ОТЧЁТ ПО ТЕМЕ

«Реализация метода простой итерации для СЛАУ»

по дисциплине «Параллельное программирование на суперкомпьютерных системах»

Выполнил:

студент гр. 3540201/20302 _____Гордейчук А.С.

подпись, дата

Проверил:

к.т.н., доцент _____Лукашин А.А.

подпись, дата

Санкт-Петребург, 2023

Постановка задачи

1. Выбрать задачу и проработать реализацию алгоритма, допускающего распараллеливание на несколько потоков / процессов.
2. Разработать тесты для проверки корректности алгоритма (входные данные, выходные данные, код для сравнения результатов). Для подготовки наборов тестов можно использовать математические пакеты, например, matlab (есть в классе СКЦ и на самом СКЦ).
3. Реализовать алгоритмы с использованием выбранных технологий.
4. Провести исследование эффекта от использования многоядерности / многопоточности / многопроцессорности на СКЦ, варьируя узлы от 1 до 4 (для MPI) и варьируя количество процессов / потоков.

Введение

Метод простой итерации (или метод Якоби) является одним из простейших итерационных методов решения систем линейных алгебраических уравнений (СЛАУ).

Механизмы распараллеливания вычислений:

- Pthreads (POSIX threads) - это стандартная библиотека языка Си, которая предоставляет многопоточную поддержку на уровне операционной системы. Pthreads обеспечивает возможность создавать и управлять множеством потоков выполнения в одном процессе. Каждый поток выполняет код независимо от других потоков, но имеет доступ к общей памяти процесса, что позволяет им взаимодействовать друг с другом и совместно выполнять задачи. Pthreads также обеспечивает механизмы синхронизации, такие как мьютексы, семафоры и условные переменные, которые позволяют потокам совместно использовать ресурсы и обмениваться информацией без конфликтов и гонок данных.
- OpenMP (Open Multi-Processing) - это стандарт для параллельного программирования, который позволяет использовать многопоточность в приложениях на языках C, C++ и Fortran. OpenMP обеспечивает простой способ создания параллельных приложений, которые могут использовать несколько ядер процессора для ускорения вычислений. OpenMP использует директивы компилятора и библиотеки для создания параллельных участков кода, которые могут выполняться параллельно на нескольких потоках. Например, директива #pragma omp parallel создает группу потоков, которые могут параллельно выполнить блок кода, помеченный директивой #pragma omp parallel for. OpenMP также предоставляет механизмы синхронизации, такие как критические секции и блокировки, которые позволяют управлять доступом к общим ресурсам и избежать гонок данных.
- MPI (Message Passing Interface) - это стандарт для обмена сообщениями между процессами в параллельных вычислениях. MPI используется для

создания распределенных приложений, которые выполняются на нескольких узлах сети и обмениваются данными через сетевое соединение. MPI определяет набор функций и директив для обмена сообщениями между процессами. Эти функции включают в себя отправку и прием сообщений, буферизацию и неблокирующие операции. MPI также определяет топологию процессов, которые образуют группы, коммуникаторы и другие структуры, которые позволяют управлять обменом сообщениями.

Реализация алгоритма

Рассмотрим СЛАУ вида Ax = b, где A - матрица коэффициентов, x - вектор неизвестных, b - вектор правой части уравнения.

Алгоритм метода простой итерации для СЛАУ:

1. Разложить матрицу A на сумму диагональной матрицы D и остаточной матрицы R: A = D + R, где D - диагональная матрица, элементы на диагонали которой равны диагональным элементам матрицы A, а R - остаточная матрица, элементы которой равны нулю на диагонали.
2. Задать начальное приближение x0.
3. Для k = 0, 1, 2, ... выполнять следующий цикл:
   - Вычислить новое приближение: $x_{k+1} = D^{-1}(b - Rx_k)$, где D^{-1} - обратная диагональная матрица.
   - Если $\|x_{k+1} - x_k\| < \epsilon$, где ϵ - заданная точность, то остановить итерационный процесс и принять x_{k+1} как решение СЛАУ.
4. Если после N итераций не достигнута заданная точность, то остановить итерационный процесс и принять x[N] как приближенное решение СЛАУ.

Тестирование алгоритма

В качестве входных параметров использовалась матрица `A` размером `1000x1000` случайных значений в диапазоне от 1 до 500 и матрица `b` размером `1x1000` с аналогичным диапазоном и различностью значений. Корректность ответов, полученых при использовании различных методов распараллеливания будет проводится с реализацией метода, в котором отсутствует какое-либо распарллеливанием (эталонное).

