Міністерство освіти і науки України
ДВНЗ "Ужгородський національний університет"
Факультет інформаційних технологій
Кафедра інформаційних управляючих систем та технологій

В. М. Коцовський

Технології розподілених систем та паралельних обчислень
Методичні матеріали до лабораторних робіт

Ужгород – 2020

ЗМІСТ

Вступ

Навчальна дисципліна "Технології розподілених систем та паралельних обчислень" вивчається студентами спеціальності 122 — "Комп'ютерні науки" у VII семестрі. Актуальність вивчення основних понять теорії паралельних обчислень спеціалістами в галузі комп'ютерних наук зумовлена наявністю значної кількості практично важливих задач великої розмірності, розв'язання яких потребує паралельного використання великої кількості розподілених обчислювальних ресурсів [1, 2]. Тому важливою складовою процесу підготовки фахівців сфери IT є вироблення знань та навичок, які стосуються розуміння та використання сучасних методів паралельного програмування [3–5].

Видання містить методичні вказівки до п'яти лабораторних робіт. У першій лабораторній розглядаються способи обробки даних в обчислювальних системах та характеристики систем функціональних пристроїв. Друга лабораторна присвячена паралельним формам графа алгоритму. У третій роботі містяться завдання з основ паралельного багатопотокового програмування у середовищах Java та .NET Framework. Темою четвертої лабораторної роботи є паралельне програмування з використанням технології OpenMP. Остання лабораторна присвячена проблемі розробки та синхронізації паралельних програм у сучасних фреймворках.

У кожному розділі наведено приклади розв'язування типових завдань лабораторних. Базовий теоретичний матеріал, достатній для успішного виконання лабораторних, наведений в [1–4]. Додаткові теоретичні відомості можуть бути знайдені в [5–11]. Визначення математичних понять та об'єктів, які необхідні для розуміння завдань лабораторних, наведено в [12–14].

Лабораторна робота № 1

Способи обробки даних в обчислювальних системах. Характеристики систем функціональних пристроїв.

Мета та завдання роботи: вироблення навичок оцінювання характеристик систем обчислювальних пристроїв, які працюють у різних режимах роботи.

Приклад 1.1. Обробка даних на конвеєрному пристрої складається із 5 стадій, тривалості яких рівні 3, 5, 2, 6 та 4 такти відповідно. Виконати наступні завдання, вважаючи, що ініціалізація конвеєра потребує 2 тактів та тривалість одного такту складає 5 нс:

1) Обчислити кількість тактів, необхідну для виконання 1000 операцій обробки даних за умови, що пристрій працює:
   а) у послідовному режимі;
   б) у конвеєрному режимі.
2) Підрахувати пікову продуктивність системи.
3) Визначити найменшу кількість операцій, при виконанні яких у конвеєрному режимі досягається прискорення не менше за 90% від граничного прискорення.

Розв'язок. Запишемо характеристики конвеєрного пристрою:

$l = 5,\ \sigma = 2,\ \tau = 5 \cdot 10^{-9}$ с, $t_1 = 3, t_2 = 5,\ t_3 = 2,\ t_4 = 6,\ t_5 = 4$. Тоді $t_{\max} = \max\{t_1, \ldots, t_5\} = 6$.

1) Знайдемо шукану кількість тактів у випадку $n = 1000$:
   а) у послідовному режимі:
   $t_s(n) = (t_1 + \ldots + t_l) \cdot n;$
   $t_s(1000) = (3 + 5 + 2 + 6 + 4) \cdot 1000 = 20000;$
   б) у конвеєрному режимі:
   $t_c(n) = (t_1 + \ldots + t_l) + (n-1)t_{\max} + \sigma;$
   $t_c(1000) = 20 + 999 \cdot 6 + 2 = 6016.$
2) Підрахуємо пікову продуктивність системи для обох режимів:

$$\pi_s = \frac{1}{(t_1 + \ldots + t_l)\tau} = \frac{1}{20 \cdot 5 \cdot 10^{-9}} = \frac{10^9}{100} = 10^7.$$

$$\pi_c = \frac{1}{t_{\max}\tau} = \frac{1}{6 \cdot 5 \cdot 10^{-9}} = \frac{10^9}{30} \approx 3.33 \cdot 10^7.$$

3) Визначимо шукану кількість операцій n, яка задовольняє умову $S(n) \geq 0{,}9S$:

$$\frac{20n}{20 + 6(n-1) + 2} \geq 0{,}9\frac{20}{6},$$

$$\frac{n}{6n + 16} \geq \frac{0.9}{6},$$

$$6n \geq (6n + 16) \cdot 0{,}9$$

$$6n \geq 5{,}4n + 14{,}4$$

$$0.6n \geq 14{,}4$$

$$n \geq \frac{144}{6} = 24$$

Відповідь: $n = 24$.

Приклад 1.2. Граф системи функціональних пристроїв (ФП) наведений на рис. 1. Відомі пікові продуктивності пристроїв системи: $\pi_1 = 10$, $\pi_2 = 5$, $\pi_3 = 8$, $\pi_4 = 6$, $\pi_5 = 7$, $\pi_6 = 9$, $\pi_7 = 12$, $\pi_8 = 8$, $\pi_9 = 10$, $\pi_{10} = 4$, $\pi_{11} = 6$, $\pi_{12} = 4$, $\pi_{13} = 6$. Знайти:

1) завантаженості усіх пристроїв системи;
2) реальну продуктивність системи;
3) завантаженість системи;
4) прискорення системи.

Розв'язок. Як видно з рис. 1, система складається з трьох незалежних підсистем. Згідно до 1-го закону Амдала реальна продуктивність кожної із підсистем визначається продуктивністю найменш продуктивного пристрою цієї підсистеми.

1) Нехай $\pi^{(i)}$ — реальні продуктивності, з якими працюють усі пристрої i-системи, $i = 1,2,3$. Тоді

$$\pi^{(1)} = \min\{\pi_1, \ldots, \pi_5\} = 5,\ \pi^{(2)} = \min\{\pi_6, \ldots, \pi_9\} = 8,\ \pi^{(3)} = \min\{\pi_{10}, \ldots, \pi_{13}\} = 4.$$

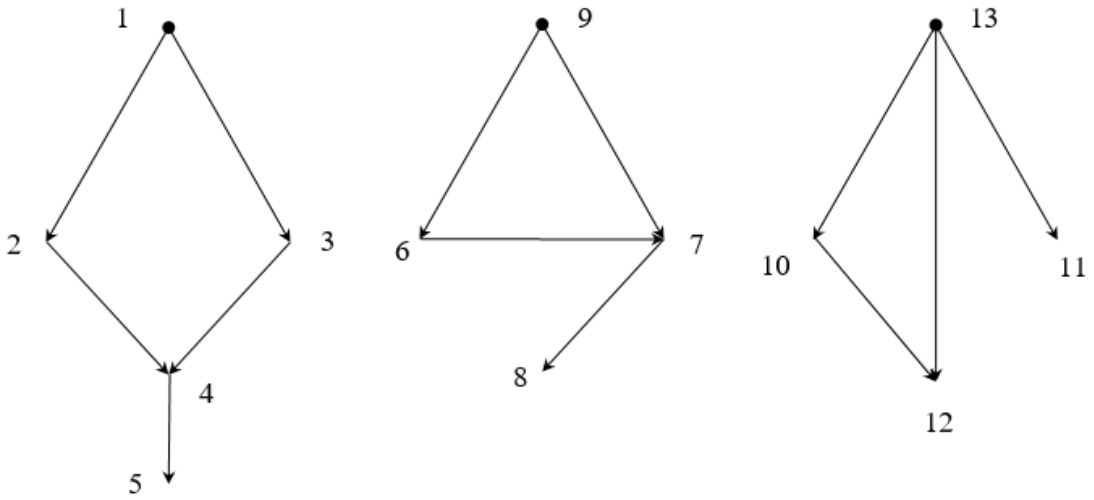

Рис. 1.

