значення.

3. Реалізувати послідовні функції, які обчислюють значення π (3 різні функції) з заданою точністю.
4. Визначити найкращу функцію з боку обчислювальної складності.
5. Визначити кількість ітерацій для отримання максимально можливої точності в разі використання даних типу double.
6. Реалізувати паралельно найкращу з обраних функцій. Створити за допомогою функції Windows потоки. Обрати кількість ітерацій як для послідовного варанту.
7. Зробити висновки по ефективності використання Windows потоків по часу та точності.
8. Забезпечити паралельне виконання за допомогою директив Open MP. Порівняти результати з попередніми. Зробити висновки по ефективності використання Open MP потоків.
9. Визначте значення прискорення для паралельних виконань програми. Порівняйте ці значення для обох варіантів паралельного виконання.

4.4 Зміст звіту

1. Текст програми

2. Результати дослідження точності обчислень.

3. Результати досліджень обчислювальної складності для послідовного виконання, паралельного виконання за допомогою функцій Windows і паралельного виконання за допомогою Open MP.

4. Висновки

4.5 Контрольні запитання і завдання

1. Як включається режим використання Open MP?
2. Як виконуються директиви Open MP, якщо режим Open MP не вмикнуно?
3. Як визначається кількість ядер процесору?
4. Що таке прискорення для паралельного виконання програми?

5 ВИКОРИСТАННЯ ОБ'ЄКТІВ СІНХРОНІЗАЦІЇ ПРИ СТВОРЕННІХ ПРОГРАМ

5.1 Мета роботи

При створенні реальних багато поточних програм виникають проблеми синхронізації,

пов’язані з виконанням критичних секцій, запобіганню блокувань, тощо. Для вирішення цих задач є декілька засобів. Метою цієї лабораторної роботи є дослідження засобів синхронізації в Open MP, порівняння їх з засобами Windows. В результаті виконання лабораторної роботи необхідно навчитися практичному застосуванню засобів синхронізації в Open MP.

5.2 Подготовка до роботи

Повторити засоби синхронізації , які забезпечує операційна система Windows (Interlocked функції, критичні секції, події, мьютекси, семафори, очикуючі таймери.) [1].

Вивчити засоби синхронізації Open MP за допомогою директив.

Вивчити засоби синхронізації Open MP за допомогою функцій.

5.2.1 Засоби синхронізації Open MP за допомогою директив

5.2.1.1 Директива *atomic*

Аналогічна використанню Interleave – функцій

Загальний вид директиви:

#pragma omp atomic

expression

Змінна, значення якої змінюється в *expression*, захищена від повторного запису, тобто використовується в режимі ексклюзивного доступу.

Приклад використання:

```
#pragma omp atomic
    count++;
```

5.2.1.2 Директиви barrier, nowait

По замовченню, при паралельному виконанні окремих секцій і циклов перехід на наступний оператор програми після паралельної ділянки виконується тільки після того, як виконані усі паралельні гілки (Це аналогічно тому що наприкінці паралельної ділянки ставиться функція WaitForMultiplyObjects для усіх потоків, час очикування INFINITE, *bWaitAll* = *true*).

Якщо таке очикування необхідно добавити не тльки в кінці паралельного блоку, а і в інших місцях, то треба використовувати директиву barrier.

Якщо таке очикування не обов’язкове в кінці паралельного блоку, використовується директива *nowait*.

Загальний вид директиви *barrier*::

#pragma omp barrier

Приклад використання директиви nowait для відключення чекання завершення

виконання усіх гілок циклу:

#pragma omp for nowait.

5.2.1.3 Директива critical

Використовується для завдання критичної секції.

Загальний вид директиви:

```
#pragma omp critical [(name)]
{
    code_block
}
```

де:

name – ім'я критичної секції, для цього імені компілятор виконує функції ініціалізації критичної секції. Якщо ім'я відсутнє, то компілятор використовує ім'я по замовченню.

