МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Г.М. Луцький,
О. В. Русанова

КОМП'ЮТЕРНІ СИСТЕМИ
ЛАБОРАТОРНИЙ ПРАКТИКУМ

Навчальний посібник

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для здобувачів ступеня бакалавра
за освітньою програмою «Комп'ютерні системи та мережі»
спеціальності 123 Комп'ютерна інженерія

Електронне мережне навчальне видання

Київ
КПІ ім. Ігоря Сікорського
2022

Автори — Луцький Георгій Михайлович, професор, д.т.н., професор кафедри обчислювальної техніки

Русанова Ольга Веніамінівна, доцент, к.т.н., доцент кафедри обчислювальної техніки

Рецензент — Тарасенко-Клятченко О.В., доцент, к.т.н., доцент кафедри системного програмування і спеціалізованих комп'ютерних систем

Відповідальний редактор — Кулаков Ю.О., професор, д.т.н., професор кафедри обчислювальної техніки

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського
(протокол № 1 від 02.09.2022 р.)
за поданням Вченої ради факультету інформатики та обчислювальної техніки
(протокол № 11 від 11.07.2022 р.)

В учбовому посібнику розглянуті вимоги до виконання лабораторного практикуму, які пов'язані з вивченням принципів паралелізму, методам і засобам організації та імплементації паралельної обробки та паралельних систем високої і надвисокої продуктивності. У лабораторних роботах вивчаються і аналізуються різні архітектури паралельних комп'ютерних систем. Описані теоретичні основи для виконання робіт. Для кожної роботи викладені мета, вихідні дані, завдання та аналіз отриманих результатів. Зазначені правила проведення лабораторних робіт та вимоги до звіту. Посібник призначений для здобувачів ступеня бакалавра за спеціальністю 123 Комп'ютерна інженерія, а також буде корисним при виконанні дипломних проєктів бакалавра, присвячених розробці та використанню сучасних високопродуктивних паралельних комп'ютерних систем.

Реєстр. № НП 21/22-488. Обсяг 2 авт. арк.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
проспект Перемоги, 37, м. Київ, 03056
https://kpi.ua
Свідоцтво про внесення до Державного реєстру видавців, виготовлювачів і розповсюджувачів видавничої продукції ДК № 5354 від 25.05.2017 р.

ЗМІСТ

ВСТУП……4
ПРАВИЛА ПРОВЕДЕННЯ ЛАБОРАТОРНИХ РОБІТ ТА ВИМОГИ ДО ЗВІТУ……...5
Лабораторна робота № 1……...7
Лабораторна робота № 2……15
Лабораторна робота № 3……20
Лабораторні роботи № 4……25
Лабораторна робота № 5……30
Лабораторна робота № 6……36
Лабораторна робота № 7……39
СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ……47

ВСТУП

Основним показником рівня розвитку обчислювальної техніки є показник продуктивності комп'ютерних систем. В цьому плані традиційний послідовний характер обробки інформації практично вичерпав себе. В даний час питання подальшого істотного підвищення продуктивності комп'ютерних систем вирішуються на основі принципів паралельної обробки інформації. Тому дисципліна «Комп’ютерні системи» присвячена питанням вивчення принципів паралелізму, методам і засобам організації та імплементації паралельної обробки та паралельних систем високої і надвисокої продуктивності. Метою дисципліни є вивчення методів і засобів побудови ефективних паралельних і розподілених обчислювальних систем широкого і спеціального призначення і питань організації паралельних обчислювальних процесів. У даному посібнику розглядаються інструкції до виконання лабораторних робіт, де вивчаються і аналізуються різні архітектури паралельних комп’ютерних систем.

ПРАВИЛА ПРОВЕДЕННЯ ЛАБОРАТОРНИХ РОБІТ ТА ВИМОГИ ДО ЗВІТУ

1. До виконання лабораторних робіт допускаються тільки ті студенти, які пройшли інструктаж з техніки безпеки і неухильно його виконують.
2. Під час проведення лабораторних занять мобільні телефони повинні бути налаштовані на беззвучний режим або вимкнуті.
3. Лабораторні роботи виконуються студентами індивідуально, згідно з завданням, погодженим з викладачем.
4. Необхідними умовами допуску до поточної лабораторної роботи є захист попередньої роботи та підготовка до майбутньої роботи (знання теоретичних відомостей, порядку виконання).
5. Лабораторні роботи супроводжуються складанням звіту і виконуються в лабораторії кафедри обчислювальної техніки. Звіт повинен включати:
    - Титульний лист, на якому має бути вказана тема лабораторної роботи, назва дисципліни, факультет, група, прізвище, ім’я, по батькові студента;
    - Мета роботи;
    - Підготовка вхідних даних (розпаралелений алгоритм згідно з варіантом) у вигляді графу задачі;
    - Результати роботи моделі заданої архітектури паралельної системи (час виконання, коефіцієнт прискорення, ефективність системи) при різній кількості процесорів у вигляді таблиці;

- Аналіз роботи і вибір оптимальної кількості процесорів для виконання заданої задачі;
- Висновки;
- Скріншоти роботи системи.

Лабораторна робота №1.

МОДЕЛЮВАННЯ ЧАСОВИХ ХАРАКТЕРИСТИК КОМПЬЮТЕРНИХ СИСТЕМ ТА МЕРЕЖ

Мета роботи. Розробити та реалізувати у вигляді програми математичну модель розрахунку часових характеристик мультипроцесорної системи або мережі.

Загальні положення

Трудомісткість алгоритмів - це кількість обчислювальної роботи, яка потрібна для реалізації алгоритму на комп'ютерній системі. Трудомісткість оцінюється алгоритму кількістю операцій, що виконуються з метою обробки, вводу та виводу інформації у процесі розв'язання задачі.

Інформація, що стосується задачі, поділяється на програму та дані, які складаються з вихідних даних, проміжних величин та кінцевих результатів. Дані можуть бути розташовані як в оперативній, так і в зовнішній пам'яті.

Оперативна та зовнішня пам'ять відрізняються методами доступу, швидкісними характеристиками. Витрати часу на звернення до оперативної пам'яті пропорційні кількості інформації, що зчитується або записується.

Витрати часу на звернення до зовнішньої пам'яті переважно визначаються часом пошуку області даних і меншою мірою залежать від кількості інформації, що передається. Тому для зменшення витрат часу на обмін інформацією з зовнішньою пам'яттю за кожне звернення варто передавати достатньо велику кількість інформації.

Інформація, яка зберігається у зовнішній пам'яті, поділяється на файли - впорядковані сукупності даних, що обробляються єдиним спільним засобом. Операції звернення до зовнішньої пам'яті розглядаються, звичайно, як операції вводу-виводу.

Складність обчислень оцінюється трудомісткістю, яка, визначається кількістю операцій, що виконуються з метою обробки, вводу або виводу інформації у процесі розв'язання задачі. Трудомісткість алгоритму - величина ймовірнісна, оскільки тривалість та кількість операторів, що виконуються за різними вихідними даними, може бути різною.

Застосування статистичних методів визначення кількості операцій - складний і тривалий процес, тому трудомісткість алгоритмів звичайно описується методами математичної статистики, наприклад, математичним описом кількості виконуваних операцій.

Трудомісткість алгоритму приблизно можна охарактеризувати такою сукупністю параметрів:

Q - середня кількість процесорних операцій, які виконуються за одну реалізацію алгоритму;

$N_1, \ldots, N_H$ - середня тривалість звернення до файлів $F_1, \ldots, F_H$;

$Q_1, \ldots, Q_H$ - середня кількість байтів , які передаються за одне звернення відповідно до файлів $F_1, \ldots, F_H$.

Значення Q характеризує трудомісткість обчислень, а значення $N_1, \ldots, N_H$;$Q_1, \ldots, Q_H$ - трудомісткість процесу вводу-виводу інформації; H - кількість файлів.

Під час розв'язання задач аналізу та синтезу комп'ютерних систем виникає необхідність опису властивостей обчислювальних процесів. Під час дослідження варто подавати обчислювальний процес у вигляді моделей, які містять інформацію тільки з властивостей самого процесу, врахування яких необхідне та можливе під час розв'язання задач аналізу та синтезу комп'ютерних систем. З урахуванням даних з трудомісткості алгоритмів можливе формулювання таких вимог до моделей.

Модель повинна визначати:

1. послідовність проходження алгоритмів запитів, які зберігаються у кожному файлі;
2. трудомісткість обслуговування запитів – кількість операцій, які повинен виконувати процесор під час обслуговування запиту на обчислення, та кількість символів інформації, що вводиться та виводиться;
3. відображати обчислювальні процеси як реалізацію випадкового процесу.

Обчислювальні процеси, що виникають під час роботи моделі, повинні відповідати реальним процесам з точністю до співпадіння, принаймні, математичних очікувань їх одноіменних характеристик.

Перші дві вимоги визнають коло даних з обчислювальних процесів, які найсуттєвіше впливають на порядок функціонування комп'ютерної системи.

Необхідність третьої вимоги продиктована результативністю ймовірнісного підходу до дослідження обчислювальних процесів. Фактор випадковості, що вводиться в модель, дозволяє створювати нескінченну кількість реалізацій обчислювального процесу, що характерно для процесів виконання алгоритмів.

Остання вимога визначає мінімальну норму «точності» моделі.

З урахуванням цих вимог будується модель обчислювального процесу. На початковому етапі необхідно оцінити трудомісткість алгоритмів. Стосовно задач аналізу трудомісткості операторів поділятють на функціональні, переходу та операторів звернення до файлів.

Функціональні оператори задають сукупність обчислювальних операцій. Оператори переходу задають правила вибору одного з можливих шляхів розвитку обчислювального процесу. Оператори звернення до файлів відображають процес обміну інформацією з зовнішніми пристроями.

Здебільшого, у системі існує декілька зовнішніх пристроїв та відповідно формується кілька файлів. Функціональних операторів та операторів переходу називають основними операторами. Для оцінювання трудомісткості алгоритмів необхідно розбити множину операторів на класи:

основних операторів

$$S_0 = \{V_{a_i}, \dots, V_{a_m}\}$$

операторів звернення до файлів

$$S_H = \{V_i^h, \dots, V_i^h\}$$

де $h = 1, \dots, H$ - номер файлу, до якого здійснюється звернення в

$V_{\beta i}$ операторі; n - загальна кількість файлів; m - загальна кількість операторів цього типу.