Завантаженості p_i пристроїв першої підсистеми рівні $\pi^{(1)}/\pi_i, (i=1,\ldots 5)$:

$$p_1=\frac{5}{10},\ p_2=1,\ p_3=\frac{5}{8},\ p_4=\frac{5}{6},\ p_5=\frac{5}{7}.$$

Завантаженості p_i пристроїв другої підсистеми рівні $\pi^{(2)}/\pi_i, (i=6,\ldots 9)$:

$$p_6=\frac{8}{9},\ p_7=\frac{2}{3},\ p_8=1,\ p_9=\frac{4}{5}.$$

Завантаженості p_i пристроїв третьої підсистеми рівні $\pi^{(3)}/\pi_i, (i=10,\ldots 13)$:

$$p_{10}=1,\ p_{11}=\frac{2}{3},\ p_{12}=1,\ p_{13}=\frac{2}{3}.$$

2) $r=r^{(1)}+r^{(2)}+r^{(3)}$, де $r^{(i)}$ — реальна продуктивність *i*-ї підсистеми, $i=1,2,3$. За першим законом Амдала (див. твердження 2.4 [1]) $r^{(i)}=l_i\cdot\pi^{(i)}$, де l_i — кількість пристроїв і-ї підсистеми. Тому

$$r^{(1)}=5\cdot 5=25,\ r^{(2)}=4\cdot 8=32,\ r^{(3)}=4\cdot 4=16.$$

Отже, $r=25+32+16=73$.

3) Для знаходження завантаженості системи використаємо формулу $r=p\cdot\pi$, де p — завантаженість, π — пікова продуктивність системи. Тоді

$$\pi=\pi_1+\ldots+\pi_{13}=10+5+8+6+7+9+12+8+10+4+6+4+6=95.$$

Тому $p=r/\pi=73/95\approx 0,77$.

4) Скористаємося формулою $S=r/\max\limits_{1\le i\le l}\pi_i$. Тоді $S=73/12\approx 6,08$.

Приклад 1.3. Визначити максимальне можливе прискорення і ефективність системи, яка складається з однакових пристроїв і призначена для реалізації алгоритму, граф якого наведений на рис. 2 (вершини графу відповідають операціям).

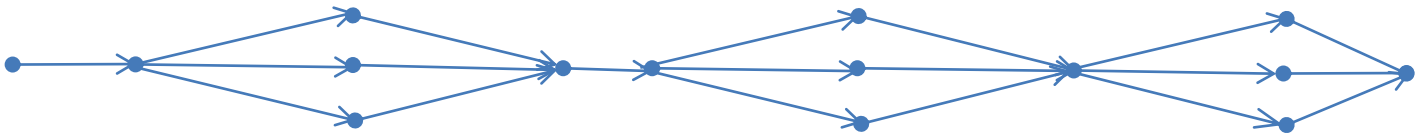

Рис. 2. Граф системи

Розв'язок. Використаємо 2-й закон Амдала. З рис. 2 можна зробити висновок, що, $N=15,\ n=6$. Звідси $\beta=6/15=2/5$. Ширина алгоритму — 3. Тому

$$S_3=\frac{3}{3\cdot 2/5+(1-2/5)}=\frac{3\cdot 5}{9}=\frac{5}{3},\quad E_3=\frac{S_3}{3}=\frac{5}{9}\approx 55{,}56\%.$$

Приклад 1.4. Нехай у алгоритмі паралельні обчислення складають 5/6. Визначити:

1) Максимальне можливе прискорення у випадку використання 5 однакових універсальних процесорів;
2) Мінімальну кількість процесорів, використання яких може забезпечити 85% граничного прискорення.

Розв'язок. Визначимо частку послідовних обчислень: $\beta=1-5/6=1/6$.

1) Скористаємося 2-м законом Амдала:

$$S_5=\frac{5}{5\cdot 1/6+1-1/6}=\frac{5\cdot 6}{10}=3.$$

2) Використаємо 3-й закон Амдала:

$$S_l\ge 0{,}8\cdot S_{\max},$$

$$\frac{l}{l/6+5/6}\ge\frac{1}{1/6},$$

$$\frac{6l}{l+5}\ge 0{,}85\cdot 6,$$

$$l\ge 0{,}85\cdot(l+5),$$

$0{,}15l \geq 4{,}25,$

$l \geq 85/3 \approx 28{,}33$

Відповідь: 29.

Варіанти завдань

Варіант № 1

1) Обробка даних на конвеєрному пристрої складається із 6 стадій, тривалості яких рівні 4, 7, 5, 2, 6 та 4 такти відповідно, ініціалізація конвеєра потребує 3 тактів та тривалість одного такту складає 2 нс.
   a) Обчислити кількість тактів, необхідну для виконання 2000 операцій обробки даних за умови, що пристрій працює:
      i) у послідовному режимі;
      ii) у конвеєрному режимі.
   b) Підрахувати пікову продуктивність системи для обох режимів функціонування.
   c) Визначити найменшу кількість операцій, при виконанні яких у конвеєрному режимі досягається прискорення не менше за 95% від граничного прискорення.
2) Граф системи ФП наведений на рис. 3. Відомі продуктивності пристроїв системи: $\pi_0 = 5, \pi_1 = 8, \pi_2 = 4, \pi_3 = 8, \pi_4 = 7, \pi_5 = 6, \pi_6 = 9, \pi_7 = 12, \pi_8 = 9, \pi_9 = 15, \pi_{10} = 4, \pi_{11} = 8, \pi_{12} = 10, \pi_{13} = 8, \pi_{14} = 11$. Визначити:
   a) завантаженості усіх пристроїв системи;
   b) завантаженість системи;
   c) реальну продуктивність системи;
   d) прискорення системи.
3) Визначити максимальне можливе прискорення і ефективність системи, яка складається з однакових пристроїв і призначена для реалізації алгоритму, граф якого наведений на рис. 5.
4) Нехай у деякому алгоритмі послідовні обчислення складають 1/4. Визначити:
   a) максимальне можливе прискорення у випадку використання $l = 6$ однакових універсальних процесорів;

b) кількість процесорів, використання яких може забезпечити від 60 до 95% від максимально можливого прискорення.

Варіант № 2

1) Обробка даних на конвеєрному пристрої складається із 7 стадій, тривалості яких рівні 3, 4, 2, 5, 2, 6 та 4 такти відповідно, ініціалізація конвеєра потребує 1 такт та тривалість одного такту складає 4 нс.
   a) Обчислити кількість тактів, необхідну для виконання 500 операцій обробки даних за умови, що пристрій працює:
      i) у послідовному режимі;
      ii) у конвеєрному режимі.
   b) Підрахувати пікову продуктивність системи для обох режимів функціонування.
   c) Визначити найбільшу кількість операцій, при виконанні яких у конвеєрному режимі досягається прискорення не більше за 80% від граничного прискорення.
2) Граф системи ФП наведений на рис. 4. Відомі продуктивності пристроїв системи: $\pi_0 = 7,\ \pi_1 = 5,\ \pi_2 = 10,\ \pi_3 = 8,\ \pi_4 = 6, \pi_5 = 6,\ \pi_6 = 9,\ \pi_7 = 12,\ \pi_8 = 8,\ \pi_9 = 5,\ \pi_{10} = 11,$ $\pi_{11} = 8,\ \pi_{12} = 7,\ \pi_{13} = \pi_{14} = 8$. Визначити:
   a) завантаженості усіх пристроїв системи;
   b) завантаженість системи;
   c) реальну продуктивність системи;
   d) прискорення системи.
3) Визначити максимальне можливе прискорення і ефективність системи, яка складається з однакових пристроїв і призначена для реалізації алгоритму, граф якого наведений на рис. 6.
4) Нехай у деякому алгоритмі паралельні обчислення складають 5/6. Визначити:
   a) максимальне можливе прискорення у випадку використання $l = 10$ однакових універсальних процесорів;
   b) найменшу кількість процесорів, використання яких може забезпечити 90% максимально можливого прискорення.

