

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»

Кафедра ЕОМ



ЗВІТ

З ЛАБОРАТОРНОЇ РОБОТИ № 3

З ДИСЦИПЛІНИ: «ПАРАЛЕЛЬНІ ТА РОЗПОДІЛЕНІ ОБЧИСЛЕННЯ»

НА ТЕМУ: «ПАРАЛЕЛЬНЕ ПРЕДСТАВЛЕННЯ АЛГОРИТМІВ.»

ВАРІАНТ 17

Виконав:
ст. гр. КІ-303
Порубайміх О.Є.
Перевірив:
старший викладач
Ногаль М.В.

Львів – 2024

МЕТА

Вивчити можливості паралельного представлення алгоритмів. Набути навиків такого представлення.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Два підходи до побудови паралельного представлення алгоритму дозволяють реалізувати високоефективні рішення для обчислювальних систем з кількома процесорами або ядрами. Розглянемо ці підходи більш детально.

Векторизація алгоритмів, організованих за принципом послідовного виконання:

Векторизація — це техніка перетворення коду, написаного для послідовного виконання, у форму, що дозволяє його паралельне виконання на векторних або паралельних процесорах. Цей процес зазвичай автоматизується за допомогою компілятора, який аналізує залежності між операторами вихідного коду. Компілятор шукає можливості для паралельного виконання незалежних операцій, що не впливають одна на одну. Такий підхід може значно прискорити виконання програм на апаратному рівні, використовуючи можливості сучасних багатоядерних та векторних процесорів.

Пряме представлення паралельних алгоритмів:

Цей підхід охоплює кілька методик розробки, які дозволяють ефективно використовувати паралельні обчислювальні можливості:

1. Кадри: Цей метод дозволяє описувати обчислювальні дії, які відбуваються в певний момент часу, що є корисним для аналізу та перевірки правильності алгоритмів.
2. Програми з одноразовим присвоєнням: У такому стилі програмування кожна змінна в алгоритмі отримує значення лише один раз, що спрощує відстеження станів у масштабованих системах і знижує ризик виникнення помилок у багатопотокових додатках.
3. Рекурсивні рівняння: Вони представляють алгоритм як систему рівнянь з використанням правил одноразового присвоєння, здатних описати поведінку алгоритму у різних часових і просторових умовах, що ідеально


підходить для паралельних обчислень.

4. Графи залежностей: Це візуальне представлення, яке демонструє всі залежності між різними частинами алгоритму. Такий граф не тільки допомагає розуміти взаємозв'язки між різними процесами, але й є ключовим у оптимізації паралельного виконання.

Обидва описані підходи сприяють підвищенню продуктивності систем шляхом ефективного розподілу обчислювальних ресурсів і можуть бути використані разом або окремо залежно від потреб програми та особливостей використовуваної апаратної платформи.

ВИКОНАННЯ РОБОТИ

Варіант завдання:

$ \begin{array}{cccc} 1 & 2 & 3 & \dots n \\ & & & \dots \\ n-2 & n-1 & n & \dots \\ n-1 & n & 0 & \dots 0 \\ n & 0 & 0 & \dots 0 \end{array} $	
---	---

Написав програму, що рахує добуток матриць з багаторазовим і одноразовим присвоєнням.

Результат роботи для матриці 3×3:

```
0 0 -243

Matrix C = A * B, program with one-time assignment :
0 0 -15
0 0 -153
0 0 -243

Matrix C = A * B, program with the localized algorithm:
0 0 -15
0 0 -153
0 0 -243

Matrix C = A * B, program with the recursive localized algorithm :
0 0 -15
0 0 -153
0 0 -243

n optimized 11
n unoptimized: 54
Process finished with exit code 0
```