Сукупність операторів та зв'язків між ними найяскравіше подано у вигляді графу алгоритму, у якому кожна дуга (i, j) графу відмічена ймовірністю переходу P_{ij} з вершини V_i до вершини V_j. Оскільки обчислювальний процес не може зупинитись у будь-якій, не враховуючи кінцевої (k), вершині, то з ймовірністю 1 відбудеться перехід до будь-якої вершини графу алгоритму. З урахуванням цього для будь-якої вершини $V_i(i=1,2, \dots, k-1)$ повинна виконуватись умова

$$\sum_{i=1}^{k} P_{ij} = 1.$$

Ймовірності P_{ij} залежать від ймовірностей виконання умови, що перевіряється оператором i з метою вибору шляху переходу. Наприклад, припустимо оператор i створює перехід до оператора j при від'ємному значенні деякої змінної x та до оператора 1 при додатньому значенні цієї змінної. Якщо відомо, що величина x рівномірно поділена в

діапазоні *(-1;+3)*, то з ймовірністю 0,25 її знак від'ємний та з ймовірністю 0,75 її знак - додатній.Отже, перехід до оператора *j* відбувається з ймовірністю P_{ij} = 0,25 і перехід до оператора 1 - з ймовірністю P_{ij} = 0,75.

Для операторів циклу ймовірність переходу визначається кількістю циклів. Наприклад, оператор циклу повинен виконуватись дев'ять разів, у цьому випадку ймовірність виходу з циклу P_{ij} = 0,1, а ймовірність виконання циклу P_{ij} = 0,9. Якщо за оператором *i* неодмінно виконується оператор *j*, то P_{ij} = 1.

Приклад. Маємо схему алгоритму, зображену на Рис. 1.1. Матриця ймовірностей переходів для цього алгоритму знаходиться в табл. 1.1.

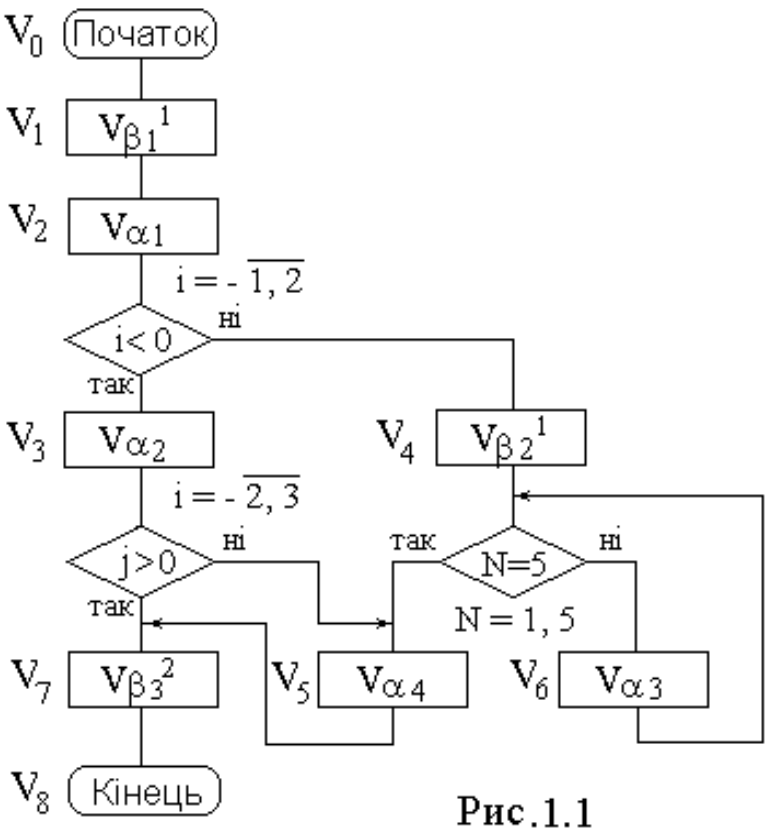


Рис.1.1

Табл. 1.1

	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8
V_1		1						
V_2			0.25	0.75				
V_3					0.5		0.5	
V_4					0.2	0.8		
V_5							1	
V_6					0.2	0.8		
V_7								1

Припустимо, що $n_1, \dots, n_{k-1}$ - середня кількість звернень до операторів $V_1, \dots, V_{k-1}$ за одну реалізацію алгоритму; k_i - кількість операцій, що складають 1-й оператор. У такому випадку характеристики трудомісткості будуть обчислені таким чином:

середня кількість процесорних операцій, що виконуються за одну реалізацію алгоритму

$$Q \le \sum_{i}^{k-1} n_i \cdot k_i \Big| , V_i \in S_0 \Big| \qquad (1)$$

середня кількість звернень до файлу F_h

$$N_h = \sum_i n_i \Big| , \quad V_i^h \in S_H \Big| \qquad (2)$$

середня кількість інформації, що передається під час одного звернення до файлу F_h

$$Q_h = \frac{1}{N_h} \sum_i n_i \cdot l_i \Big| \qquad (h = 1, \dots, H) \quad (3)$$

де l_i – середня кількість інформації, що передається під час виконанні операції i.

У виразі (1) додавання виконується для всіх верхівок, що належать до основних операторів, у виразах (2), (3) – для всіх верхівок, що відображають звернення до цього файлу.

Здебільшого, ймовірності переходів P_{kj} постійні для заданого алгоритму і перехід до наступного оператора визначається тільки розподілом ймовірності P_{kj}, тобто процес обчислення є марківським процесом з k станами $S_1, \ldots, S_k$, що відповідають перебуванню процесу у верхівках $V_1, \ldots, V_k$ графу алгоритму. Стани $S_1, \ldots, S_{k-1}$ – безповоротні (процес після деякої кількості кроків неодмінно залишає їх), стан **S** – поглинальний (досягнувши його, процес припиняється). Для спрощення позначок припустимо, що початковий стан S_1. Послідовність зміни станів визначається графом алгортму, дуги якого позначені ймовірностями переходів P_{ij}. Позначений граф алгоритму можна розглядати як граф марківського ланцюга. Середня кількість $n_1, \ldots, n_{k-1}$ перебувань марківського процесу у безповоротних станах $S_1, \ldots, S_{k-1}$ визначається коренями системи лінійних алгебраїчних рівнянь:

$$n_i = \delta_{1i} + \sum_{j=1}^{k-1} P_{ij} \cdot n_j \qquad (i = 1, \ldots, k-1) \qquad 4)$$

де δ_{1i} – символ Кронекера ($\delta_{ij} = 1$ при $i = j$ та $\delta_{ij} = 0$ при $i \neq j$).

Значення $n_1, \ldots, n_{k-1}$ визначають розв'язання системи лінійних алгебраїчних рівнянь (4), канонічний запис яких має вигляд:

$$\begin{array}{c} (P_{11} - 1) \cdot n_1 + P_{21} \cdot n_2 + \ldots + P_{(k-1)1} \cdot n_{k-1} = -1; \\ P_{12} \cdot n_1 + (P_{22} - 1) \cdot n_2 + \ldots + P_{(k-1)2} \cdot n_{k-1} = 0; \\ \cdot \end{array}$$

. . . . . .

$$P_{1(k-1)} \cdot n_1 + P_{2(k-1)} \cdot n_2 + \ldots + (P_{(k-1)(k-1)} - 1) \cdot n_{k-1} = 0. \qquad (5)$$

Під час формування коефіцієнтів системи (5) необхідно звернути увагу на те, що коефіцієнти P_{ij} записуються не за рядками, а за стовпчиками, тобто,

наприклад, на місці другого елементу першого рядка стоїть елемент з індексом 21, а не коефіцієнт з індексом 12 матриці переходів.

Визначивши за формулами (1), (2) та (3) величини Q, N_h, Q_h, можна обчислити середню трудомісткість етапу розрахунку, на підставі якої оцінюється оптимальна швидкодія процесора.

Середня трудомісткість етапу розрахунку визначається за формулою:

$$Q_0 = \frac{Q}{N} \qquad (6)$$

де N - сума середньої кількості n_i звернень до основних операторів (S_0), тобто

$$N = \sum_i n_i, \; V_i \in S_0 \qquad (7)$$

Завдання та вихідні дані до лабораторної роботи № 1

Вихідні дані.

1) граф-схема алгоритму (за погодженням з викладачем);
2) k_i – кількість операцій, що складають $V\alpha_i$ оператор (за погодженням з викладачем);
3) l_i – середня кількість інформації, що передається під час виконання $V\beta_i^m$ оператора звернення до файлу, де m – номер файлу, до якого здійснюється звернення.

Завдання на лабораторну роботу.

1) За заданою граф-схемою алгоритму побудувати матрицю ймовірностей переходів;
2) Визначити основні параметри трудомісткості алгоритму:
   a. середню кількість операцій, що виконуються за одне прокручування алгоритму;
   b. середню кількість звернень до кожного з файлів;

c. середню кількість інформації, що передається під час одного звернення до файлу;

3) Середню трудомісткість етапу обчислень.

Лабораторна робота №2.

ВИВЧЕННЯ РОБОТИ БАГАТОПРОЦЕСОРНИХ КС(СМП) ЗІ СПІЛЬНОЮ ПАМ'ЯТТЮ.

Мета роботи. Аналіз функціонування та ефективності багатопроцесорних КС зі спільною пам'яттю.

Загальні положення

Мультипроцесорні системи реалізують крупно і середньозернистий паралелізм. Кожне завдання може бути розділене на множину незалежних підзадач. Реальні додатки мають послідовно-паралельний характер. Саме тому, деякі підзадачі можуть виконуватися паралельно, а деякі - послідовно.

Мультипроцесорні системи мають розподілене управління. Сьогодні ці системи із загальною (що розподіляється) пам'яттю називають SMP (симетричні мультипроцесорні) системами.

Спершу розглянемо характеристики систем SMP. Передача даних в системах SMP здійснюється за пам'ятю, що забезпечує швидкий обмін даними. Основними недоліками SMP систем є конфлікти пам'яті і обмежена масштабованість. Розглянемо основні типи структур SMP.