Варіант № 3

1) Обробка даних на конвеєрному пристрої складається із 5 стадій, тривалості яких рівні 4, 3, 8, 6 та 4 такти відповідно, ініціалізація конвеєра потребує 2 тактів та тривалість одного такту складає 1 нс.
   a) Обчислити кількість тактів, необхідну для виконання 1500 операцій обробки даних за умови, що пристрій працює:
      i) у послідовному режимі;
      ii) у конвеєрному режимі.
   b) Підрахувати пікову продуктивність системи для обох режимів.
   c) Визначити найменшу кількість операцій, при виконанні яких у конвеєрному режимі досягається прискорення не менше за 98% від граничного прискорення.
2) Граф системи ФП наведений на рис. 3. Відомі продуктивності пристроїв системи: $\pi_0 = 15,\ \pi_1 = 9,\ \pi_2 = 4,\ \pi_3 = 8,\ \pi_4 = 7,\ \pi_5 = 6,\ \pi_6 = 9,\ \pi_7 = 12,\ \pi_8 = 8,\ \pi_9 = 12,\ \pi_{10} = 7,\ \pi_{11} = 8,\ \pi_{12} = 10,\ \pi_{13} = 8,\ \pi_{14} = 11$. Визначити:
   a) завантаженості усіх пристроїв системи;
   b) завантаженість системи;
   c) реальну продуктивність системи;
   d) прискорення системи.
3) Визначити максимальне можливе прискорення і ефективність системи, яка складається з однакових пристроїв і призначена для реалізації алгоритму, граф якого наведений на рис. 5.
4) Нехай у деякому алгоритмі послідовні обчислення складають 30%. Визначити:
   a) максимальне можливе прискорення у випадку використання $l = 7$ однакових універсальних процесорів;
   b) найбільшу кількість процесорів, при використанні яких ефективність системи не менша за 15%.

Варіант № 4

1) Обробка даних на конвеєрному пристрої складається із 7 стадій, тривалості яких рівні 4, 2, 6, 5, 9, 6 та 3 такти відповідно. Виконати наступні завдання, вважаючи, що ініціалізація конвеєра потребує 1 такт та тривалість одного такту складає 4 нс.

a) Обчислити кількість тактів, необхідну для виконання 800 операцій обробки даних за умови, що пристрій працює:
   i) у послідовному режимі;
   ii) у конвеєрному режимі.
b) Підрахувати пікову продуктивність системи для обох режимів функціонування.
c) Визначити діапазон для числа операцій, при виконанні яких у конвеєрному режимі досягається прискорення, яке складає від 85% до 95% граничного прискорення.

2) Граф системи ФП наведений на рис. 4. Відомі продуктивності пристроїв системи: $\pi_0 = 9, \pi_1 = 6, \pi_2 = 12, \pi_3 = 7, \pi_4 = 5, \pi_5 = 8, \pi_6 = 3, \pi_7 = 10, \pi_8 = 8, \pi_9 = 7, \pi_{10} = 11, \pi_{11} = 9, \pi_{12} = 7, \pi_{13} = \pi_{14} = 8$. Визначити:
   a) завантаженості усіх пристроїв системи;
   b) завантаженість системи;
   c) реальну продуктивність системи;
   d) прискорення системи.
3) Визначити максимальне можливе прискорення і ефективність системи, яка складається з однакових пристроїв і призначена для реалізації алгоритму, граф якого наведений на рис. 6.
4) Нехай у деякому алгоритмі послідовні обчислення складають 25%. Визначити:
   a) максимальне можливе прискорення у випадку використання $l = 8$ однакових універсальних процесорів;
   b) найменшу кількість процесорів, використання яких може забезпечити 90% максимально можливого прискорення.

Варіант № 5

1) Обробка даних на конвеєрному пристрої складається із стадій, тривалості яких рівні 2, 3, 7, 1, 9 та 4 такти, ініціалізація конвеєра потребує 2 тактів та тривалість одного такту складає 2 нс.
   a) Обчислити кількість тактів, необхідну для виконання 3000 операцій обробки даних за умови, що пристрій працює:

i) у послідовному режимі;

ii) у конвеєрному режимі.

b) Підрахувати пікову продуктивність системи для обох режимів функціонування.

c) Визначити найменшу кількість операцій, при виконанні яких у конвеєрному режимі досягається прискорення не менше за 85% від граничного прискорення.

2) Граф системи ФП наведений на рис. 4. Відомі продуктивності пристроїв системи: $\pi_0 = 9, \pi_1 = 8, \pi_2 = 5, \pi_3 = 8, \pi_4 = 7, \pi_5 = 6, \pi_6 = 9, \pi_7 = 12, \pi_8 = 9, \pi_9 = 15, \pi_{10} = 7, \pi_{11} =$, $= 8, \pi_{12} = 10, \pi_{13} = 8, \pi_{14} = 11$. Визначити:

a) завантаженості усіх пристроїв системи;

b) завантаженість системи;

c) реальну продуктивність системи;

d) прискорення системи.

3) Визначити максимальне можливе прискорення і ефективність системи, яка складається з однакових пристроїв і призначена для реалізації алгоритму, граф якого наведений на рис. 5.

4) Нехай у деякому алгоритмі паралельні обчислення складають 60%. Визначити:

a) максимальне можливе прискорення у випадку використання $l = 20$ однакових універсальних процесорів;

b) кількість процесорів, використання яких може забезпечити від 70 до 85% від максимально можливого прискорення.

Варіант № 6

1) Обробка даних на конвеєрному пристрої складається із стадій, тривалості яких рівні 3, 10, 2, 7, 2, 6 та 3 такти, ініціалізація конвеєра потребує 4 тактів та тривалість одного такту — 1 нс.

a) Обчислити кількість тактів, необхідну для виконання 500 операцій обробки даних за умови, що пристрій працює:

i) у послідовному режимі;

ii) у конвеєрному режимі.

b) Підрахувати пікову продуктивність системи для обох режимів функціонування.

c) Визначити діапазон для числа операцій, при виконанні яких у конвеєрному режимі досягається прискорення, яке складає від 80% до 90% граничного прискорення.

2) Граф системи ФП наведений на рис. 3. Відомі продуктивності пристроїв системи: $\pi_0 = 4, \pi_1 = 3, \pi_2 = 10, \pi_3 = 8, \pi_4 = 2, \pi_5 = 4, \pi_6 = 9, \pi_7 = 3, \pi_8 = 8, \pi_9 = 9, \pi_{10} = 11, \pi_{11} = 8, \pi_{12} = 6, \pi_{13} = \pi_{14} = 8$. Визначити:
   a) завантаженості усіх пристроїв системи;
   b) завантаженість системи;
   c) реальну продуктивність системи;
   d) прискорення системи.
3) Визначити максимальне можливе прискорення і ефективність системи, яка складається з однакових пристроїв і призначена для реалізації алгоритму, граф якого наведений на рис. 6.
4) Нехай у деякому алгоритмі послідовні обчислення складають 1/5. Визначити:
   a) максимальне можливе прискорення у випадку використання $l = 9$ однакових універсальних процесорів;
   b) наскільки відсотків прискориться алгоритм, якщо кількість процесорів зросте з 5 до 30.

Варіант № 7

1) Обробка даних на конвеєрному пристрої полягає у виконанні стадій тривалості 6, 7, 5, 10, 6 та 3 такти, ініціалізація конвеєра займає 3 такти, один такту триває 1 нс.
   a) Обчислити кількість тактів, необхідну для виконання 3000 операцій обробки даних за умови, що пристрій працює:
      i) у послідовному режимі;
      ii) у конвеєрному режимі.
   b) Підрахувати пікову продуктивність системи для обох режимів функціонування.
   c) Визначити найменшу кількість операцій, при виконанні яких у конвеєрному режимі досягається прискорення не менше за 85% від граничного прискорення.

