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

Отчёт № 4.
Анализ влияния числа процессов на время работы алгоритма умножения матрицы на вектор.

Работу выполнил
Пилюгин В.И.

Москва 2016

Постановка задачи и формат данных.

Задача: Реализовать параллельный алгоритм умножения матрицы на вектор
Формат выходного файла: бинарный файл содержащий элементы выходного вектора

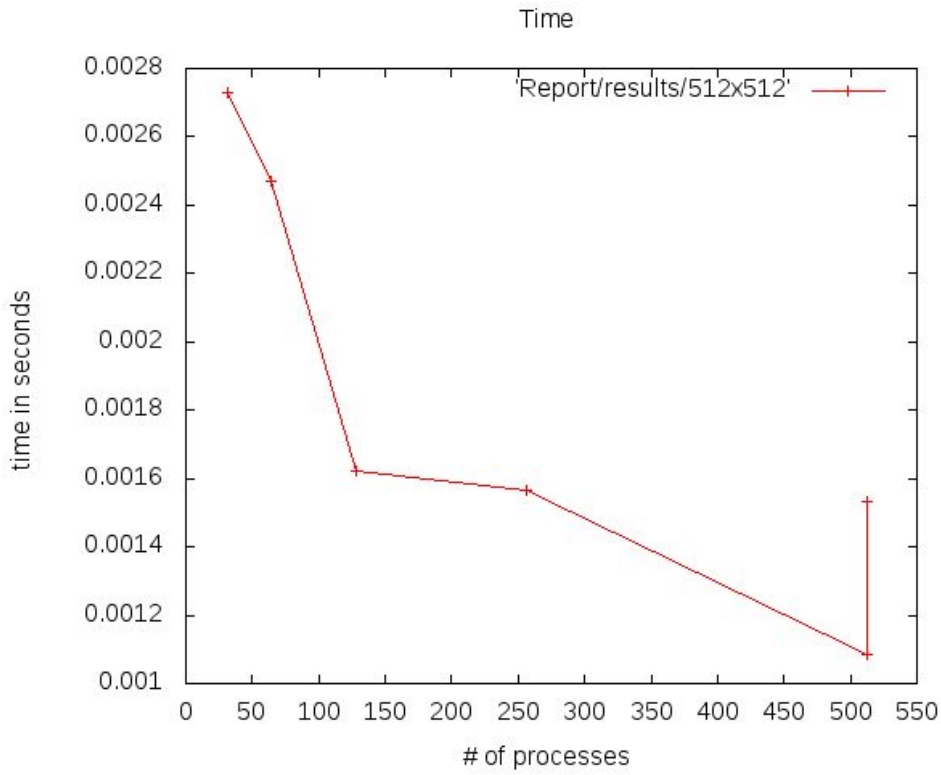
Описание алгоритма.

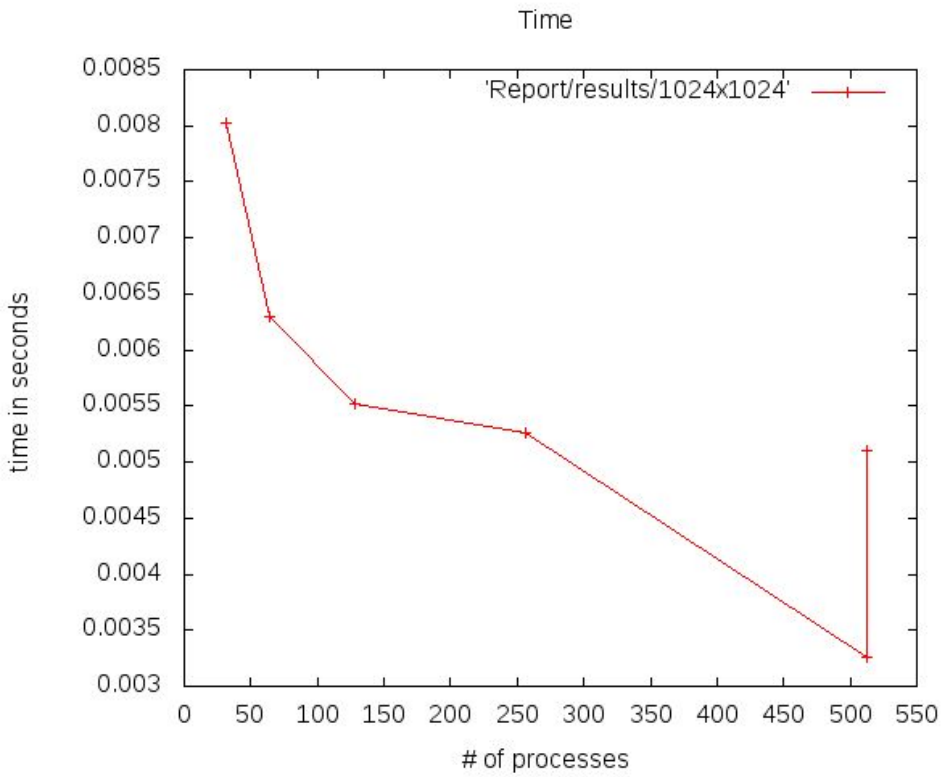
Анализ времени выполнения: Для оценки времени выполнения программы использовалась функция MPI_Wtime().
Верификация: Для проверки корректности работы программы проводилось перемножение в однопроцессорном режиме и сравнение двух файлов.
Основные функции:

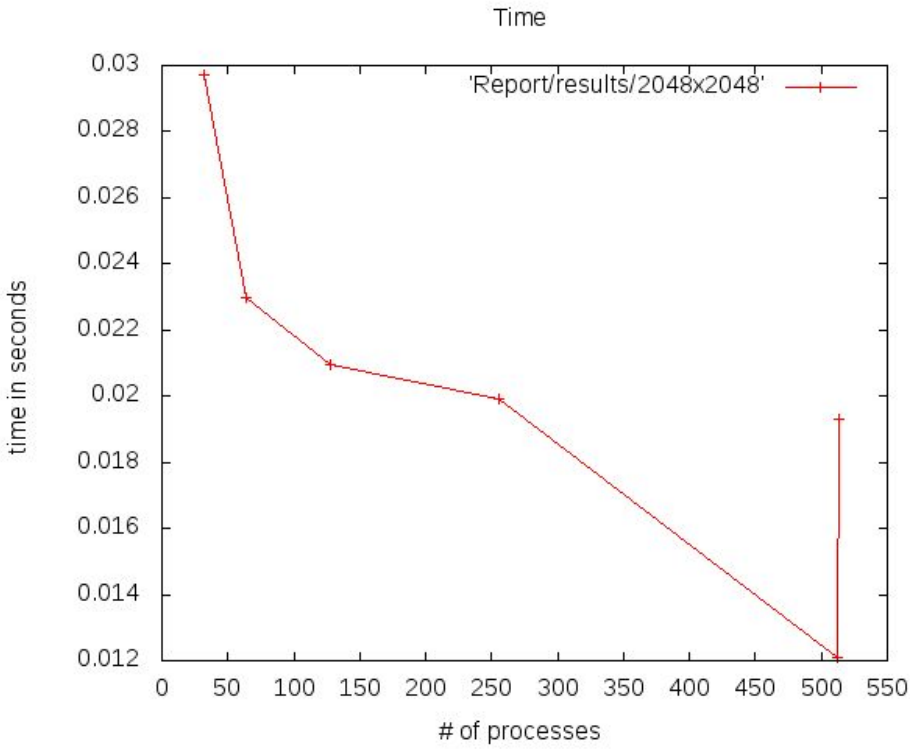
- Конфигурационный файл для задания размеров и имен файлов матриц
- Скрипт для запуска на BlueGene/P для рассчета времени в зависимости от количества процессов

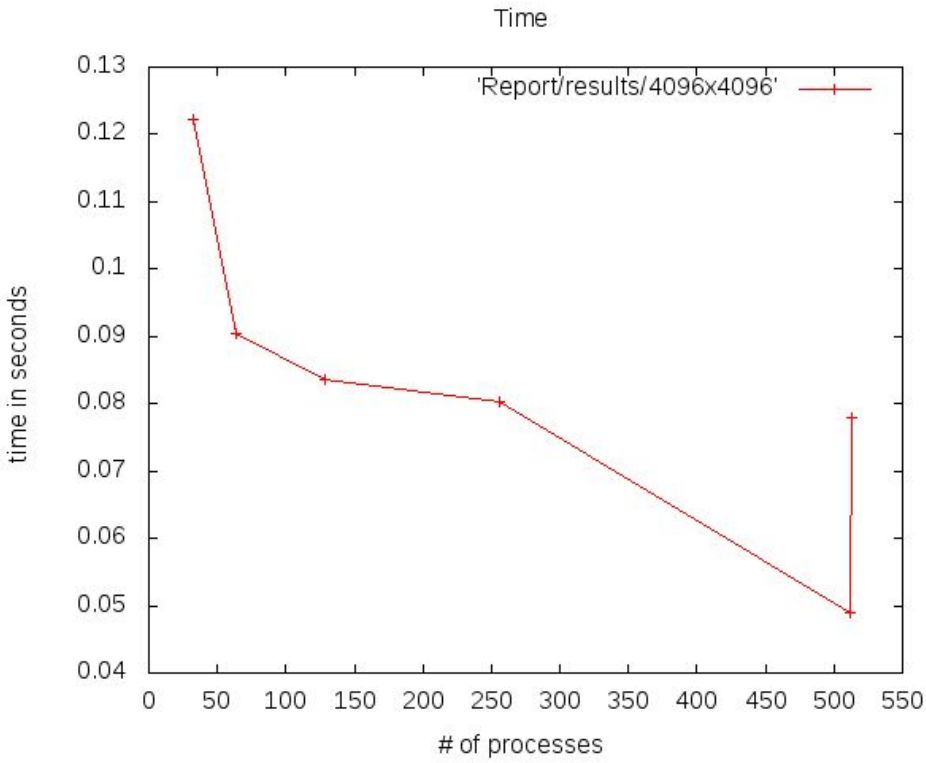
Результаты выполнения.