Перша структура SMP системи показана на рис.1

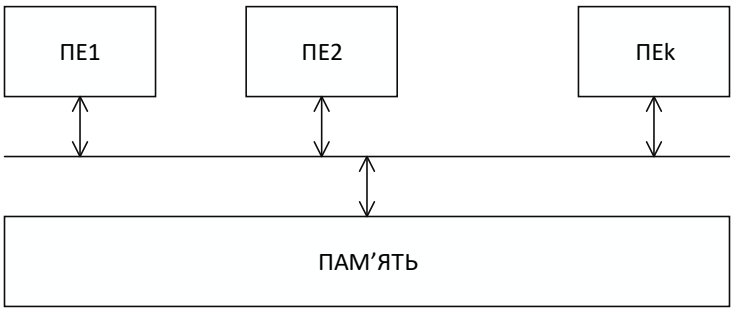


Рис.1. Структура системи із загальною пам’ятю

В цьому випадку всі РЕ мають загальну пам'ять і доступ до загальної пам'яті. Таким чином тільки один РЕ може звертатися до пам'яті за одиницю часу. Перевагою такої структури є швидка передача даних. Недоліком є тимчасова затримка через конфлікти РЕ, пов'язані з доступом до загальної пам'яті. Масштабування системи (збільшення кількості ПЕ) призводить до збільшення кількості запитів і відповідно більшою затримкою. Використання кеш-пам'яті в РЕ може зменшити цей недолік.

Друга структура системи SMP показана на рис.2

Кеш кожного РЕ може використовуватися для зберігання своїх власних програм і проміжних даних. Таким чином, загальна пам'ять використовується тільки для даних, які використовуються декількома ПЕ. Переваги цієї структури полягають в швидкому доступі до кеш-пам'яті, паралельному доступі РЕ до їх власним кешу і зменшенні кількості доступів до загальної пам'яті. Але в цьому випадку так само до пам'яті може звертатися тільки один РЕ. Тому, якщо у нас є багато спільних даних для декількох РЕ, кількість конфліктів пам'яті може залишатися великою. Зменшення можливо за використання загальної пам'яті, яка складається з банків адресної пам'яті.

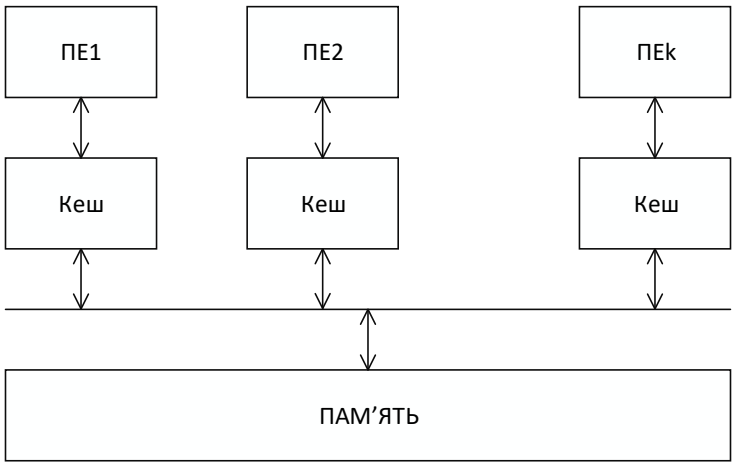


Рис.2. Структура SMP-системи із загальною пам’ятю і локальною кеш-пам’ятю

Структура цієї SMP-системи показана на рис.3

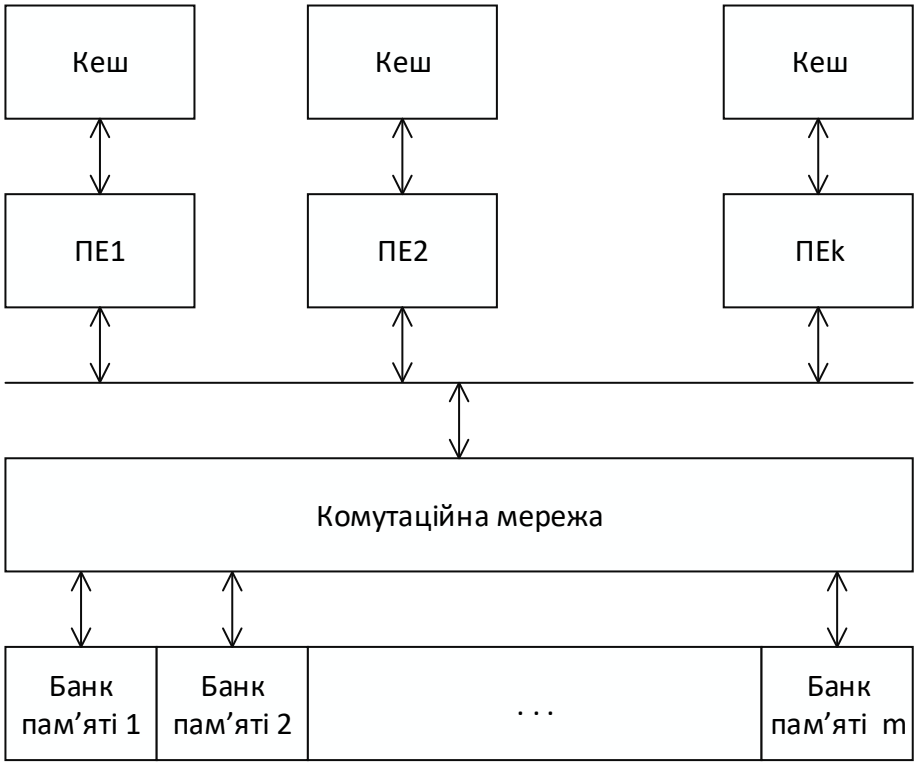


Рис.3. Структура SMP-системи зі спільною пам’ятю, що складається із

локально адресованих ділянок пам’яті

В цьому випадку m PE одночасно може звертатися до різних банків даних. Ми маємо конфлікти тільки в тому випадку, коли різним процесорам необхідний доступ до єдиного банку пам'яті. Взаємозалежне мережеве використання для підключення PE з різними банками пам'яті. Це структура сучасної SMP-системи, яка забезпечує мінімальне число конфліктів пам'яті.

Завдання на лабораторну роботу

Вихідні дані:

1) Задача (табл.1), розмірність матриці 4 або 5;coef. α and β values (tabl.1)

Завдання на лабораторну роботу.

1. Створіть граф паралельної обробки для додатка, де вузли - підзадачі, а ребра - з'єднання між залежними подзадачами;
2. Визначити значення Te (в тактах) для кожної підзадачі додатки, якщо t + = 1 такт; T * = * t +, t / = * t +;
3. Виконувати підзадачі оптимального призначення (розкладу) на процесорах систем SMP (рис. 3) з різною кількістю процесорів (від 2 до ширини графіка). Використовуйте програму моделювання Eucaliptus;
4. Визначення часу виконання, прискорення і ефективності додатків для різної кількості систем SMP;
5. Порівняйте показники ефективності систем SMP з «ручним» і «автоматичним» розкладом в Eucaliptus.
6. Виберіть оптимальне число процесорів для SMP для використання програми і результати аналізу роботи системи SMP.

Табл.1

№ вариант	ОБЧИСЛЮВАЛЬНИЙ АЛГОРИТМ	Coef.α	Coef.β
1	Множення двох матриць	2	3
2	Рішення системи лінійних рівнянь за допомогою формули Крамера	3	4
3	Рішення системи лінійних рівнянь методом Гаусса	2	4
4	LU-розкладання матриці	3	5
5	Рішення системи лінійних рівнянь методом простої ітерації (Якобі)	2	5
6	Рішення системи лінійних рівнянь методом Гаусса-Зейделя	2	4
7	Рішення системи лінійних рівнянь методом квадратних коренів	4	5
8	Рішення системи лінійних рівнянь за схемою Халецького	3	5
9	Рішення системи лінійних рівнянь методом Ньютона	2	3
10	Пошук оберненої матриці	2	4

Лабораторна робота №3.

ВИВЧЕННЯ РОБОТИ ПАРАЛЕЛЬНИХ КС ІЗ ЗАГАЛЬНИМ УПРАВЛІННЯМ І ЗАГАЛЬНОЮ (ЩО РОЗДІЛЯЄТЬСЯ) ПАМ'ЯТТЮ

Мета роботи. Аналіз функціонування і ефективності паралельних багатопроцесорних КС із загальною (що розділяється) пам'яттю.

Загальні положення

Паралельні комп'ютерні системи із загальним управлінням (ПСЗУ) орієнтовані на дрібнозернистий паралелізм (арифметичний рівень). Структура ПСЗУ показана на рис.1

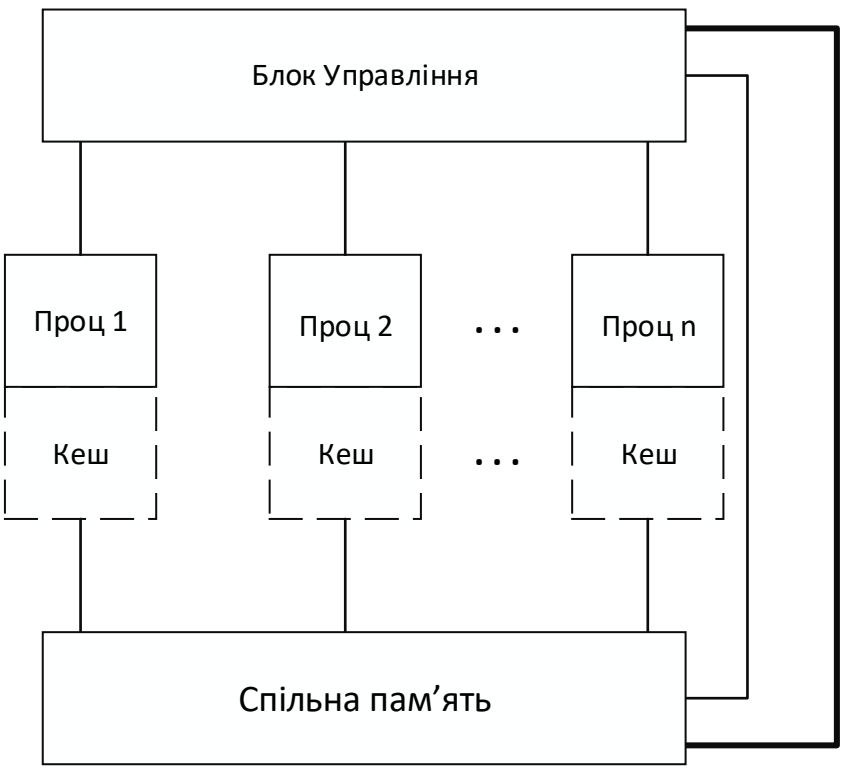


Рис.1. Структура ПСЗУ

Як бачимо, ПСЗУ складається з n процесорів, які управляються єдиним блоком управління.