2) Граф системи ФП наведений на рис. 3. Відомі продуктивності пристроїв системи: $\pi_0 = 7, \pi_1 = 5, \pi_2 = 4, \pi_3 = 7, \pi_4 = 5, \pi_5 = 8, \pi_6 = 7, \pi_7 = 12, \pi_8 = 9, \pi_9 = 5, \pi_{10} = 14, \pi_{11} = 8, \pi_{12} = 10, \pi_{13} = 8, \pi_{14} = 11$. Визначити:
   a) завантаженості усіх пристроїв системи;
   b) завантаженість системи;
   c) реальну продуктивність системи;
   d) прискорення системи.
3) Визначити максимальне можливе прискорення і ефективність системи, яка складається з однакових пристроїв і призначена для реалізації алгоритму, граф якого наведений на рис. 5.
4) Нехай у деякому алгоритмі паралельні обчислення складають 80%. Визначити:
   a) максимальне можливе прискорення у випадку використання $l = 10$ однакових універсальних процесорів;
   b) кількість процесорів, використання яких може забезпечити від 50% до 80% від максимально можливого прискорення.

Варіант № 8

1) При обробці даних на конвеєрному пристрої виконуються стадії тривалості 2, 3, 5, 6, 2, 6, 4 та 2 такти, тривалість одного такту складає $0,5$ нс.
   a) Обчислити кількість тактів, необхідну для виконання 1500 операцій обробки даних за умови, що пристрій працює:
      i) у послідовному режимі;
      ii) у конвеєрному режимі.
   b) Підрахувати пікову продуктивність системи для обох режимів функціонування.
   c) Визначити найбільшу кількість операцій, при виконанні яких у конвеєрному режимі досягається прискорення не більше за 90% від граничного прискорення.
2) Граф системи ФП наведений на рис. 4. Відомі продуктивності пристроїв системи: $\pi_0 = 7, \pi_1 = 5, \pi_2 = 10, \pi_3 = 8, \pi_4 = 6, \pi_5 = 6, \pi_6 = 9, \pi_7 = 12, \pi_8 = 8, \pi_9 = 5, \pi_{10} = 11, \pi_{11} = 8, \pi_{12} = 7, \pi_{13} = \pi_{14} = 8$. Визначити:
   a) завантаженості усіх пристроїв системи;

b) завантаженість системи;

c) реальну продуктивність системи;

d) прискорення системи.

3) Визначити максимальне можливе прискорення і ефективність системи, яка складається з однакових пристроїв і призначена для реалізації алгоритму, граф якого наведений на рис. 6.

4) Нехай для деякого алгоритму максимальне прискорення рівне 4. Визначити:

   a) максимальне можливе прискорення у випадку використання $s = 3$ однакових універсальних процесорів;

   b) кількість процесорів, використання яких може забезпечити від 90% до 95% максимально можливого прискорення.

Варіант № 9

1) При обробці даних на конвеєрному пристрої виконуються стадії тривалості 4, 3, 8, 6 та 4 такти, ініціалізація конвеєра потребує 1 такт, тривалість одного такту складає 6 нс.

   a) Обчислити кількість тактів, необхідну для виконання 700 операцій обробки даних за умови, що пристрій працює:

      i) у послідовному режимі;

      ii) у конвеєрному режимі.

   b) Підрахувати пікову продуктивність системи для обох режимів функціонування системи.

   c) Визначити найменшу кількість операцій, при виконанні яких у конвеєрному режимі досягається прискорення не менше за 86% від граничного прискорення.

2) Граф системи ФП наведений на рис. 3. Відомі продуктивності пристроїв системи: $\pi_0 = \pi_1 = 8,\ \pi_2 = 7, \pi_3 = 5,\ \pi_4 = 9, \pi_5 = 10,\ \pi_6 = 9, \pi_7 = 12, \pi_8 = 7, \pi_9 = 14,\ \pi_{10} = 5,\ \pi_{11} = 8,$ $\pi_{12} = 12, \pi_{13} = 8, \pi_{14} = 14$. Визначити:

   a) завантаженості усіх пристроїв системи;

   b) завантаженість системи;

   c) реальну продуктивність системи;

   d) прискорення системи.

3) Визначити максимальне можливе прискорення і ефективність системи, яка складається з однакових пристроїв і призначена для реалізації алгоритму, граф якого наведений на рис. 5.
4) Нехай максимальне можливе прискорення алгоритму у випадку 10 універсальних процесорів рівне 3. Визначити:
   a) максимальне можливе прискорення у випадку використання 20 однакових універсальних процесорів;
   b) кількість процесорів, використання яких може забезпечити від 50 до 60% від максимально можливого прискорення.

Варіант № 10

1) При обробці даних на конвеєрному пристрої виконуються стадії тривалості 3, 8, 5, 4, 6 та 2 такти, ініціалізація конвеєра потребує 2 тактів та тривалість одного такту — 2 нс.
   a) Обчислити кількість тактів, необхідну для виконання 4000 операцій обробки даних за умови, що пристрій працює
      i) у послідовному режимі;
      ii) у конвеєрному режимі.
   b) Підрахувати пікову продуктивність системи для обох режимів функціонування системи.
   c) Визначити найменшу кількість операцій, при виконанні яких у конвеєрному режимі досягається продуктивність не менша за 80% від пікової продуктивності.
2) Граф системи ФП наведений на рис. 4. Відомі продуктивності пристроїв системи: $\pi_0 = 5, \pi_1 = 10, \pi_2 = 8, \pi_3 = 6, \pi_4 = 7, \pi_5 = 6, \pi_6 = 9, \pi_7 = 12, \pi_8 = 8, \pi_9 = 5, \pi_{10} = 11, \pi_{11} = 8, \pi_{12} = 7, \pi_{13} = \pi_{14} = 8$. Визначити:
   a) завантаженості усіх пристроїв системи;
   b) завантаженість системи;
   c) реальну продуктивність системи;
   d) прискорення системи.
3) Порівняти, як зміниться максимальне можливе прискорення і ефективність алгоритму, граф якого наведений на рис. 6, у випадку реалізації цього алгоритму у системі із двох процесорів.

4) Нехай у деякому алгоритмі паралельні обчислення складають 3/4. Визначити:
   a) максимальне можливе прискорення у випадку використання $l = 11$ однакових універсальних процесорів;
   b) найменшу кількість процесорів, використання яких може забезпечити 92% максимально можливого прискорення.

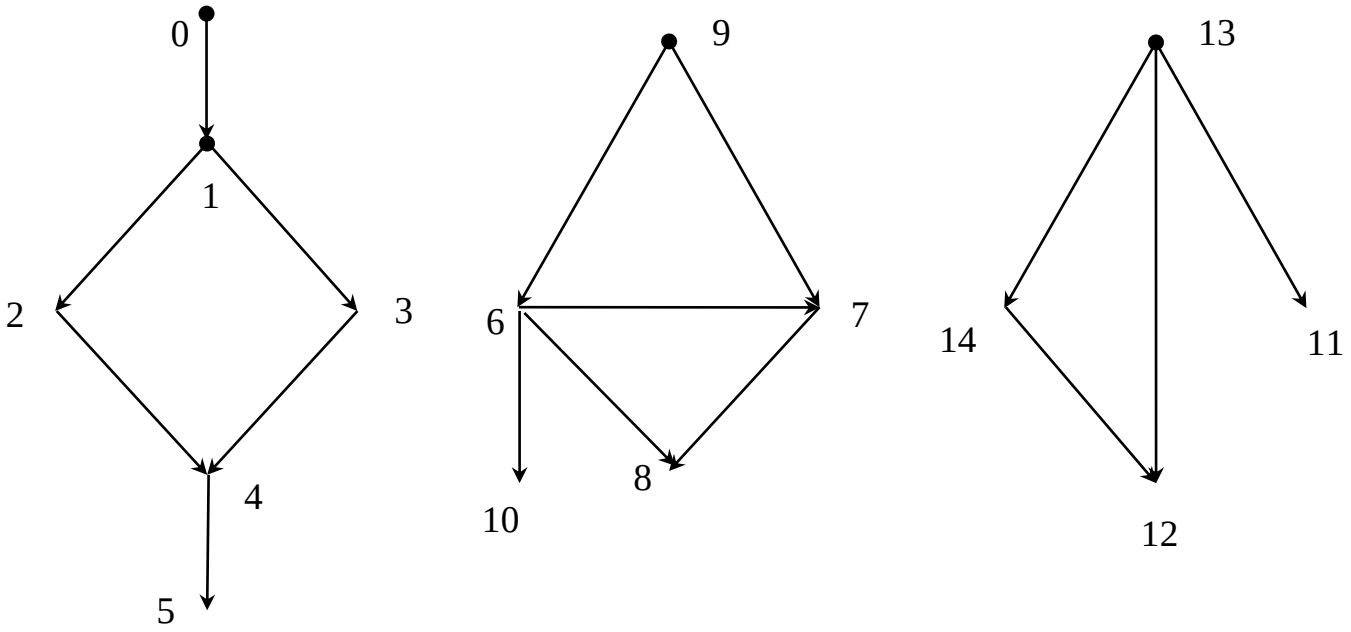

Рис. 3.

