продуктивності - організація паралельної обробки інформації, тобто одночасне вирішення завдань або суміщення в часі етапів розв'язання однієї задачі.

2.У всьому різноманітті способів організації паралельної обробки можна виділити три основні напрямки:

поєднання в часі різних етапів різних завдань;

одночасне рішення різних завдань або частин однієї задачі;

конвеєрна обробка інформації.

ВИСНОВОК

3.Параллелізм незалежних гілок - один з найбільш поширених типів паралелізму в обробці інформації. Суть його полягає в тому, що при вирішенні великої задачі можуть бути виділені окремі незалежні частини-гілки програми, які при наявності декількох оброблювальних пристроїв можуть виконуватися паралельно і незалежно один від одного.

4. Для того щоб за допомогою декількох оброблюючих пристроїв вирішити завдання, що має незалежні паралельні гілки, необхідна відповідна організація процесу, яка визначає шляхи вирішення задачі і виробляє необхідну інформацію про готовність кожної гілки.

ВИСНОВОК

5. Добре піддаються паралельній обробці такого типу задачі матричної алгебри, лінійного програмування, спектральної обробки сигналів, прямі і зворотні перетворення Фур'є і ін.
6. Конвеєрна обробка - може бути реалізований в системі і з одним процесором, розділеним на деяке число послідовно включених операційних блоків, кожен з яких спеціалізується на виконанні чітко визначеної частини операції.
7. Наявність умовних переходів порушує роботу конвеєра і призводить до його «холостих» пробігів.