Це системи з пам'яттю, що розділяється. Процесори працюють з інструкціями, які зчитуються із загальної пам'яті, але кожен з процесорів може мати свою власну локальну пам'ять (кеш) для зберігання проміжних даних.

Існує два варіанти роботи таких систем.

Перший варіант полягає в наступному. Блок управління (БУ) оглядає усі інструкції з пам'яті зі швидкістю, що набагато перевищує швидкість виконання однієї команди в процесорі. Після цього БУ аналізує можливість виконання команди. Якщо наявний вільний процесор, то команда буде виконана. Потім БУ аналізує наступну команду і так далі. Перегляд інструкцій і їх виконання можна продовжити, навіть якщо у нас немає вільних процесорів. У цих випадках ці готові інструкції можуть бути записані тимчасовими в буфер БУ. Аналіз програм та організацію паралельних обчислень виконує БУ під час виконання з використанням методу інтерпретації. Таким чином, комп’ютерні системи, що керуютьяс потоком даних (Dataflow системи) відносяться до цього варіанту ПСЗУ. Розпаралелювання в системах потоків даних виконується спочатку під час компіляції, а потім динамічно під час вирішення програми. Виконання команди можливо, коли дані готові. Це дані визначають порядок виконання команд. Ці системи відносяться до асинхронного типа. Перевага систем потоків даних полягає в програмній багатоплановості через проблеми з розпаралелюванням.

Другий варіант ПСЗУ пов'язаний з системою VLIW (very long instruction word). В цьому випадку БУ зчитує із загальної пам'яті вектор команд, де число елементів дорівнює числу встановлених процесорів (для кожного

процесора своя інструкція). Компілятор перетворює традиційну програму в набір VLIW, який вирішується в системі послідовно. Ця система відноситься до системи з динамічною реконфігурацією часу комп'ютера. Це синхронна система. Тому кількість тактів визначається найскладнішою інструкцією, яка включає даний VLIW. Можливе використання неефективних процесорів через різних комплексів інструкцій і їх рішення по часу. Щоб усунути цей недолік, можна використовувати RISC-архітектуру для процесорів, де більшість команд мають однаковий час виконання.

Обидва варіанти ПСЗУ забезпечують паралельне виконання різних інструкцій, незважаючи на один БУ. Більш того, вони мають обмежену масштабованість через загальну (загальною) пам'яті. Але перший і другий варіанти ПСЗУ мають різні вимоги до пам'яті. Для перших ми потребуємо високошвидкісному доступі до пам'яті. Час доступу до пам'яті має бути менше у n раз в порівнянні з часом виконання на процесорі. Для другого варіанту ПСЗУ нам потрібна багаторозрядні пам'ять (сто і навіть тисяча розрядів), тому що доступ до одиночної пам'яті повинен забезпечувати читання вектора команди.

Ми розглянемо такі показники ефективності:

1. Прискорення (S). Визначається за наступною формулою: $S = \frac{T_0}{T_P}$ де T_0 - час виконання завдання на одному процесорі, а Tp - час вирішення додатки на паралельній системі. Значення S змінюватиметься у такому діапазоні: $n \geq S > 0$. Якщо S <1, то наявне знищення швидкості обчислення. Якщо S = 1, то прискорення не відбувається. Якщо S> 1, то наявне прискорення.
2. Ефективність застосування (Ea). Він визначає як відношення прискорення до числа процесорів.В ідеальній паралельній системі

прискорення дорівнює n, а ефективність застосування дорівнює 1.

Завдання і вихідні дані до лабораторної роботи №3

Вихідні дані.

1) Арифметичний стиснення (табл.1);
2) Коеф. α і β (табл.1)

Завдання на лабораторну роботу.

1) Створити паралельний граф (дерево) для арифметичного виразу, в якому вузли відображають арифметичні операції, а ребра - зв'язки між залежними операціями;
2) Визначити значення τ_0 (в тактах) для кожної арифметичної операції, якщо t + = 1 такт; t*= α * t+, t/= β *t+;;
3) Виконувати операції оптимального призначення (розкладу) на процесорах системи dataflow з різною кількістю процесорів (від 2 до ширини графу). Використовуйте програму моделювання eucalyptus;
4) Виконувати операції оптимального призначення (розкладу) на процесорах системи vliw із різною кількістю процесорів (від 2 до ширини графу). Використовуйте програму моделювання eucalyptus;
5) Визначати час виконання, прискорення і ефективність додатків для різних номерів потоків даних і vliw;
6) Виберіть оптимальні процесори числа потоків даних і vliw-систем для використання арифметичного виразу і виконайте аналіз результатів роботи систем
7) Порівняйте показники ефективності потоків даних і VLIW-систем.

Табл.1

№ вар	Арифметичний вираз	Число процесорів	Коеф.α	Коеф.β
1	*(A+B+C)*(D+G)/E+L*K/F*	3	2	3
2	*A*B+C*D+G*K/(L+H)*E*	2	3	4
3	*(A+B/C*G)*(K+E+L)/R+D*	3	2	4
4	*A*B/C+D*E/(G+K/L)+M*	2	3	5
5	*A+B+C/D+G*(K/L+M+N)*	4	2	5
6	*A+B*(C+D*E*(G+L/K))+N*	2	2	4
7	*A*(B+C/D)+G*K*L+M/N*	3	4	5
8	*A/B+(C+D*E)*(G+K/L*M)*	2	3	5
9	*(A*B+C/D+G*K)(M+N+E)*	4	2	3
10	*A/B+C/D+G*(K+L*(M+N))*	4	2	4

Лабораторна робота №4.

ВИВЧЕННЯ РОБОТИ МАТРИЧНИХ КОМП'ЮТЕРНИХ СИСТЕМ

Мета роботи. Проаналізувати функціональність та ефективність матричних КС.

Загальні положення

Матричні системи класифікують як ОКМД систему.

Розглянемо загальні типи структур матричних систем. Перша з них складається з n процесорних елементів (ПЕ), мережі міжмережевого з'єднання і програмної (інструкції) пам'яті. Процесорний елемент містить процесор і локальну пам'ять даних. На сьогоднішній день інша стуктура вважається більш оптимальною(рис.2). Різниця другої структури в наявності скалярного процесора і локальної пам'яті. Цей процесор використовують для скалярних операцій.

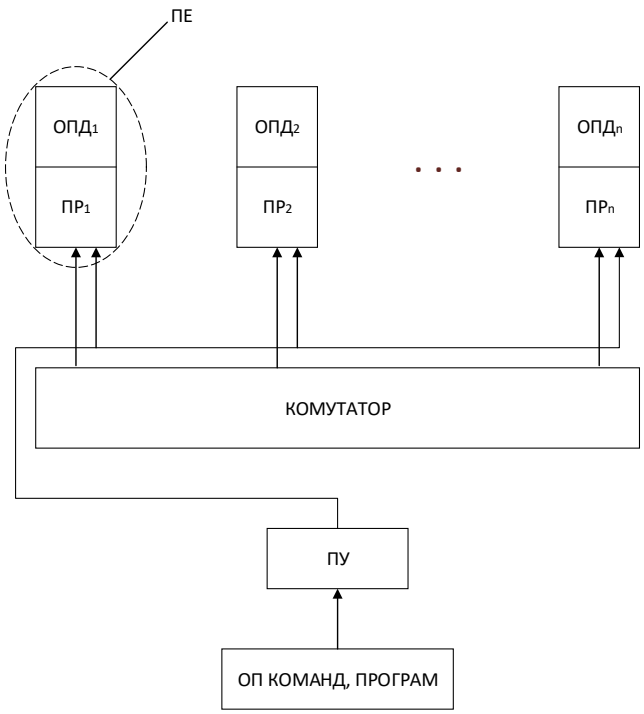


Рис.1. Загальна структура матричної КС без скалярного процесору

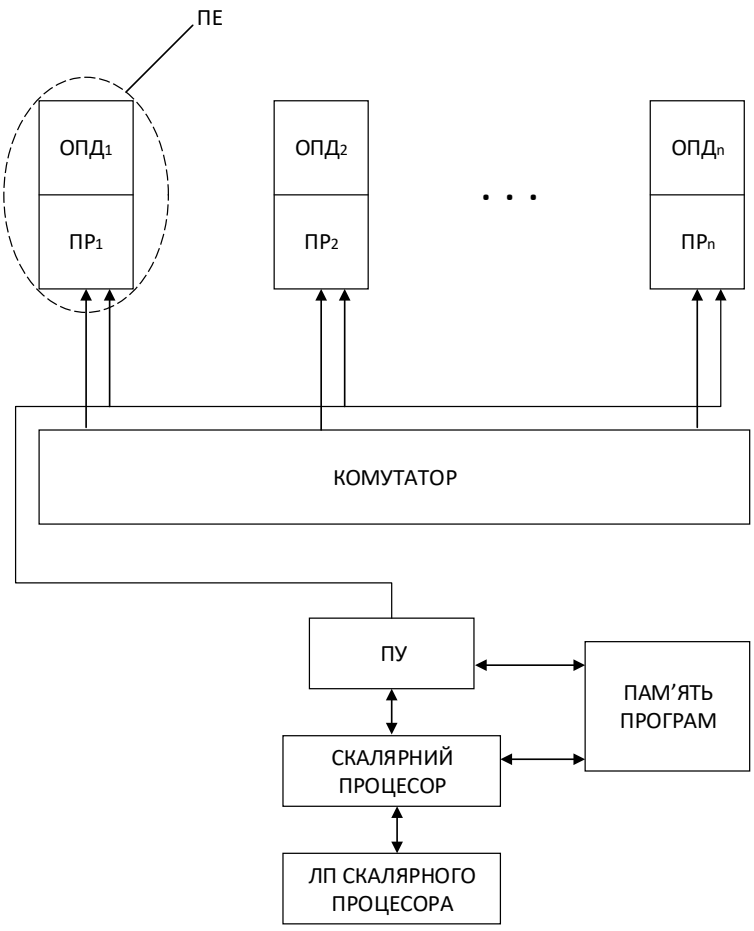


Рис.2. Загальна структура матричної КС зі скалярним процесором

Основною перевагою матричних систем є простота структури. Матричні КС мають найкраще співвідношення продуктивності та вартості для задач з природним паралелізмом, таких як матричні і векторні операції, рішення систем лінійних та диференційних рівнянь і т. д.