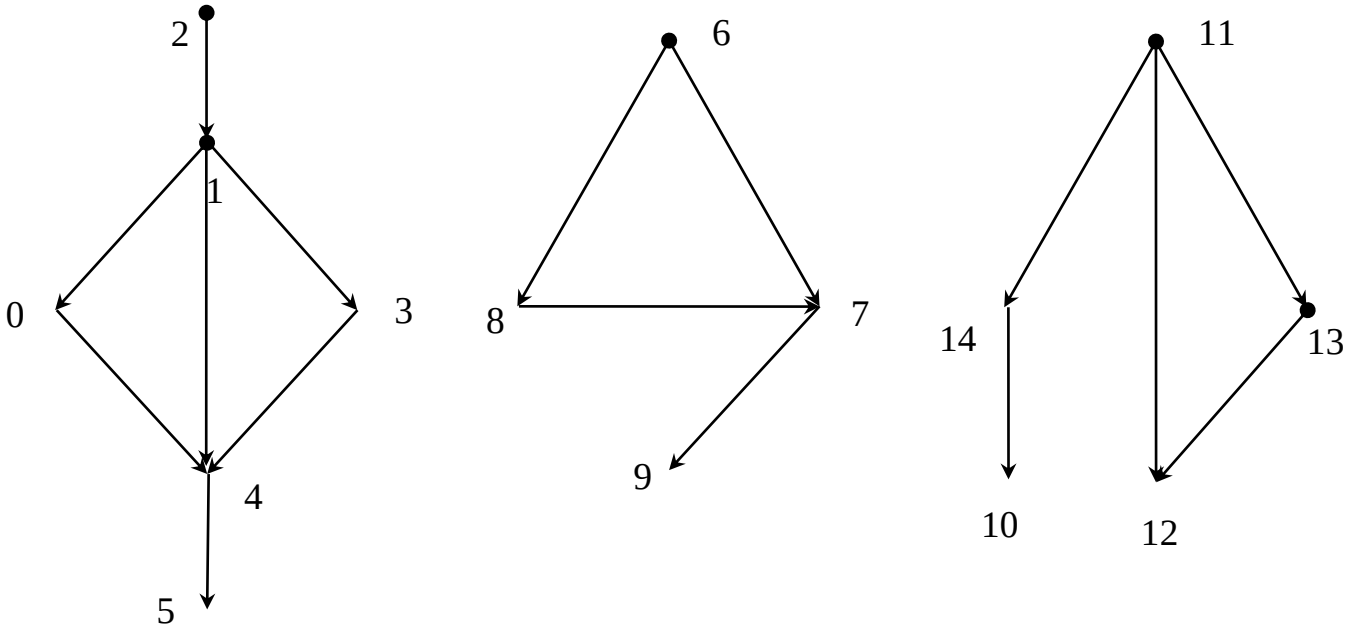

Рис. 4.

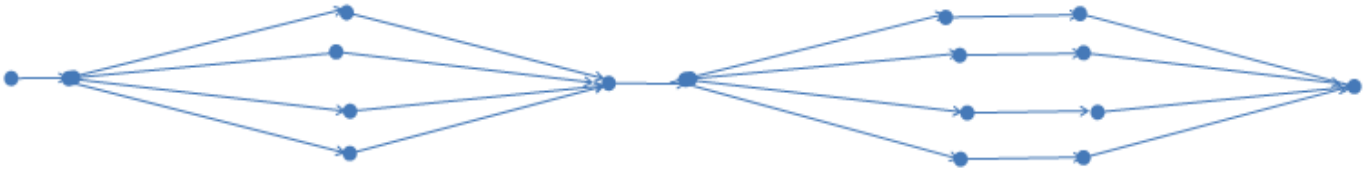

Рис. 5.

Рис. 6.

Лабораторна робота № 2

Граф алгоритму та концепції паралелізму

Мета та завдання роботи: вироблення навичок побудови та аналізу паралельних форм обчислювальних алгоритмів.

Вказівки та вимоги до роботи: у кожному варіанті потрібно зобразити 3 графи алгоритму (для кожної системи окремо). У кожному випадку вказати висоту і ширину паралельної форми алгоритму та обчислити прискорення і ефективність реалізації алгоритму за формулами

$$S_l(n) = \frac{T_1(n)}{T_l(n)}, \quad E_l(n) = \frac{S_l(n)}{l},$$

де $T_1(n)$ — час, за який можна реалізувати алгоритм на одному процесорі, $T_l(n)$ — час реалізації алгоритму в системі з l процесорів (висота алгоритму), n — розмірність задачі, $S_l(n)$ — прискорення реалізації алгоритму на цій системі, $E_l(n)$ — ефективність реалізації алгоритму у паралельній системі (завантаженість системи). У третьому завданні визначити мінімальну кількість процесорів, які забезпечують досягнення максимального можливого прискорення при паралельній реалізації алгоритму.

Приклад 2.1. Зобразити граф алгоритму паралельного обчислення значення виразу

$$a_1a_2 + a_2a_3 + a_3a_4 + a_4a_5 + a_5a_6 + a_6a_7 + a_7a_1$$

на обчислювальному пристрої

1) із одним універсальним процесором;
2) із трьома універсальними процесорами;
3) в умовах концепції необмеженого паралелізму.

Для кожної паралельної форми обчислити її висоту, ширину, прискорення та ефективність реалізації алгоритму.

Розв'язок. Будемо вважати, що у алгоритмі спочатку обчислюються зліва направо усі 7 добутків, а потім — 6 сум (у такому самому порядку).

1) Розглянемо реалізацію алгоритму у послідовній системі. Відповідний граф наведено на рис 7. Запишемо відповідні характеристики: $n = 7$, $l = 1$, $T_1(7) = 13$, $S_1(7) = 1$, $E_1(7) = 1$.

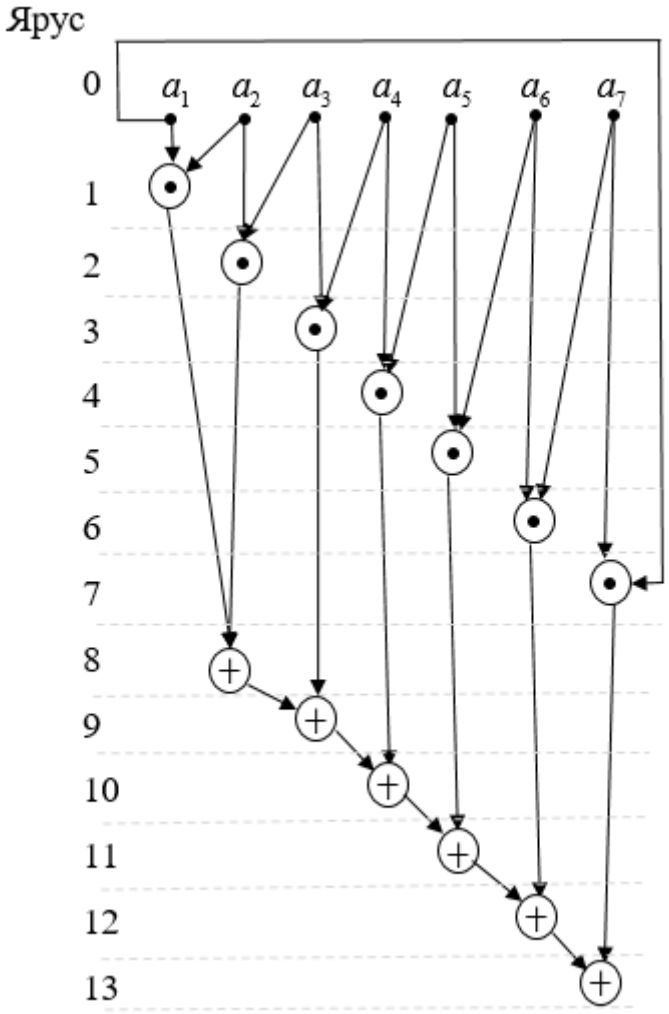

Рис. 7. Граф алгоритму у випадку $l=1$.