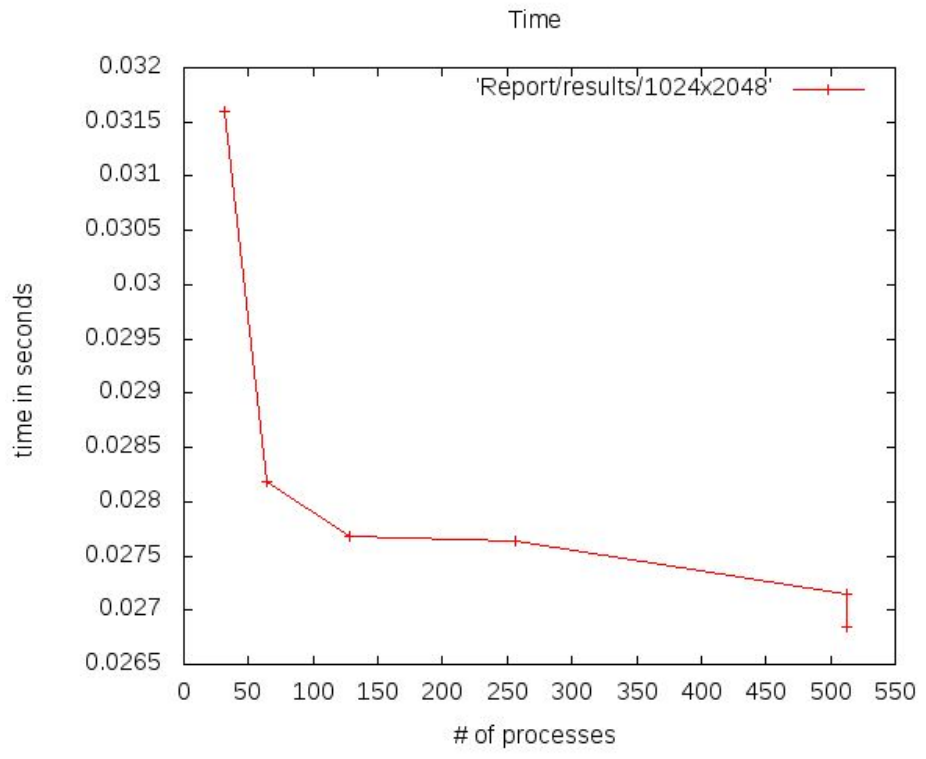
Проводилось перемножение матрицы на вектор размерами 512x512, 1024x1024, 2048x2048, 4096x4096, 1024x2048, 2048x512. Зависимость времени выполнения от количества процессов представлена на графике (время в секундах).


Time
'Report/results/512x512'
time in seconds
0.0028
0.0026
0.0024
0.0022
0.002
0.0018
0.0016
0.0014
0.0012
0.001
0
50
100
150
200
250
300
350
400
450
500
550
# of processes