Недоліки матричних систем наступні:

1. Невідповідність логічного і фізичного векторів.
2. Завдання може мати як векторні, так і скалярні операції, що надалі приведе до простою ПЕ, які не були залучені до вирішення.
3. Мережеві зв'язки можуть бути статичними і динамічними. Використання статичних зв'язків супроводжується певними проблемами ефективної організації передачі даних. Кращим підходом

організації динамічних зв'язків є використання комутаторів. Вони дозволяють організовувати найвищу швидкість процесорної комунікації, але дуже дорогі. Їх вартість пропорційна кількості процесорів.

4. Деградація продуктивності як наслідок розгалуження алгоритму. В результаті оператора умови вибирається одна гілка рішення. Тобто процесорні елементи іншої гілки простоюють. Тому з ростом розмірності (n) системи продуктивність має нелінійну залежність і дорівнює log2n (рис.3).
5. Можливість виконання одного типу команд тягне за собою проблеми при підготовці завдання для вирішення на матричних КС. Приклад такої підготовки представлений на рис.4.

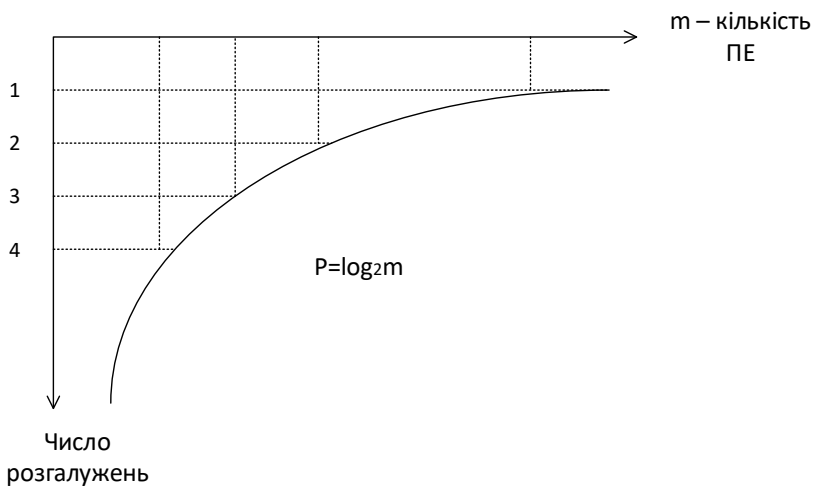


Рис.3 Залежність ефективності та кількості розгалужень матричної системи

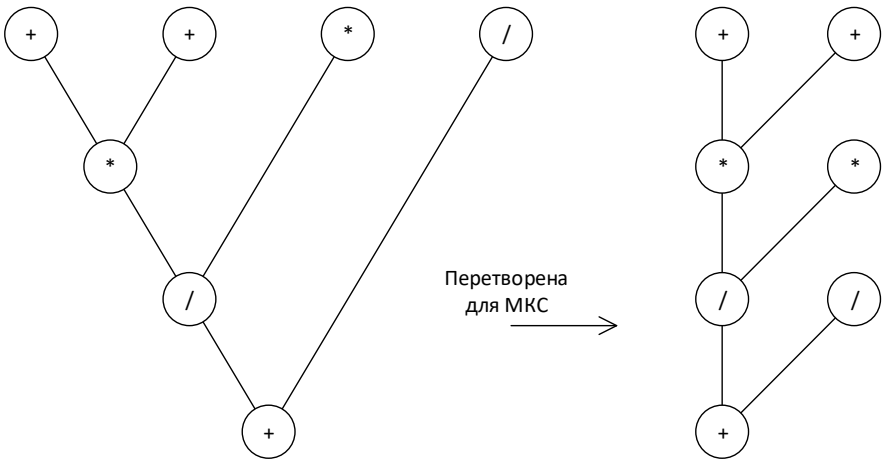


Рис.4 Підготовка дерева паралельних операцій для вирішення на матричній системі

Будемо розглядати наступні показники:

1. Прискорення (S). Обчислюється за такою формулою: де T0 час вирішення на одному процесорі і Tp час вирішення на паралельній системі. Значення S змінюється в проміжку: If S <1, тоді швидкість знижується. Якщо S = 1 тоді прискорення немає. Якщо S> 1, тоді прискорення наявне.
2. Ефективність (Ea). Визначається відношенням прискорення до кількості процесорів.

Для ідеальної паралельної системи прискорення дорівнює n і ефективність дорівнює 1.

Завдання та вихідні дані до лабораторної роботи

Вихідні дані:

1) Арифметичний вираз (табл.1);
2) коефіцієнт. α і β (табл.1)

Завдання на лабораторну роботу.

1) За заданим алгоритмом побудувати граф, в якому вершинам відповідає обчислення гілок алгоритму, а дугам - зв'язки (інформаційні або за пам'ятю) між гілками алгоритму;
2) Підготувати граф до матричної Кс, де кожному рівню відповідає окрема операція
3) Визначити значення Те (в тактах) для кожної операції, враховуючи, що t + = 1 такт; $t_* = \alpha \blacksquare * t_+$, $t_/ = \beta \blacksquare * t_+$;
4) Виконати оптимальне планування послідовності операцій для матричної КС при різній кількості процесорів (від двох до ширини графа). Використовувати для цього програму Eucalyptus;
5) Знайти час виконання, прискорення, ефективність для матричних КС при різних параметрах.
6) Вибрати оптимальну кількість процесорів для вирішення даного арифметичного виразу. Підготувати результати аналізу роботи системи.

Tabl.1

№ Вар.	Арифметичний вираз	Коеф. α	Коеф. β
1	*(A+B+C)*(D+G)/E+L*K/F*	2	3
2	*A*B+C*D+G*K/(L+H)*E*	3	4
3	*(A+B/C*G)*(K+E+L)/R+D*	2	4
4	*A*B/C+D*E/(G+K/L)+M*	3	5
5	*A+B+C/D+G*(K/L+M+N)*	2	5
6	*A+B*(C+D*E*(G+L/K))+N*	2	4
7	*A*(B+C/D)+G*K*L+M/N*	4	5
8	*A/B+(C+D*E)*(G+K/L*M)*	3	5
9	*(A*B+C/D+G*K)(M+N+E)*	2	3
10	*A/B+C/D+G*(K+L*(M+N))*	2	4

Лабораторна робота №5.

ВИВЧЕННЯ РОБОТИ МАКРОКОНВЕЄРНИХ КС

Мета роботи. Аналіз функціонування та ефективності конвеєрних КС.

Загальні положення

Конвеєр являє собою процесор, розподілений на P частин (шарів або рядків), які виконують послідовно етапи кожної обчислювальної задачі (операції). У той час, коли j-й шар процесора виконує j-й етап деякої k-ї задачі, (j-1)-й шар може виконувати (j-1)-й етап (k+1)-ї задачі , (j-2)-й шар може виконувати (j-2)-й етап (k+2)-ї задачі і т.д. , тобто 1-й шар процесора може в цей же час виконувати 1-й етап (i+j-1)-ї задачі (де j=2, ... ,P).Таким чином, кожна задача чи операція виконується за P етапів під час проходження P шарів конвеєрного процесора.

У КС з конвеєрним процесором може виконуватись одночасно декілька (P) операцій з перетворенням даних, що відповідає визначенню обчислювальної системи. Але ці операції у будь-який момент часу обов’язково знаходяться на різних етапах виконання і, в принципі, не можуть розпочинатись усі одночасно.

Конвеєрні системи з розподіленим керуванням, орієнтовані на паралелізм незалежних гілок, називають макроконвеєрами (крупно та середньозернистого паралелізму). На рис.1 показано конвеєрну систему з розподіленим керуванням.

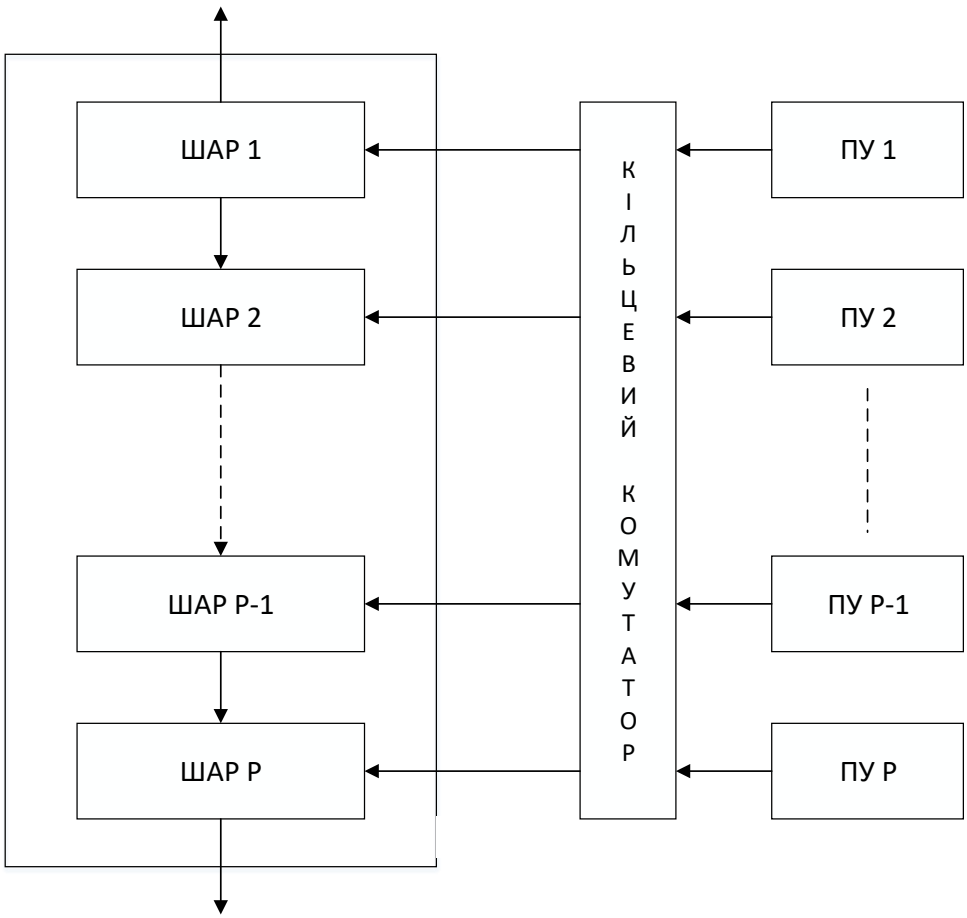


Рис.1. Загальна структура макроконвеєрної КС

У ній є *Р* пристроїв управління (ПУ) - за кількістю шарів, на які розділено конвеєрний процесор Принципово важливим є належність *Р* окремих вузлів формування адреси наступної інструкції та регістрів інструкцій; можливо, що у кожного ПУ є власні індекси та базові регістри, регістри ключів захисту та інші індивідуальні ланцюги. Частина ланцюгів може бути спільною для різних ПУ (наприклад, формувач адреси операндів, дешифратор коду операцій), які використовуються всіма ПУ за чергою. Ці спільні ланцюги на рисунку входять до складу шарів процесора.