2) Розглянемо випадок системи з трьома процесорами. Відповідний граф алгоритму ширини 3 наведено на рис. 8.

Тоді $n=7,\ l=3,\ T_3(7)=6,\ S_3(7)=\frac{13}{6}\approx 2{,}17,\ E_3(7)=\frac{13}{6}:3=\frac{13}{18}\approx 0{,}72$.

3) Розглянемо реалізацію алгоритму у випадку системи, кількість процесорів якої необмежена. Оскільки у алгоритмі операції додавання не можуть виконуватися раніше, ніж відповідні доданки-множники будуть обчисленні, то з урахуванням твердження 3.3 для висоти алгоритму $T_l(7)$ справджується оцінка

$$T_l(7)\le 1+\lceil \log_2 7\rceil = 4.$$

У випадку $l=6$ можна вказати алгоритм висоти 4, граф якого наведено на рис. 9.

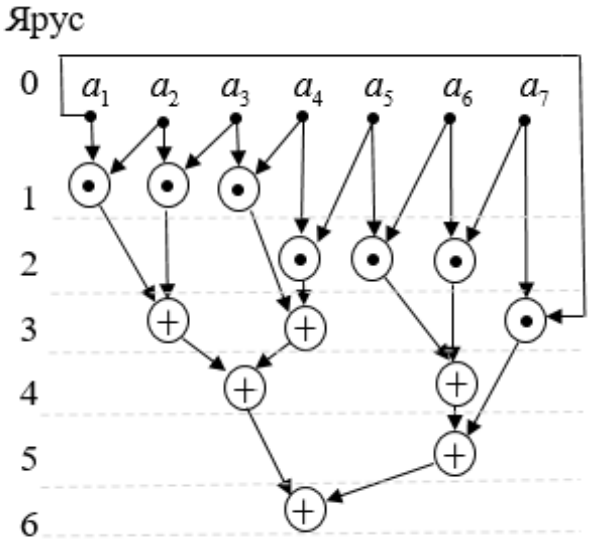

Рис. 8. Паралельна форма графа алгоритму у випадку $l = 3$.

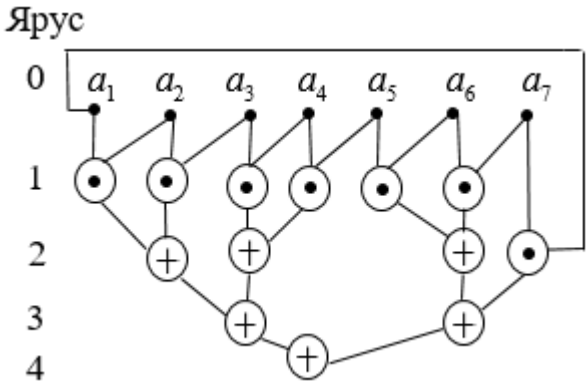

Рис. 9. Граф алгоритму у випадку $l = 6$.

Для цієї паралельної форми $l = 6, T_6(7) = 4, S_6(7) = \frac{13}{4} = 3,25, E_6(7) = \frac{13}{4} : 6 = \frac{13}{24} \approx$ $\approx 0,54$. Можна показати, що якщо $l < 6$, то $T_l(7) > 4$. Дійсно, із специфіки операцій алгоритму випливає, що на останньому ярусі виконується тільки одна операція, на передостанньому — не більше двох операцій, на третьому знизу — не більше чотирьох. Тому загалом на трьох нижніх ярусах може виконуватися не більше 7 операцій. Тому для попередніх ярусів залишається не менше, ніж $13 - 7 = 6$ операцій, які у випадку $l < 6$ не можуть бути виконані на одному ярусі. Тому кількість ярусів має бути не меншою за 5.

Варіанти завдань

Варіант 1

Зобразити граф алгоритму знаходження скалярного добутку двох 10-вимірних векторів в обчислювальній системі з:

1) чотирма універсальними процесорами;
2) в умовах концепції необмеженого паралелізму.

Варіант 2

Зобразити граф алгоритму паралельного обчислення значення виразу $a_1 + a_1 a_2 + a_1 a_2 a_3 + ... + a_1 a_2 \ldots a_7 a_8$ в обчислювальній системі з:

1) трьома універсальними процесорами;
2) в умовах концепції необмеженого паралелізму.

Варіант 3

Задана прямокутна числова матриця 3×4. Зобразити граф алгоритму знаходження суми $s_{12} + s_{13} + s_{23}$, де s_{ij} — скалярний добуток *i*-го та *j*-го рядків матриці, в обчислювальній системі із:

1) п’ятьма універсальними процесорами;
2) в умовах концепції необмеженого паралелізму.

Варіант 4

Зобразити граф алгоритму паралельного обчислення значення виразу $a_1\left(a_1 + a_2\right) \times \times\left(a_1 + a_2 + a_3\right)\left(a_1 + a_2 + \ldots + a_{10}\right)$ в обчислювальній системі із:

1) чотирма універсальними процесорами;
2) в умовах концепції необмеженого паралелізму.

Варіант 5

Побудувати граф алгоритму паралельного обчислення значення виразу

$$a_1 + a_1^2 a_2 + a_1^4 a_2^2 a_3 + ... + a_1^{32} a_2^{16} \ldots a_5^2 a_6$$

в обчислювальній системі із:

1) трьома універсальними процесорами;
2) в умовах концепції необмеженого паралелізму.

Варіант 6

Побудувати граф алгоритму обчислення значення многочлена шостої степені у точці за схемою Горнера в обчислювальній системі із:

1) двома універсальними процесорами;
2) в умовах концепції необмеженого паралелізму.

Варіант 7

Побудувати граф алгоритму паралельного обчислення суми квадратів елементів 16-вимірного вектора в обчислювальній системі із:

1) чотирма універсальними процесорами;
2) в умовах концепції необмеженого паралелізму.

Варіант 8

Зобразити граф алгоритму паралельного обчислення значення виразу $a_1 + a_1 \times \times(a_1 + a_2) + ... + a_1 (a_1 + a_2) \cdot \ldots \cdot (a_1 + a_2 + \ldots + a_6)$ в обчислювальній системі із:

1) трьома універсальними процесорами;
2) в умовах концепції необмеженого паралелізму.

Варіант 9

Зобразити граф алгоритму знаходження добутку двох 3×3-матриць в обчислювальній системі із:

1) п’ятьма універсальними процесорами;
2) в умовах концепції необмеженого паралелізму.

Варіант 10

Задана прямокутна числова матриця розмірності 3×5. Зобразити граф алгоритму знаходження суми $p_1 + p_2 + p_3$, де p_i — добуток елементів *i*-го рядка матриці, в обчислювальній системі із:

1) чотирма універсальними процесорами;
2) в умовах концепції необмеженого паралелізму.

Лабораторна робота № 3

Паралельні обчислення з використанням багатопотокового програмування

Мета та завдання роботи: вироблення навичок реалізації обчислювальних алгоритмів з використанням засобів багатопотокового програмування.

Вказівки до роботи: У кожному варіанті потрібно написати багатопотокову версію програми на одній із сучасних мов програмування (Java, C++, C#, …) та виміряти її прискорення на кількох тестових прикладах.