Ход работы

Сперва был выполнен вход на суперкомпьютер используя защищённый сетевой протокол SSH для удалённого доступа на сервер.

```
→ scc ssh -i /.ssh/id_rsa tm5u7@login1.hpc.spbstu.ru
Warning: Identity file /.ssh/id_rsa not accessible: No such file or directory.
Last login: Thu Mar  2 11:25:59 2023 from 94.19.149.192


  /$$$$$$   /$$$$$$   /$$$$$$         /$$$$$$  /$$$$$$$  /$$       /$$$$$$$  /$$   /$$
 /$$__  $$ /$$__  $$ /$$__  $$       /$$__  $$| $$__  $$| $$      | $$__  $$| $$  | $$
| $$  \__/| $$  \__/| $$  \__/      | $$  \__/| $$  \ $$| $$$$$$$ | $$  \ $$| $$  | $$
|  $$$$$$ | $$      | $$             |  $$$$$$ | $$$$$$$/| $$__  $$| $$$$$$$/| $$  | $$
 \____  $$| $$      | $$              \____  $$| $$____/ | $$  \ $$| $$____/ | $$  | $$
 /$$  \ $$| $$    $$| $$    $$       /$$  \ $$| $$      | $$  | $$| $$      | $$  | $$
|  $$$$$$/|  $$$$$$/|  $$$$$$/      |  $$$$$$/| $$      | $$$$$$$/| $$      |  $$$$$$/
 \______/  \______/  \______/        \______/ |__/      |_______/ |__/       \______/


Slurm partitions:
  * tornado     : nodes: 612
                  cpu: 2 x Intel Xeon CPU E5-2697 v3 @ 2.60GHz
        cores/hwthreads: 28 / 56
                  mem: 64G
                  net: 56Gbps FDR Infiniband
  * tornado-k40 : nodes: 56
                  cpu: 2 x Intel Xeon CPU E5-2697 v3 @ 2.60GHz
        cores/hwthreads: 28 / 56
           co-processor: 2 x Nvidia Tesla K40x
       co-processor mem: 12G
                  mem: 64G
                  net: 56Gbps FDR Infiniband

Storage: 1 PB Lustre FS

List of available software: module avail

tm5u7@login1:~
$
```

Далее необходимо было выделить свободный кластер, используя команду:

- `salloc -N1 -p tornado-k40 -t <here set time of using claster>`
- `squeue` — просмотр выделеных кластеров
- `ssh <id node>` — получение доступа к узлу

После успешного подключения к узлу, необходимо подключить модули и зависимости:

- `module purge`
- `module add compiler/gcc/11.2.0`
- `module add mpi/openmpi/4.1.3/gcc/11`
- `module add python/3.9`

Далее также нужно создать конфигурационные файлы, которые и будут запускаться. В моем случае я объеденил их в Makefile, который содержит следующий набор строк:

```
all:
	gcc justcode.c -o justcode -lm
	gcc pthreadcode.c -o pthreadcode -lm
	mpicc mpicode.c -o mpicode -lm
	gcc openmpcode.c -o openmpcode -lm -fopenmp
```

После выполнения всех этих действий можно запускать эти файлы, получать значения и сравнивать с эталонным, результаты тестирования представлены в таблице 1, было использовано 4, 10, 16 потоков и для 1го узла.

Таблица 1.

	4	10	16
MPI C	0.005248	0.003891	0.003079
MPI Python	0.129182	0.127078	0.124502
Open mp	0.019296	0.017902	0.016502
Pthread	0.044921	0.042014	0.049831
Reference		0.038217	

В таблице 2 представлено сравнинии MPI C и MPI Python, для узлов 1 — 4, количество процессов 16.

Таблица 2.

	1	2	3	4
MPI C	0.003079	0.002891	0.002034	0.002157
MPI Python	0.124502	0.122893	0.123961	0.125921

Заключение

В данной работе был исследован эффект от распараллеливания метода простой итерации на суперкомпьютере с использованием механизмов Pthread, MPI и OpenMP. В ходе исследования было установлено, что распараллеливание позволяет ускорить процесс решения системы линейных алгебраических уравнений. Для получения более качественных результатов распараллеливания необходимо иметь большую матрицу, чем была использована в данной работе.

При правильно использовании современных технологий параллельного программирования на суперкомпьютерах может ускорить процесс решения систем линейных алгебраических уравнений и повысить эффективность работы научных и инженерных приложений. Однако, необходимо учитывать, что оптимальный выбор механизмов зависит от конкретной задачи и характеристик используемого суперкомпьютера.