Пристрої керування приєднано до шарів конвеєрного процесора через кільцевий комутатор . Комутатор обладнано так , що в 1-му такті роботи КС

до 1-го шару процесора під'єднано 1-й ПУ , який починає виконання своєї задачі (операції) : у 2-му такті 1-й ПУ перемикається на 2-й шар процесору , а до 1-го шару під'єднується 2-й ПУ і т.д. У *P*-му такті 1-й ПУ під'єднаний до *P*-го шару процесора , де закінчується виконання 1-ї задачі (операції) , до (*P*-1)-го шару під'єднано 2-й ПУ , ... , до 1-го шару під'єднано *P*-й ПУ. Після закінчення 1-ї задачі (операції) 1-го ПУ в (*P*+1)-му такті 1-й пристрій знову під'єднується до 1-ї сходинки процесора, де починає виконання 2-ї задачі (операції) і т.д.

Система, побудована зазначеним чином, аналогічна за своїми можливостями *P*-процесорній системі типу мультипросеорна система і орієнтована на використання паралелізму незалежних гілок.

Розглянутий конвеєрний процесор є багатофункціональним , тобто дозволяє виконувати одночасно різні операції. Однак при цьому тривалість такту роботи шарів процесора залежить від операцій, що виконуються, і визначатиметься тривалістю виконання найдовшої операції.

Завдання та вихідні дані до лабораторної роботи № 5

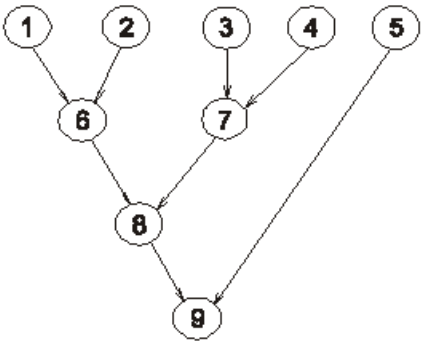
Вихідні дані.

1. Граф задачі для обчислення, вершинам якого відповідають обічлювальні гілки алгоритму (рис. 2). Варіант задачі (див. табл.1 лаб.роб.№2). Розмірність матриці узгоджується з викладачем. Студенти іноземних груп мають можливість також скористуватись варіантом, вказаним у таблиці.
2. Кількість шарів конвеєрного процесора (2 та 4) ;

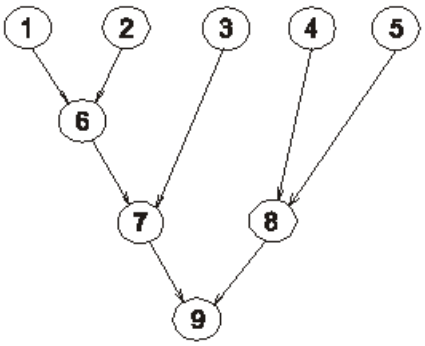
Завдання на лабораторну роботу

1. Побудвати граф алгоритму обчислювальної задачі;
2. Виконати задачу оптимального потактового завантаження (задачу планування обчислень) для макроконвеєрної КС;

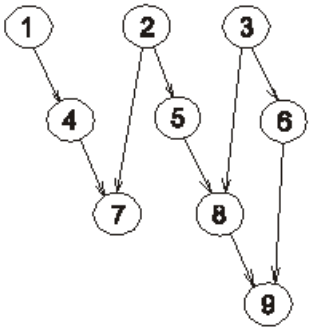
3. Визначити час виконання, прискорення та ефективність для різної кількості шарів макроконвеєру;
4. Визначити оптимальну кількість шарів макроконвеєрної КС для задачі, що розглядається. Надати аналіз результатів роботи системи.



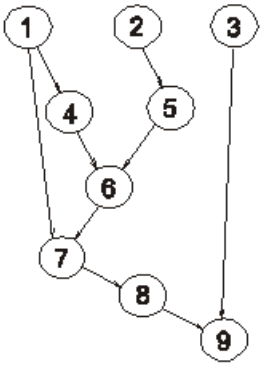
Варіант 1



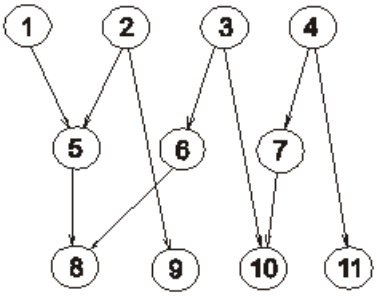
Варіант 2



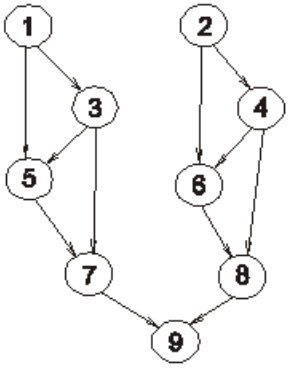
Варіант 3



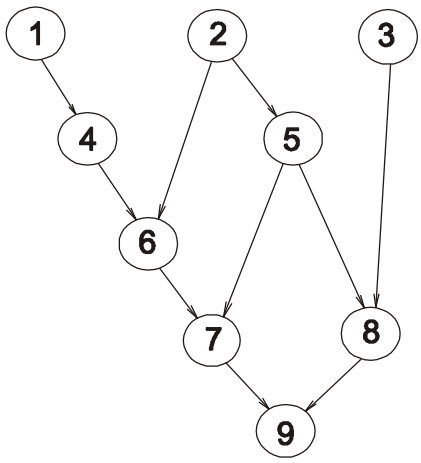
Варіант 4



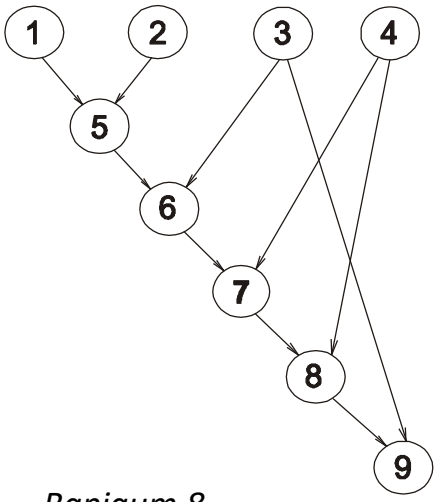
Варіант 5



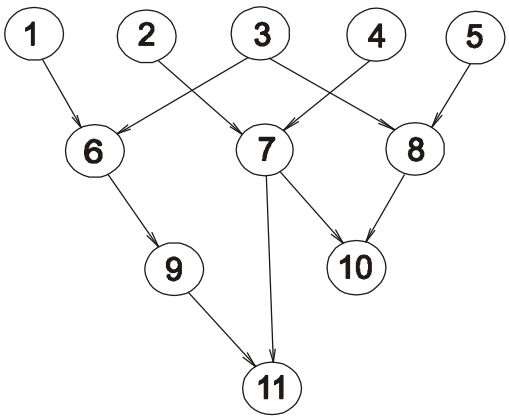
Варіант 6



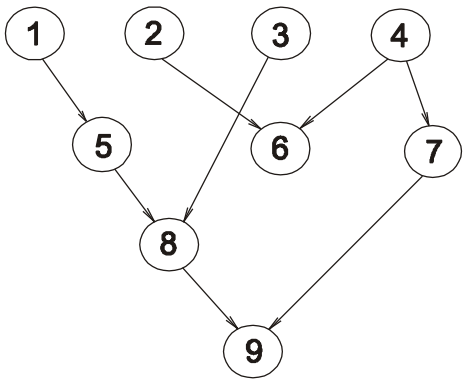
Варіант 7



Варіант 8



Варіант 9



Варіант 10

Рис.2 Варіанти паралельного графу

Лабораторна робота №6.

ВИВЧЕННЯ РОБОТИ КОНВЕЄРНИХ КС

Мета роботи. Аналіз функціонування та ефективності конвеєрних КС.

Загальні положення

Конвеєрні системи зі спільним керуванням, орієнтовані на використання паралелізму суміжних операцій (дрібнозернистий паралелізм). Пристрій керування (рис.1) тут один, але регістрів для зберігання вказівок (РгВ) стільки ж, скільки є шарів у процесорі.

1-а інструкція програми зчитується в РгВ(1), і в 1-му шарі процесора починається її виконання . У наступному такті 1-а інструкція передається в РгВ(2) , її виконання продовжує 2-й шар процесора , а в РгВ(1) зчитується 2-а інструкція програми і в 1-му шарі конвеєрного процесора починається її виконання і т.д.

Розглянутий конвеєрний процесор є багатофункціональним. Відрізнятимемо динамічну та статичну перебудову (переналаштування?) процесора для виконання різних операцій. Таким чином, переналаштування часу виконується динамічно. Такі конвеєри мають найбільшу потенційну продуктивність, але є найдорожчими.

Динамічна перебудова конвеєра дозволяє виконувати різні операції одночасно в різних шарах конвеєра. Такт конвеєра в кожний момент часу визначається тривалістю виконання найдовшої операції (у випадку використання конвеєра зі змінним тактом) чи тривалістю виконання найдовшої з усіх операцій, на виконання яких орієнтовано цей багатофункціональний конвеєр (у випадку використання конвеєра з постійним тактом).

У випадку статичної перебудови конвеєра на виконання нової операції необхідно дочекатись звільнення конвеєра від попередньої операції і тільки після цього завантажувати наступну операцію (якщо вона відмінна від виконуваної) .

Незважаючи на багатофункціональність, цей конвеєр за визначений час працює як монофункційний пристрій, у якому всі шари виконують певний єдиний тип інтсрукції. Переналаштування від одного типу інструкції до іншого можливе лише за звільнення конвеєру.