Приклад 3.1. Написати програму обчислення квадрату довжини *n*-вимірного вектора із використанням розпаралелення обчислень.

Розв'язок. Нижче наведено багатопотокову програму на мові Java. Для багатопотокових обчислень використовуються об'єкти класу `Adder`, які реалізовують інтерфейс `Runnable` шляхом перевизначення методу `Run()`.

```
class Adder implements Runnable {
    double[] array;
    int from;
    int to;
    private double result;

    public Adder(double[] array, int from, int to) {
        this.array = array;
        this.from = from;
        this.to = to;
    }

    public void run() {
        result = 0;
        for (int i = from; i < to; i++) {
            result += array[i]*array[i];
        }
    }

    public double getResult(){
        return result;
    }
}

public class MyClass {
    public static void main(String[] args){
        final int N = 60_000_000;
        final int ThreadCount = 4;
        double[] array = new double[N];
        for (int i = 0; i < N; i++) {
```

```
            array[i] = Math.random();
        }
        int portionSize = N / ThreadCount;
        Adder[] adders = new Adder[ThreadCount];
        Thread[] threads = new Thread[ThreadCount];
        long start = System.nanoTime();
        for (int i = 0; i < ThreadCount; i++) {
            adders[i] = new Adder(array, i*portionSize, (i+1)*portionSize);
            threads[i] = new Thread(adders[i]);
            threads[i].start();
        }
        double result = 0;
        for (int i = ThreadCount*portionSize; i < N; i++) {
            result += array[i];
        }
        for (int i = 0; i < ThreadCount; i++) {
            try{
                threads[i].join();
            }
            catch (InterruptedException e){
                System.out.println("an error occured");
            }
        }
        for (int i = 0; i < ThreadCount; i++) {
            result += adders[i].getResult();
        }
        long duration1 = System.nanoTime() - start;
        System.out.printf("Length is %g. Duration in parallel mode is
%d", result, duration1);
        start = System.nanoTime();
        result = 0;
        for (int i = 0; i < N; i++) {
            result += array[i]*array[i];
        }
        long duration2 = System.nanoTime() - start;
        System.out.printf("\n Length is %g. Duration in serial mode is
%d\n", result, duration2);
        System.out.printf("Speed-up is %g", (double)duration2 / duration1);
    }
}
```

Варіанти завдань

Варіант 1

Задана прямокутна числова матриця $m \times n$. Написати паралельну програму знаходження пари рядків, скалярний добуток яких найменший.

Варіант 2

Написати програму для паралельного пошуку моди варіаційного ряду довжини n (значення, яке найчастіше зустрічається серед елементів числового масиву).

Варіант 3

Задана квадратна числова $n \times n$-матриця $A = \left(a_{ij}\right)$. Написати паралельну програму знаходження кількості пар індексів i та j, для яких величина $S(i, j) = \sum_{k=0}^{n-1} a_{ki} a_{j,n-k-1}, \left(i, j \in \overline{0, n-1}\right)$ приймає від'ємне значення.

Варіант 4

Написати програму для паралельного множення послідовності перестановок.

Варіант 5

Написати паралельну програму для знаходження кількості спільних елементів двох числових масивів (значень, які зустрічаються в обох масивах).

Варіант 6

Задана прямокутна числова матриця $m \times n$. Написати паралельну програму знаходження пар стовпців матриці, манхетенська відстань між якими є найбільшою.

Варіант 7

Вагою елемента матриці A назвемо суму відмінних від нього елементів матриці, які містяться у одному рядку чи одному стовпці з ним. Написати паралельну програму для знаходження елементів найбільшої ваги.

Варіант 8

Задана квадратна числова матриця A. Назвемо порядком елемента кількість відмінних від нуля елементів, які розташовані на діагоналях матриці, на перетині яких міститься цей елемент. Написати паралельну програму для знаходження суми елементів найбільшого порядку.

Варіант 9

Написати паралельну версію програми знаходження кількості входжень однієї послідовності у іншу послідовність (вважати, що послідовності — одновимірні масиви).

Варіант 10

Написати паралельну версію програми знаходження такої $k \times k$ підматриці B заданої числової $n \times n$ матриці A, сума усіх елементів якої найбільша (підматриця має бути розташована на перетині k послідовних рядків та стовпчиків матриці A).

Лабораторна робота № 4

Технологія OpenMP

Вказівки до роботи: У кожному варіанті потрібно виконати два завдання із використанням технології OpenMP.

Мета та завдання роботи: ознайомлення із можливостями технології OpenMP та використання OpenMP в межах концепції внутрішнього паралелізму.

Перше завдання — програмна реалізація паралельного алгоритму із 2-го завдання відповідного варіанта лабораторної роботи № 2.

Друге завдання — розробка паралельної програми. Потрібно побудувати графіки залежності прискорення реалізації алгоритму $S_m(n)$ від розмірності задачі у випадку використання m потоків $\left(m \in \{2,4,6,8,10,20\}\right)$.

Приклад 4.1. З використанням OpenMP написати програму обчислення матричної норми $\|A\|_1 = \max\limits_{1 \le j \le n} \sum\limits_{i=1}^{m} |a_{ij}|$. Нижче наведено розв'язок, розроблений з використанням парадигми ітеративного паралелізму [8].

```
#include "stdafx.h"
#include<stdio.h>
#include<omp.h>
#include<iostream>
#include<fstream>
using namespace std;

double norm_1(double** matrix, int m, int n) {
    double norm = -1;
    for (int j = 0; j < n; j++) {
        double sum = 0;
        for (int i = 0; i < m; i++)
            sum += matrix[i][j]>=0 ? matrix[i][j] : -matrix[i][j];
        if (sum > norm) norm = sum;
    }
    return norm;
}

double norm_1_parallel(double** matrix, int m, int n, int thread_count) {
    double norm = -1;
#pragma omp parallel for num_threads(thread_count)
    for (int j = 0; j < n; j++) {
        double sum = 0;
```

```
        for (int i = 0; i < m; i++)
            sum += matrix[i][j] >= 0 ? matrix[i][j] : -matrix[i][j];
        if (sum > norm)
#pragma omp critical
        {
            if(sum > norm) norm = sum;
        }
    }
    return norm;
}

int main()
{
    const int DIM_COUNT = 7;
    int dims[DIM_COUNT] = { 100, 500, 1000, 2000, 3000, 5000, 10000};
    const int THREADS_COUNT = 6;
    int threads[THREADS_COUNT] = { 2, 4, 6, 8, 10, 20 };
    double** matrix = new double*[dims[DIM_COUNT-1]];
    cout << "Initialization...\n";
    for (int i = 0; i < dims[DIM_COUNT - 1]; i++) {
        matrix[i] = new double[dims[DIM_COUNT - 1]];
        for (size_t j = 0; j < dims[DIM_COUNT - 1]; j++)
            matrix[i][j] = 2*(double)rand() / (RAND_MAX + 1) - 1;
    }
    fstream fs = fstream("d:\\result.txt", ios::out);
    for (int d = 0; d < DIM_COUNT; d++) {
        int m = dims[d], n = dims[d];
        double start = omp_get_wtime();
        double norm = norm_1(matrix, m, n);
        double duration = omp_get_wtime() - start;
        cout << "Dimension: " << m << '\n';
        printf("\tIn serial mode ||A||_1 = %e; duration is %e\n", norm, duration);
        for (int t = 0; t < THREADS_COUNT; t++) {
            start = omp_get_wtime();
            norm = norm_1_parallel(matrix, m, n, threads[t]);
            double speedup = duration / (omp_get_wtime() - start);
            printf("\tFor %d threads ||A||_1 = %e; speedup is %e\n",
threads[t], norm, speedup);
            fs << speedup << '\t';
        }
        fs << '\n';
    }
    fs.close();
    for (size_t i = 0; i < dims[DIM_COUNT - 1]; i++)
        delete[] matrix[i];
    delete[] matrix;
}
```

Варіанти завдань

Варіант 1

Написати програму для обчислення матричних норм $\|A\|_\infty$ та $\|A\|_F$.

