Московский Государственный Университет им. М.В. Ломоносова
Факультет Вычислительной Математики и Кибернетики
Кафедра Суперкомпьютеров и Квантовой Информатики

Системы и средства параллельного программирования

Отчёт № 5
Параллельный алгоритм DNS матричного умножения.

Работу выполнила
Зайденварг Елизавета

Москва 2020

Постановка задачи и формат данных

Задача

- Разработать параллельную программу с использованием технологии MPI, реализующую алгоритм умножения плотных матриц на C=AB. Тип данных – double. Провести исследование эффективности разработанной программы на системе Blue Gene/P.
- Параметры, передаваемые в командной строке:
  - имя файла – матрица A размером n x n
  - имя файла - матрица B размером n x n
  - имя файла – результат, матрица C

Формат задания матриц – как в первом задании.

Требуется

1. Разработать параллельную программу с использованием технологии MPI. Предусмотреть равномерное распределение элементов матриц блоками. Для
организации работы с файлами использовать функции MPI для работы с параллельным вводом-выводом.
2. Исследовать эффективность разработанной программы в зависимости от размеров матрицы и количества используемых процессов. Построить графики времени работы, ускорения и эффективности разработанной программы. Время на ввод/вывод данных не включать.
3. Исследовать эффективность использования параллельной работы с файлами. Для каждого из вариантов построить графики накладных расходов, связанных с вводом/выводом.
4. Исследовать влияние мэппинга параллельной программы на время работы программы.
5. Построить таблицы: времени, ускорения, эффективности.
   Для варианта 5^3 процессоров рассмотреть два варианта мэппинга – стандартный, принятый по умолчанию и произвольный. Для произвольного мэппинга предусмотреть генерацию строк файла для задания случайного значения XYZT.
6. *Ускорение* (speedup), получаемое при использовании параллельного алгоритма для p процессоров, определяется величиной:
   Speedup(n) = T 1 (n)/T p (n),
   где T 1 (n)- время выполнения задачи на одном процессоре.
   Tp(n)- время параллельного выполнения задачи при использовании p процессоров.
   7. Построить графики – для каждого из заданных значений размеров матрицы (1024x1024, 2048x2048, 4096x4096).

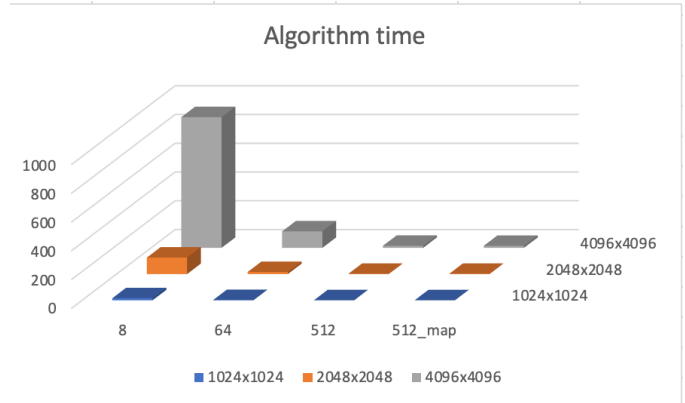
Результаты

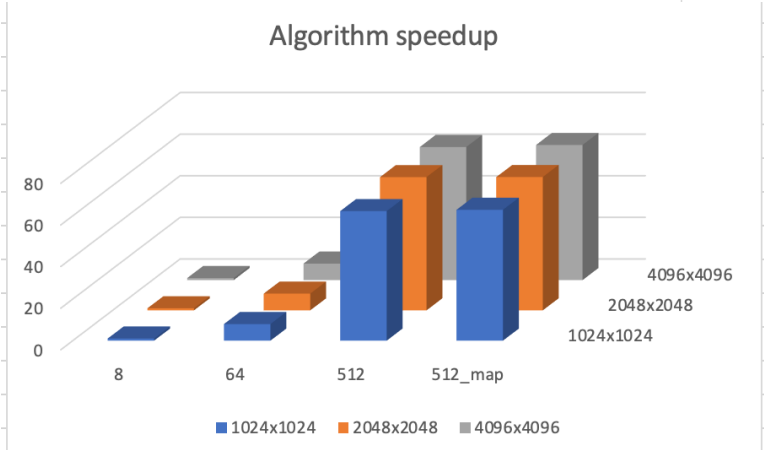
1. Алгоритм

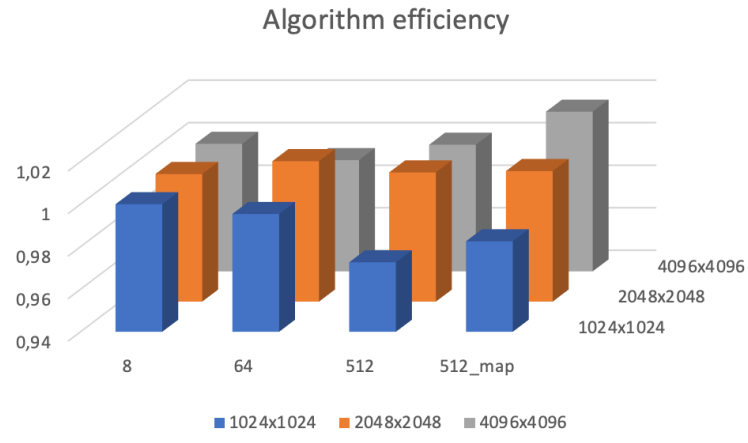
TIME			
	1024x1024	2048x2048	4096x4096
8	14,215	114,458	908,813
64	1,78507	14,2213	114,479
512	0,228351	1,78708	14,2063
512_map	0,226051	1,78591	13,9891