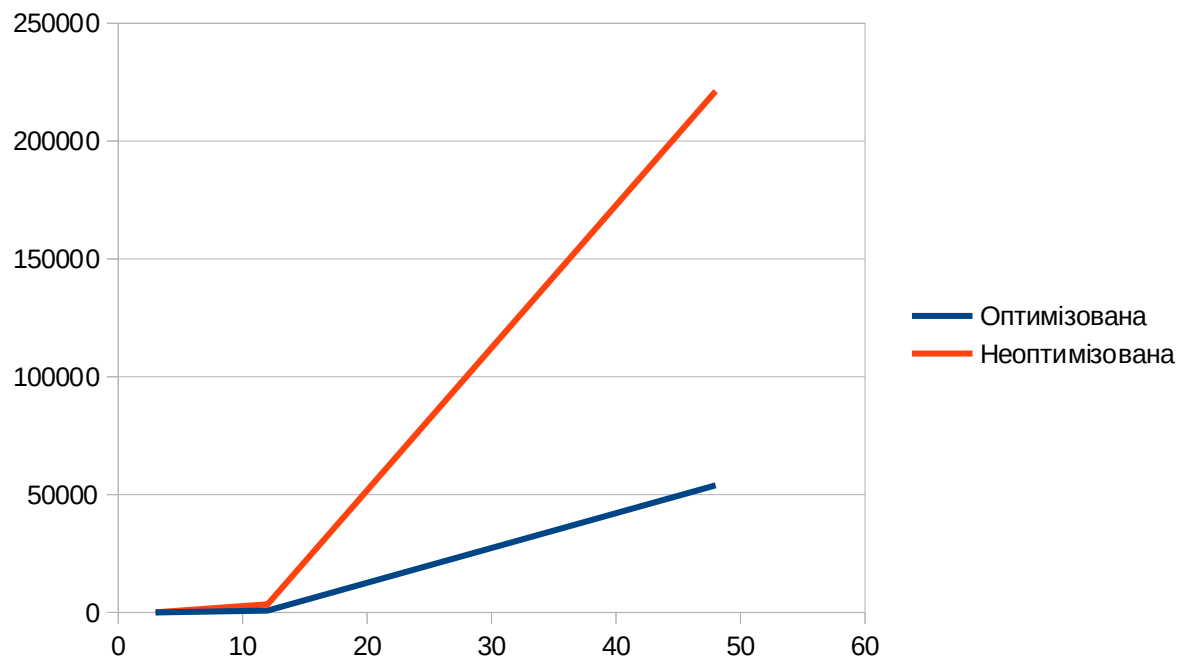
Як видно з рисунку, кількість операцій, що включають множення на 0 та

1, значно перевищує кількість операції, де ці дії виключено.

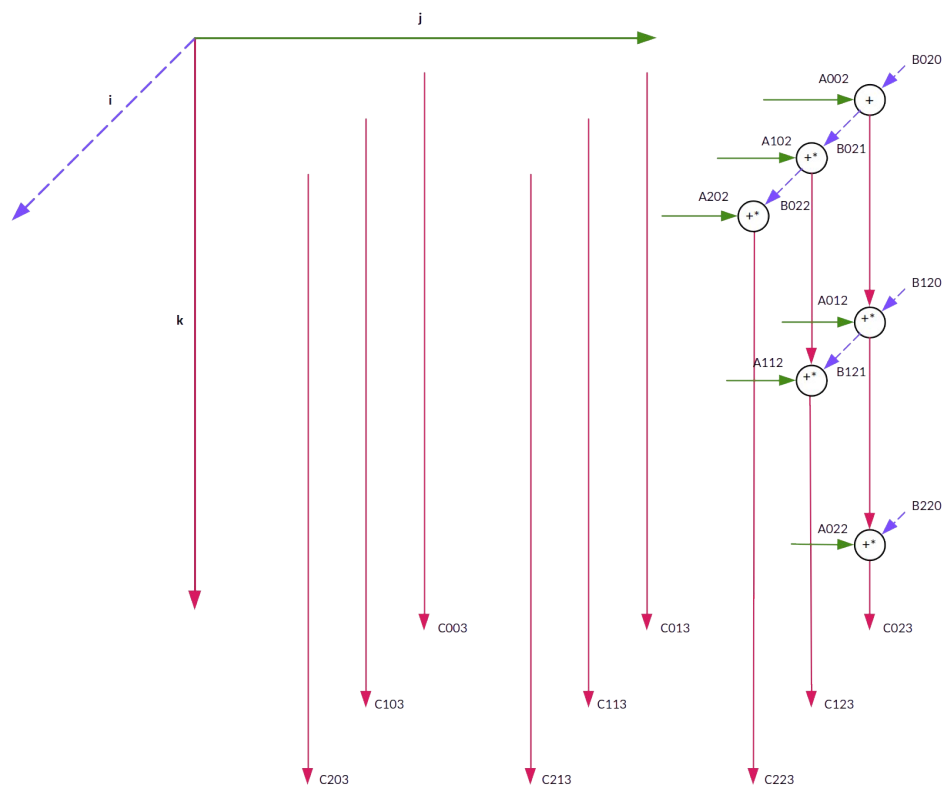
Порівняння результатів обчислень для різних N:

Кількість операцій в програмі

N		Оптимізована	Неоптимізована
3	11		54
12	775		3456
48	53955		221184



Побудував локалізовний граф для N=3:



ВИСНОВОК

Я вивчив можливості паралельного представлення алгоритмів. Набув навичок такого представлення.