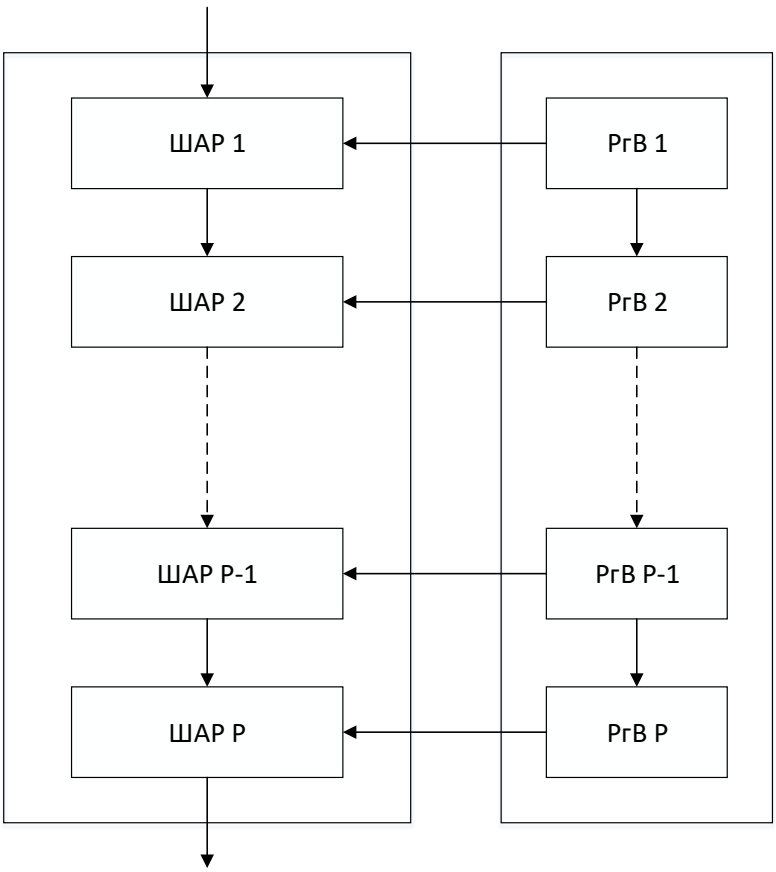


Рис.1 Загальна структура конвеєрних КС

Завдання та вихідні дані до лабораторної роботи № 6

Вихідні дані

1. Арифметичний вираз (табл.1);
2. Коеф. α и β (табл.1);
3. Кількість шарів (2 та 4).

Завдання на лабораторну роботу.

1. Побудувати граф для заданого арифметичного виразу, де всі вузли – операції, ребра – зв'язки між залежними операціями. Вагу вузла вважати рівною коеф.α та β для кількості слоїв рівній 4. Для кількості слоїв рівної 2 збільшити значення коефіцієнтів удвічі.
2. Виконати задачу потактового завантаження (задачу для планування обчислень) для конвеєра с динамічною реконфігурацією за різної кількості шарів (2 та 4).
3. Виконати задачу потактового завантаження (задачу для планування обчислень) для конвеєра с динамічною реконфігурацією за різної кількості шарів (2 та 4).
4. Визначити оптимальну кількість шарів для конвеєра за кожного типу реконфігурації. Надати аналіз результатів роботи систем;
5. Порівняти показники ефективності конвеєрів різного типу реконфігурації.

Табл.1

Вар.	Арифм. вираз	Коеф. α	Коеф. β
1	*(A+B+C)*(D+G)/E+L*K/F*	2	3
2	*A*B+C*D+G*K/(L+H)*E*	3	4
3	*(A+B/C*G)*(K+E+L)/R+D*	2	4
4	*A*B/C+D*E/(G+K/L)+M*	3	5
5	*A+B+C/D+G*(K/L+M+N)*	2	5
6	*A+B*(C+D*E*(G+L/K))+N*	2	4
7	*A*(B+C/D)+G*K*L+M/N*	4	5

8	*A/B+(C+D*E)*(G+K/L*M)*	3	5
9	*(A*B+C/D+G*K)(M+N+E)*	2	3
10	*A/B+C/D+G*(K+L*(M+N))*	2	4

Лабораторна робота №7.

ВИВЧЕННЯ РОБОТИ БАГАТОПРОЦЕСОРНИХ КС ІЗ РОЗПОДІЛЕНОЮ (ЛОКАЛЬНОЮ) ПАМ'ЯТТЮ

Мета роботи . Аналіз функціонування та ефективності багатопроцесорних КС з розподіленою пам'яттю.

Загальні положення

Альтернативу КС з розподіленою пам'яттю складають системи з локальною пам′яттю: у них кожен процесор може адресуватись тільки до власної пам'яті.

Обміни між процесорами відбуваються передачою повідомлень, які містять чисельну чи іншу інформацію. Використання в якості базових процесорів для цих КС звичайних мікропроцесорів викликає достатньо серйозні труднощі, у багатьох випадках пов′язані з відсутністю в них будь-якої апаратної підтримки паралельних обчислень і з їх орієнтацією на шинну архітектуру. Фірма Inmos розробила родину RISC мікропроцесорів, отримавших назву «трансп′ютери», пристосованих спеціально для паралельних КС . Трансп′ютери мають достатньо ефективну апаратну підтримку для організації паралельних обчислень. Обмін даними трансп′ ютери здійснюють послідовними двокрапковими каналами. На відміну від шинної архітектури, пропускна здібність мультитрансп'ютерної КС зростає з кількістю трансп'ютерів у ній, і ця система не має принципових обмежень на кількість паралельно працюючих процесорів.

Трансп′ютер містить в собі центральний процесор (ЦП), оперативний запам’ятовуючий пристрій (ОЗП) для зберігання даних та програм, паралельний інтерфейс зовнішньої пам’яті і чотири послідовних двунапямлених канали передачі даних для прямих зв’язків з іншими трансп’ютерами. Чіп Т800 містить, крім того, арифметичний пристрій (АП) для операцій над числами з плаваючою крапкою. Дуже важливим є той факт , що канали трансп’ютера мають автономні контролери вводу-виводу. Це дає можливість передавати дані одночасно всіма каналами і паралельно з роботою ЦП (і АП для Т800), який (які) може в цей час звертатись як до внутрішньої,так і до зовнішньої пам’яті.

Структуру трансп’ютера Т800 зображено на рис.1.

Трасп′ютери орієнтовано на виконання програм , написаних мовою Оккам.

У них апаратно реалізовані всі аспекти моделі цієї мови, тому спеціальні програмні засоби для планування паралельних процесів та організації обміну між ними не потрібні.

Трансп′ютери забезпечують можливість виконання процесів з двома рівнями приорітету : високим - 0 та низьким - 1 . Паралельні процеси можуть розподілятись між різними процесорами в мультитрансп’ютерній КС або виконуватись на одному трансп’ютері. Архітектура трансп’ютера передбачає ефективні засоби для підтримки обох видів паралельної обробки. Для одночасного оброблення великої кількості паралельних процесів на одному трансп’ютері існує спеціальний планувальник , який виконує розподіл часу між ними (виділення кванта часу). У будь-який момент часу паралельні процеси розподіляються на два класи – активні процеси (виконуються або готові до виконання) та неактивні процеси (очікують ввід, вивід або певного моменту часу). Існують дві черги активних процесів (з

високим і низьким приорітетами). Процеси з низьким приорітетом можуть виконуватись тільки тоді, коли черга процесів з високим приорітетом пуста.

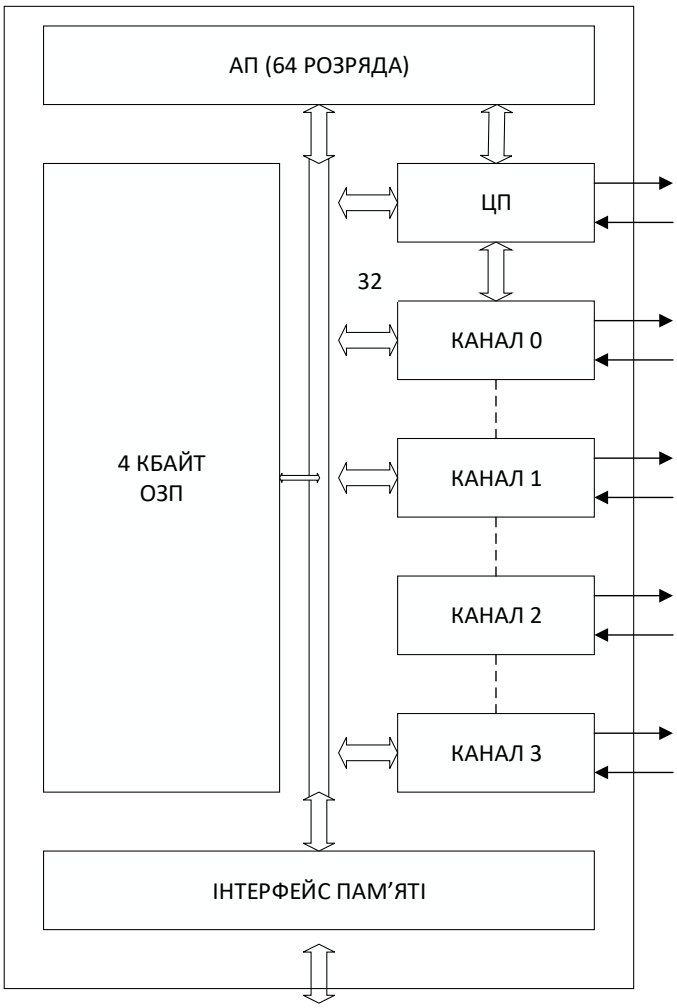


Рис.1

Комунікації між паралельними процесами здійснюються через канали. Канал між двома процесами, які виконуються на одному трансп'ютері (внутрішній канал), задається одним словом у пам'яті, а обмін здійснюється за допомогою пересилань між робочими областями цих процесів у пам'яті трансп'ютера. Канал між процесами , які виконуються на різних трансп'ютерах , виконаний апаратно у вигляді двокрапкової лінії зв'язку між цими трансп'ютерами. У моделі мови Оккам комунікації

відбуваються тільки тоді , коли і приймальний, і передавальний процеси готові до обміну даними. Тому процес, який першим досягнув готовності, повинен очікувати (у черзі очікувальних процесів), коли і другий процес також буде готовим до цього.