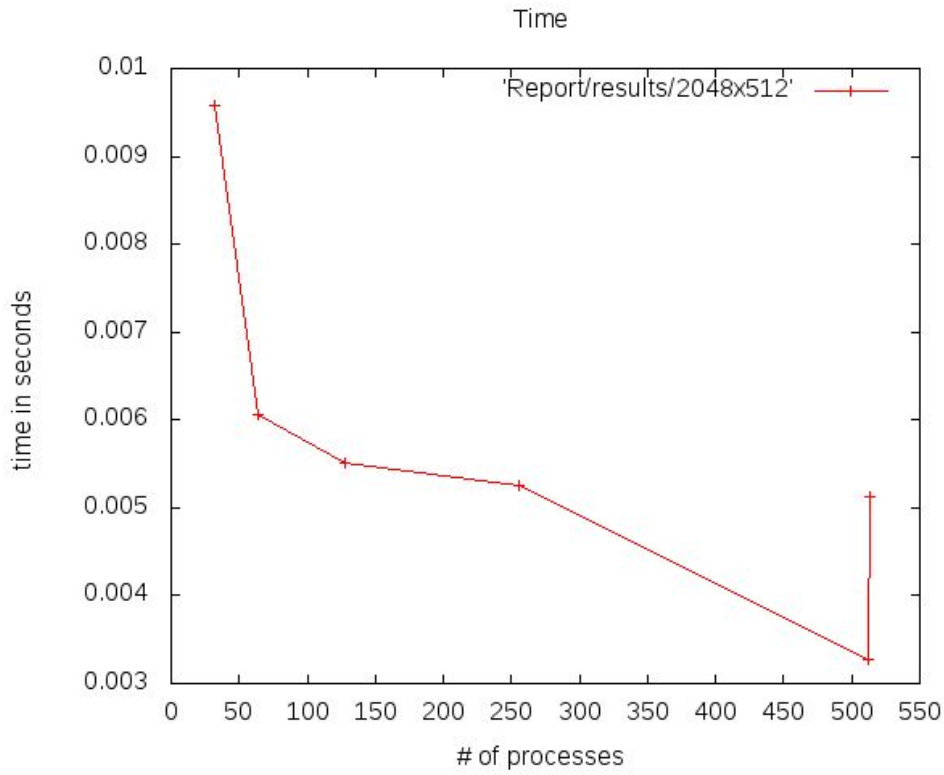

Time
'Report/results/1024x1024'
time in seconds
0.0085
0.008
0.0075
0.007
0.0065
0.006
0.0055
0.005
0.0045
0.004
0.0035
0.003
0
50
100
150
200
250
300
350
400
450
500
550
# of processes


Time
'Report/results/2048x2048'
time in seconds
# of processes
0.03
0.028
0.026
0.024
0.022
0.02
0.018
0.016
0.014
0.012
0
50
100
150
200
250
300
350
400
450
500
550


Time
'Report/results/4096x4096'
time in seconds
# of processes
0.13
0.12
0.11
0.1
0.09
0.08
0.07
0.06
0.05
0.04
0
50
100
150
200
250
300
350
400
450
500
550


Time
'Report/results/1024x2048'
time in seconds
# of processes
0.032
0.0315
0.031
0.0305
0.03
0.0295
0.029
0.0285
0.028
0.0275
0.027
0.0265
0
50
100
150
200
250
300
350
400
450
500
550

Москва 2016



Исходные данные:

Время в мс

m	n	1	32	64	128	256	512	маппинг
512	512	12.731	2.73	2.47	1.622	1.566	1.087	1.534
2048	512	50.798	9.577	6.052	5.515	5.26	3.269	5.13
1024	1024	50.958	8.022	6.3	5.525	5.263	3.265	5.1
1024	2048	104.418	31.6	28.186	27.685	27.64	27.155	26.851
2048	2048	205.3	29.72	22.994	20.955	19.931	12.11	19.315
4096	4096	837.062	122.32	90.48	83.485	80.377	49.035	77.851

Ускорение:

m	n	1	32	64	128	256	512	маппинг
512	512	1	4.66	5.15	7.85	8.13	11.71	8.30
2048	512	1	5.30	8.39	9.21	9.66	15.54	9.90
1024	1024	1	6.35	08.09	9.22	9.68	15.61	9.99
1024	2048	1	3.30	3.70	3.77	3.78	3.85	3.89
2048	2048	1	6.91	8.93	9.80	10.30	16.95	10.63
4096	4096	1	6.84	9.25	10.03	10.41	17.07	10.75

Эффективность:

m	n	1	32	64	128	256	512	маппинг
512	512	100.0%	14.6%	8.1%	6.1%	3.2%	2.3%	1.6%
2048	512	100.0%	16.6%	13.1%	7.2%	3.8%	3.0%	1.9%
1024	1024	100.0%	19.9%	12.6%	7.2%	3.8%	3.0%	2.0%
1024	2048	100.0%	10.3%	5.8%	2.9%	1.5%	0.8%	0.8%
2048	2048	100.0%	21.6%	14.0%	7.7%	4.0%	3.3%	2.1%
4096	4096	100.0%	21.4%	14.5%	7.8%	4.1%	3.3%	2.1%

Основные выводы.

Исследования показывают, что увеличение числа процессов в 512 раз позволяет добиться ускорения только лишь в 16 раз. Это дает эффективность распараллеливания на уровне 3%. При увеличении числа процессов эффективность убывает достаточно интенсивно на всей рассмотренной области изменений параметров запуска. Уменьшение эффективности на рассмотренной области работы параллельной программы связано с увеличением числа пересылок с ростом числа процессов и как следствие ростом накладных расходов на организацию вычислений. Основной вклад в значение эффективности вносит область с малым числом процессов, там эффективность достигает максимума в 21,6%. Далее эффективность очень резко падает до уровня около 2%. Задача имеет низкую вычислительную сложность, поэтому ее параллельная реализация показывает очень низкую эффективность.