SPEEDUP			
	1024x1024	2048x2048	4096x4096
8	1	1	1
64	7,96327315	8,04835001	7,93868744
512	62,250658	64,0474965	63,9725333
512_map	62,8840394	64,0894558	64,9657948

EFFICIENCY			
	1024x1024	2048x2048	4096x4096
8	1	1	1
64	0,99540914	1,00604375	0,99233593
512	0,97266653	1,00074213	0,99957083
512_map	0,98256312	1,00139775	1,01509054







2. Ввод + вывод

TIME

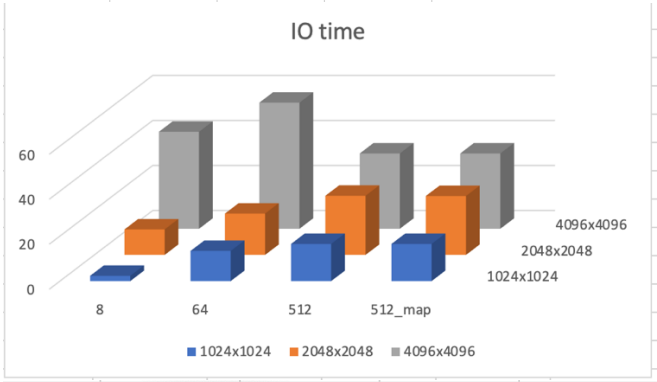
	1024x1024	2048x2048	4096x4096
8	2,33666	11,336	43,0911
64	13,3971	18,3782	56,0243
512	16,4879	26,286	33,4716
512_map	16,4912	26,175	33,4661

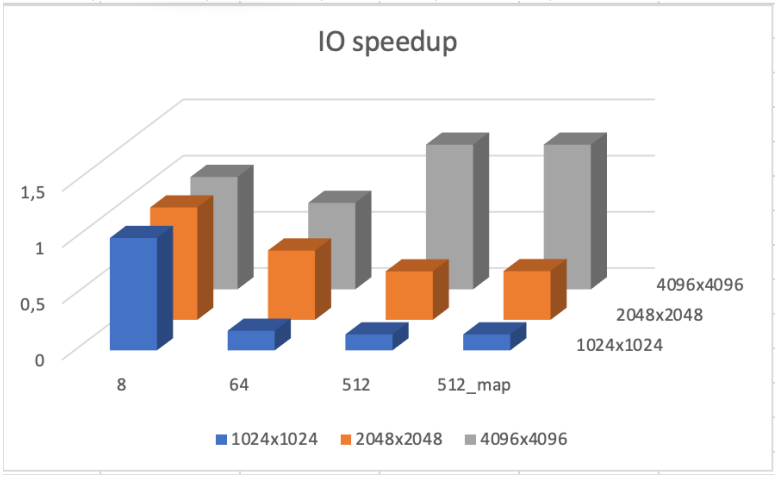
SPEEDUP

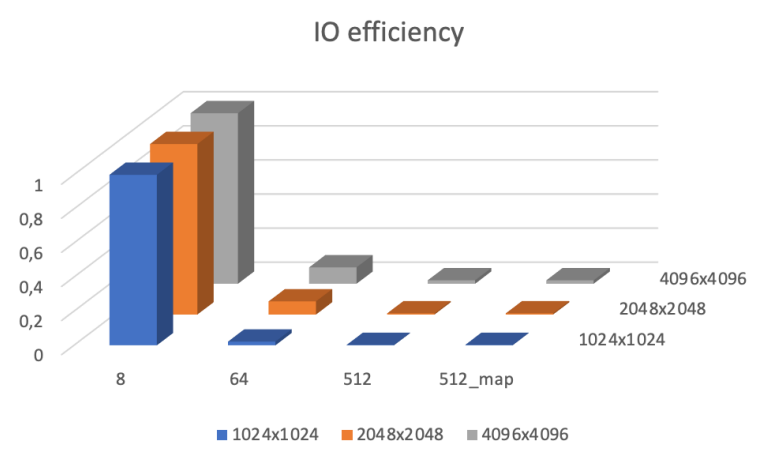
	1024x1024	2048x2048	4096x4096
8	1	1	1
64	0,17441536	0,61681775	0,76915017
512	0,14171969	0,43125618	1,28739289
512_map	0,14169133	0,433085	1,28760447

EFFICIENCY

	1024x1024	2048x2048	4096x4096
8	1	1	1
64	0,02180192	0,07710222	0,09614377
512	0,00221437	0,00673838	0,02011551
512_map	0,00221393	0,00676695	0,02011882







Выводы

Параллелизм дает хорошие результаты с ростом объема данных. Ускорение алгоритма растет до определенного предела и затем перестает увеличиваться, что связано с ростом накладных расходов на организацию параллелизма и на пересылки данных. Однако время ввода/вывода увеличивается с ростом числа процессов, что связано с накладными расходами, связанными с разделением ресурсов, и взаимной блокировкой процессов при вводе/выводе.