Ця лабораторна робота складається з двох частин. Перша частина пов'зана з дослідженням моделі обміну даними між двома трансп'ютерами. Друга частина пов'язана з дослідженням моделі мультитрансп′ютерної системи.

Модель обміну даними між трансп'ютерами може бути досліджена на прикладі функціонування системи з двох трансп'ютерів, з'єднаних між собою одним каналом, (рис.2). Каналу присвоєна назва *A*. Всередині трансп'ютера процеси також обмінюються через канали. Ці канали ,на відміну від апаратно виконаного каналу *A*, логічні і створюються за необхідністю. Вони можуть мати будь-які назви, визначені користувачем (будь-які літери латинської абетки) . Необхідно враховувати, що кількість звернень до каналу для передачі даних повинно відповідати кількості звернень до цього каналу для прийняття даних.

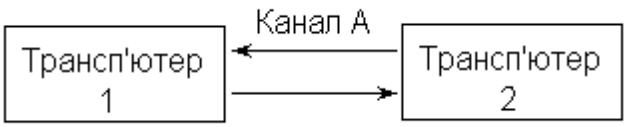


Рис.2

У моделі потактово відображається стан ЦП, черг активних і очікувальних процесів (з низьким та високим приорітетами) трансп′ютерів, а також стан каналу *A*. У моделі передбачено, що в ЦП надходять процеси із черг активних процесів з урахуванням їх приоритетів (спочатку із черги з високим приоритетом, а коли вона порожня, то із черги з низьким приоритетом).

Якщо процесу необхідно передати або прийняти дані через канал, то він переводиться в чергу очікувальних процесів відповідного приоритету. Після вступу процесу в чергу очікувальних процесів виконується перевірка, чи відчинено необхідний канал. Якщо канал відчинено , то процес зчитує або передає дані в канал. Якщо канал з необхідною назвою відсутній, то виконується відкриття каналу, процес розміщує там адресу, за якою будуть прийматись або передаватись дані. Закінчивши обмін, обидва процеси, які приймали участь у обміні даними, заносяться із черги очікувальних у кінець черги активних процесів з відповідним приоритетом. Канал, через який вони обмінювались, зачиняється.

У розглянутій моделі кожнен процес повинен мати шість етапів обробки, які чергуються з п’ятьма етапами обміну. Кожнен етап обробки в ЦП характеризується тривалістю, яка задається в тактах. Етапи обміну через канали характеризуються тривалістю обміну (в тактах) : назвою каналу, через який відбувається обмін; напрямком обміну (1- приймання даних, 0- передавання даних). У цій моделі на кожному трансп’ютері може одночасно оброблятись до п’яти таких процесів. Кількість одночасно відкритих внутрішніх каналів обміну - п’ять.

У другій частині лабораторної роботи досліджується модель мультитрансп′ютерної КС. На рис.3 зображено структуру розглянутої КС, де $Тр_1$ – 1-й трансп’ютер КС .В даній КС сусідні трансп’ютери з’єднані послідовними трансп’ютерними каналами. Крім того , всі трансп’ютери з’єднані «швидким зв’язком» через комутаційну мережу. Розглянута програмна модель дозволяє досліджувати функціонування та ефективність роботи мультитрансп’ютерної КС.

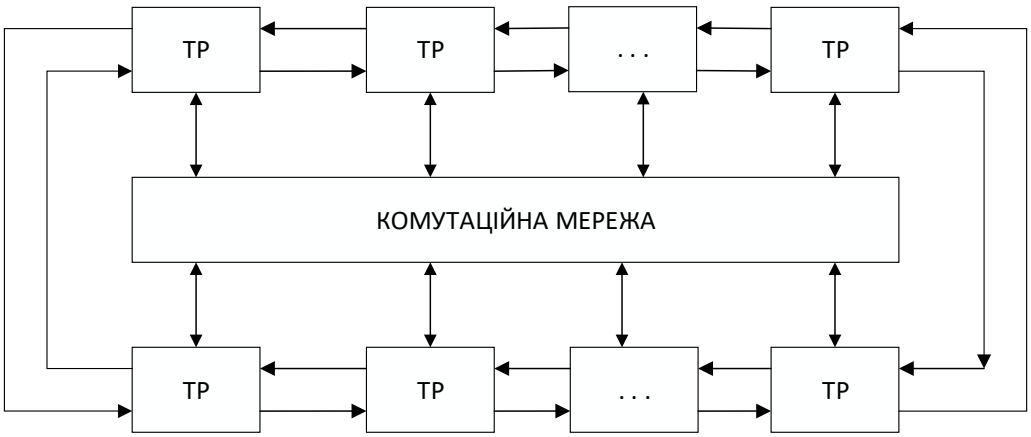


Рис.3 Система для лабораторної роботи

Завдання та вихідні дані до лабораторної роботи № 7

1-ша частина виконання лабораторної роботи

Вихідні дані 1-ї частини роботи:

1. Кількість процесів, що обробляються в 1-му трансп'ютері (К1) (табл.3.1);
2. Кількість процесів, що обробляються в 2-му тпрансп'ютері (К2) (табл.3.1);
3. Характеристика кожного процесу:
   - тривалість кожного етапу обробки (в тактах);
   - тривалість етапів обміну (в тактах);
   - напрямок обміну (прийняття-In, передавання-Out);
   - назва каналу.

Ці дані студенти формують самостійно, погоджуючи з викладачем (приклад завдання характеристики процесу наведено в табл.3.2).

Завдання:

1. Дослідити роботу програмної моделі;
2. Визначити середню тривалість знаходження кожного процесу в КС;

3. Визначити загальню тривалість роботи системи;
4. Визначити оптимальний порядок надходження процесів у трансп' ютерах з метою мінімізації загального часу роботи системи.

1-ша частина виконання лабораторної роботи

Вихідні дані 2-ї частини роботи

2. Дані, що характеризують КС :
   - Тривалість встановлення зв'язку через канали трансп'ютера (B1) (у тактах) (табл.1);
   - Тривалість встановлення зв'язку через «швидкий зв'язок» (B2) (табл.1);
   - Швидкість обміну через канали транспютера (C1) (од.інф. /такт) (табл.1);
   - Швидкість обміну через канали «швидкого зв'язку» (C2) (табл.1);
   - Тривалість кванта роботи трансп'ютера (у тактах) (D) (табл.1);
3. Граф обчислювального алгоритму:
   - Кількість гілок алгоритму, що відповідають кількості процесів для трансп'ютерної КС;
   - Тривалість виконання кожного процесу на трансп'ютері (в тактах);
   - Кількість точок зв'язку (обміну) кожного процесу з іншими процесами:
   - Для кожного зв'язку задається:
     - Номер такта, на якому відбувається обмін;
     - Об'єм інформації, що передається/приймається;
     - Номери процесів, з якими обмінюється інформацією цей процес

Дані, що характеризують граф обчислювального алгоритму , студенти формують самостійно , погоджуючи їх з викладачем.

Завдання:

1. виконати задачу оптимального потактового розподілу процесів за трансп'ютерами мультитрансп'ютерної КС ; критерієм оптимізації є мінімальний тривалість виконання заданого алгоритму на КС ;
2. дослідити роботу програмної моделі і проаналізувати отримані статистичні дані ;
3. визначити типи алгоритмів , під час виконання яких може бути досягнена максимальна ефективність роботи розглянутої мульти-трансп'ютерної КС

Табл.1

№ варіанту	**K1**	**K2**	**B1**	**B2**	**C1**	**C2**	**D**
1	3	3	40	200	1	10	20
2	2	2	20	100	2	20	10
3	3	2	15	75	2	20	6
4	2	2	8	40	5	50	4
5	3	2	40	200	1	10	20
6	2	4	40	100	2	10	6
7	2	3	15	75	3	30	10
8	2	3	8	40	5	60	3

Табл.2

Етапи	Тривалість	In -1 Out-0	Ім'я Каналу
1 оброб. 1 обмін	10 15	1	A
2 оброб. 2 обмін	15 2	0	F
3 оброб. 3 обмін	20 3	1	D
4 оброб. 4 обмін	30 5	1	K
5 оброб. 5 обмін	25 7	0	C
6 оброб.	5		

СПИСОК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. Програмне забезпечення комп'ютерних систем. Програмування та компіляція /Русанова О.В., Корочкін О.В. – Київ : КПІ ім. Ігоря Сікорського, 2020. – 94 с. Електронний ресурс .Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 6 від 31.01.2020 р.) за поданням Вченої ради ФІОТ (протокол № 4 від 25.11.2019 р.) https://ela.kpi.ua/handle/123456789/48296
2. Паралельні та розподілені обчислення. Вибрані розділи: Навч. посібник для здобувачів ступеня бакалавр за спеціальністю 123 «Комп'ютерні системи та мережі» / Корочкін О.В., Русанова О.В. – Київ : КПІ ім. Ігоря Сікорського, 2020. – 123с Електронний ресурс. Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 6 від 31.01.2020 р.) за поданням Вченої ради ФІОТ (протокол № 4 від 25.11.2019 р.) http://ela.kpi.ua/handle /123456789/48224
3. Томас Бройнль. Паралельне програмування: Початковий курс.-К.:Вища школа, 1997.-358 с.
4. Жуков І.А., Корочкін О.В. Паралельні та розподілені обчислення. Навч. посібник. Друге видання. – К.: Корнійчук, 2014. – 284 с. //comsys.kpi.ua
5. Loutsky G., Zhukov I., Korochkin A. Parallel Computing. – Kyiv, Kornechuk, 2007. -216 pp. //comsys.kpi.ua 5. Жуков

І.А., Корочкін О.В. Паралельні та розподілені обчислення. Навч. посібник. – К.: Корнійчук, 2005. – 226 с. //comsys.kpi.ua

6. Korochkin O. Multicore programming in Ada. Навч. посібник з гріфом НТУУ “КПІ” [Електроний ресурс]., Київ, НТУУ-КПІ, 2018.- 114 с. //comsys.kpi.ua

7. Hesham El-Rewini, Ted G.Lewis. Distributed and Parallel Computing-Manning
   Publications Co., 1997.-447 p.
8. V.Kumar, A.Grama, A.Gupta, GKarypis. Introduction to Parallel Computing.
   Design and Analysis of Algorithms- Benjamin/Cummings Pub.Co, 1995.-597 p.