Варіант 2

Написати програму для обчислення відстаней між *n*-вимірними дійсними векторами за формулами: $d_p(\mathbf{x},\mathbf{y}) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$, $p \in \mathbb{N}$, $d_\infty(\mathbf{x},\mathbf{y}) = \max\limits_{1 \le i \le n} |x_i - y_i|$.

Варіант 3

Задана прямокутна числова матриця $m \times n$. Написати програму знаходження пари рядків, скалярний добуток яких найменший.

Варіант 4

Написати програму для паралельного швидкого сортування.

Варіант 5

Написати програму для паралельного сортування методом вибору.

Варіант 6

Написати паралельну реалізацію алгоритму сортування злиттям.

Варіант 7

Написати паралельну програму знаходження розв'язку системи лінійних алгебраїчних рівнянь.

Варіант 8

Написати паралельну програму обчислення детермінанта матриці.

Варіант 9

Написати паралельну версію решета Ератосфена.

Варіант 10

Написати паралельну версію програми наближеного обчислення визначеного інтегралу за формулою $\int_a^b f(x)dx$ методами прямокутників, трапецій та Сімпсона.

Лабораторна робота № 5

Паралельне багатопотокове програмування на базі сучасних фреймворків

Мета та завдання роботи: ознайомлення із підтримкою парадигми паралельного програмування в сучасних фреймворках та забезпечення синхронізації даних за допомогою взаємного виключення та умовної синхронізації.

Вказівки до роботи: Завдання потрібно виконати з використанням багатопотокового програмування (на мовах C#, C++, Kotlin, Java, …). Написати дві версії класів програми. У першій реалізувати синхронізацію потоків за допомогою *семафорів*, у другій — *моніторів*. Написати програму, яка моделює паралельне функціонування об’єктів предметної області.

Варіанти завдань

Варіант 1

Розв’язати задачу про виробників-споживачів у випадку одного споживача та n виробників, які можуть використовувати буфер розміру k. Споживач має зчитати усі n_i значень, які йому має передати i-й виробник ($i = 1, \ldots, m$).

Варіант 2

Трамвай. Трамвай, у який може вміститися n пасажирів, рухається по циклічному маршруту з k зупинками, на яких входять та виходять пасажири. Реалізувати об’єкти «трамвай» та «пасажир» і змоделювати процес руху.

Варіант 3

Принтери. Нехай `n` користувачів спільно та багаторазово використовують два принтери. Перед використанням принтера користувачі викликають функцію `request`. Ця функція чекає, поки один з двох принтерів не звільниться, і повертає ідентифікатор вільного принтера. Після використання принтера користувач звільняє його, викликаючи функцію `release`.

Варіант 4

Курник. Є `n` пташенят та мама-квочка. Пташенята їдять із загальної миски, яка вміщує `F` порцій їжі. Кожне пташеня з’їдає порцію їжі, спить деякий час, а потім знову

їсть. Коли закінчується їжа, то повідомляється квочка, яка наповнює миску `F` новими порціями їжі. Далі ці дії повторюються. Реалізувати класи «курник», «курча» та ««квочка»» і змоделювати функціонування курника.

Варіант 5

Ліфт. Ліфт, у який може вміститися *m* пасажирів, стоїть на 1-му поверсі. Пасажири викликають ліфт та входять (якщо є вільні місця) і виходять з нього на потрібному їм поверсі. Реалізувати класи «ліфт» та «пасажир» і змоделювати процес руху та посадки/висадки пасажирів.

Варіант 6

Машини на мосту. До вузького мосту під'їжджають машини з півночі та півдня. Машини, які рухаються у одному напрямку, можуть долати міст одночасно, а в протилежних — ні. Змоделювати процес руху.

Варіант 7

Обід філософів. П'ять філософів сидять біля круглого столу. Вони проводять життя, чергуючи прийоми їжі та роздуми. У центрі столу знаходиться велике блюдо спагеті. У процесі їжі філософи повинні користуватися двома виделками. На жаль, їм дали всього п'ять виделок. Між кожною парою філософів лежить одна виделка і вони домовилися, що кожен буде користуватися тільки тими виделками, які лежать поруч з ним (зліва і справа). Завдання — написати програму, що моделює поведінку філософів. Програма повинна уникати ситуації, в якій всі філософи голодні, але жоден з них не може взяти обидві виделки — наприклад, коли кожен з них тримає по одній вилці і не хоче віддавати її.

Варіант 8

Вулик. Є `n` бджіл та `1` ведмідь. Вони використовують один горщик, який уміщує `H` порцій меду. Спочатку горщик порожній. Поки горщик не наповниться, ведмідь спить, потім з'їдає увесь мед та засипає. Кожна бджола (багаторазово) збирає одну порцію меду та кладе її у горщик. Бджола, яка приносить останню порцію меду та заповнює горщик, будить ведмедя.

Варіант 9

Банківський рахунок. Кілька людей (потоків) використовують спільний рахунок. Можна розміщати або знімати кошти з рахунку. Поточний баланс рівний сумі усіх вкладених грошей мінус сума знятих коштів. Баланс не може бути від'ємним. Розміщати кошти можна без затримки, при вилучені коштів можливою є пауза, поки на рахунку не буде достатньої суми. Реалізувати два методи: `deposit(amount)` та `withdraw(amount)`.

Варіант 10

Американські гірки. Є `n` потоків-пасажирів і один потік-вагончик. Пасажири чекають черги проїхати в вагончику, який вміщує `C` людей, `C < n`. Вагончик може їхати тільки заповненим. Напишіть коди потоків-пасажирів і потоку-вагончика і розробіть засоби для їх синхронізації. Реалізувати три операції: `takeRide`, яку викликають пасажири, `load` і `unload`, які викликає потік-вагончик.

Література

1. Коцовський В.М. Технологія програмування та створення програмних продуктів: Методичний посібник. Ужгород: Видавництво УжНУ "Говерла", 2017. 60 с.
2. Коцовський В. М. Технології розподілених систем та паралельних обчислень. Частина II: Методичний посібник. Ужгород: Видавництво УжНУ "Говерла", 2017. 76 с.
3. Коцовський В. М. Теорія паралельних обчислень. Частина I: Методичний посібник. Ужгород: Видавництво УжНУ "Говерла", 2019. 51 с.
4. Коцовський В. М. Теорія паралельних обчислень. Частина II: Методичний посібник. Ужгород: Видавництво УжНУ "Говерла", 2019. 52 с.
5. Воеводин В. В., Воеводин Вл. В. Параллельные вычисления. СПб.: БХВ-Петербург, 2002. 608 с.
6. Качко Е. Г. Параллельное программирование: Учебное пособие. Харьков: "Форт", 2011. 528 с.
7. Антонов А. С. Параллельное программирование с использованием технологии OpenMP. М.: Изд-во МГУ, 2009. 77 с.
8. Эндрюс Г. Р. Основы многопоточного, параллельного и распределенного программирования. М.: Издательский дом "Вильямс", 2003. 512 с.
9. Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ, 2-е издание. М.: "Вильямс", 2005. 1296 с.
10. Шилдт Г. Java. Полное руководство. М.: ООО "И.Д. Вильямс", 2012. 1104 с.
11. Троелсен Э. Язык программирования C# 2010 и платформа .NET 4.0, 5-е изд. М.: "Вильямс", 2011. 1392 с.
12. Коцовський В. М. Основи дискретної математики: навчальний посібник. Ужгород: ПП «АУТДОР-ШАРК», 2020. 128 с.
13. Коцовський В. М. Дискретна математика та теорія алгоритмів. Частина I: Конспект лекцій для студентів спеціальностей: 6.122 — "Комп’ютерні науки", 6.121 — "Інженерія програмного забезпечення". Ужгород: Видавництво УжНУ "Говерла", 2019. 52 с.
14. Коцовський В. М. Дискретна математика та теорія алгоритмів. Частина II: Конспект лекцій для студентів спеціальностей: 6.122 — "Комп’ютерні науки", 6.121 — "Інженерія програмного забезпечення". Ужгород: Видавництво УжНУ "Говерла", 2019. 52 с.