Аналогічна функціям EnterCriticalSection та LeaveCriticalSection для code_block.

5.2.2 Засоби синхронізації Open MP за допомогою функцій

Функції дозволяють організувати критичні секції, тобто їх використання фактично дублює використання директиви critical.

5.3 Порядок виконання лабораторної роботи

Скласти послідовну та паралельну програми для пошуку ключових даних.

Перший файл складається з упорядкованих в алфавітному порядку ключів, кожний записан 1 в одному рядку і складається з 15 символів. Перший рядок файла – кількість ключів. Файл гарантовано поміщається в оперативній памяті

Другий файл складається з ключів, які треба знайти. Файл може не поміщатися в оперативній памяті. Для кожного ключа з другого файлу треба знайти його в першому файлі. Результат записати в третій файл. Формат запису для рядків, які знайдено: ….. is found at index … . Формат запису для рядків, які не знайдені: … is NOT FOUND.

Результати для обох програм повинні співпадати.

Порівняти програми по потрібному часу для виконання.

Обчислити показники для паралельного режиму

Приклади даних

Файл 1:

10

123456789012345

AABBCCDDEEFFGGH

MMNNNNNNNNNNNNN

MMNNNNNNNNNNNNO

NNNNNNNNNN12345

NNNNNNNNNN12346

This is a key22

aabbCCDDEEFFGGh

mMNNNNNNNNNNNNN

not the lastkey

Файл 2.

MMNNNNNNNNNNNNN

NOT the lastkey

mMNNNNNNNNNNNNN

AABBCCDDEEFFGGH

AABBCCDDEEFFGG0

Файл результату

MMNNNNNNNNNNNNN is found at index 2

NOT the lastkey is NOT FOUND

mMNNNNNNNNNNNNN is found at index 8

AABBCCDDEEFFGGH is found at index 1

AABBCCDDEEFFGG0 in NOT FOUND

5.4 Зміст звіту

1 Текст програми

2 Обчислювальна складність для обох режимів.

3. Показники для паралельного режиму.

4 Висновки

5.5 Контрольні запитання і завдання

1. Які директиви використовують для синхронізації в Open MP?
2. Які функції використовують для синхронізації в Open MP?

3. Складіть таблицю відповідності функцій синхронізації Windows і засобів синхронізації в Open MP.
4. Як знайти ділянки коду, які можна виконувати паралельно?

6 РОЗРОБКА БІБЛІОТЕКИ ДЛЯ МАТРИЧНИХ ОБЧИСЛЕНЬ З ВИКОРИСТАННЯМ ПАРАЛЕЛЬНИХ АЛГОРИТМІВ

6.1 Мета роботи

Багато практичних задач пов'язані з виконанням різних операцій над матрицями. Ці операції, як за правило, допускають паралельну обробку. Метою даної лабораторної роботи є розробка бібліотеки для виконання арифметичних операцій для квадратних матриць

6.2 Подготовка до роботи[5]

1. Вивчить методи складання, множення вектора на матрицю та матриць, транспонування для квадратних матриць
2. Для цих методів визначте ділянки, які можна виконувати паралельно.
3. Вивчить особливості виконання операцій для матриць з бітовими елементами

6.3 Порядок виконання лабораторної роботи

1. Складіть функції для довання матриць загального виду для послідовного режиму виконання.
2. Складіть функції для довання бітових матриць для послідовного режиму виконання.
3. Складіть функцію для множення матриць для послідовного режиму виконання
4. Складіть функцію для множення бітових матриць для послідовного режиму виконання
5. Оптимізуйте складені функції за часом виконання.
6. Переведіть усі вище розроблені функції в паралельний режим.
7. Визначте показники для паралельного режиму для усіх функцій

6.4 Зміст звіту

1. Функції для послідовного режиму
2. Функції для паралельного режиму

---

[5] Для отримання найвищої оцінки замість звичайних використовувати бітові матриці