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

Практикум

Отчёт

Решение СЛАУ методом отражений с параллельной реализацией

Работу выполнил
Косарев И.М.

Москва 2018

Постановка задачи и формат данных

Задача: Реализовать параллельное решение СЛАУ методом отражений

Компиляция и запуск:
mpicxx main.cpp -o householder

mpirun -np <n> householder <file>
mpirun -np <n> householder <matrix size>

Доступна генерация и считывание из файла.

Формат файла-матрицы:
Количество строк, столбцов и элементы матрицы построчно.
Столбцов строго столько же, сколько строк.
Система должна иметь единственное решение.

Описание алгоритма

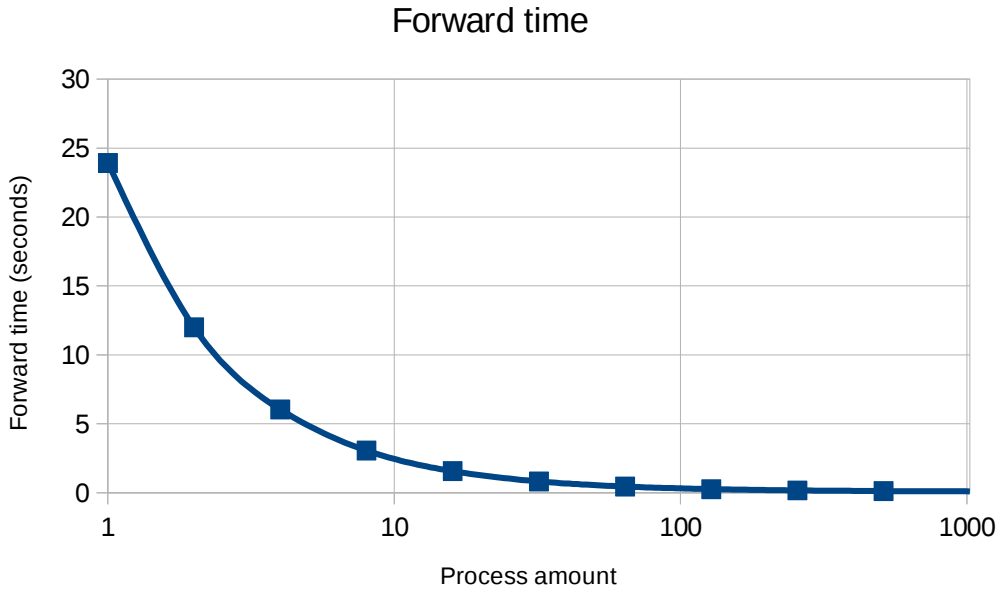
Идея метода заключается в разложении матрицы коэффициентов на произведение самосопряженной матрицы и верхнетреугольной.
В процессе преобразования матрица последовательно умножается на (n - 1) матриц, постепенно преобразующих её к верхнетреугольной засчёт становления столбцов коллинеарными к столбцам единичной матрицы. Это возможно, так как для любых двух векторов vecA и vecB существует vecW, такой что матрица U = E - 2 * vecW * vecW при умножении на vecA даст vecA', коллинеарный вектору vecB.
Мы последовательно строим матрицы преобразования для іго столбца нашей матрицы и іго столбца единичной матрицы
U = E — (vecA_i — vecE_i) * (vecA_i — vecE_i) / (1 + (vecA_i, vecE_i));
Получив верхнетреугольную матрицу, можем применять стандартный обратный ход.

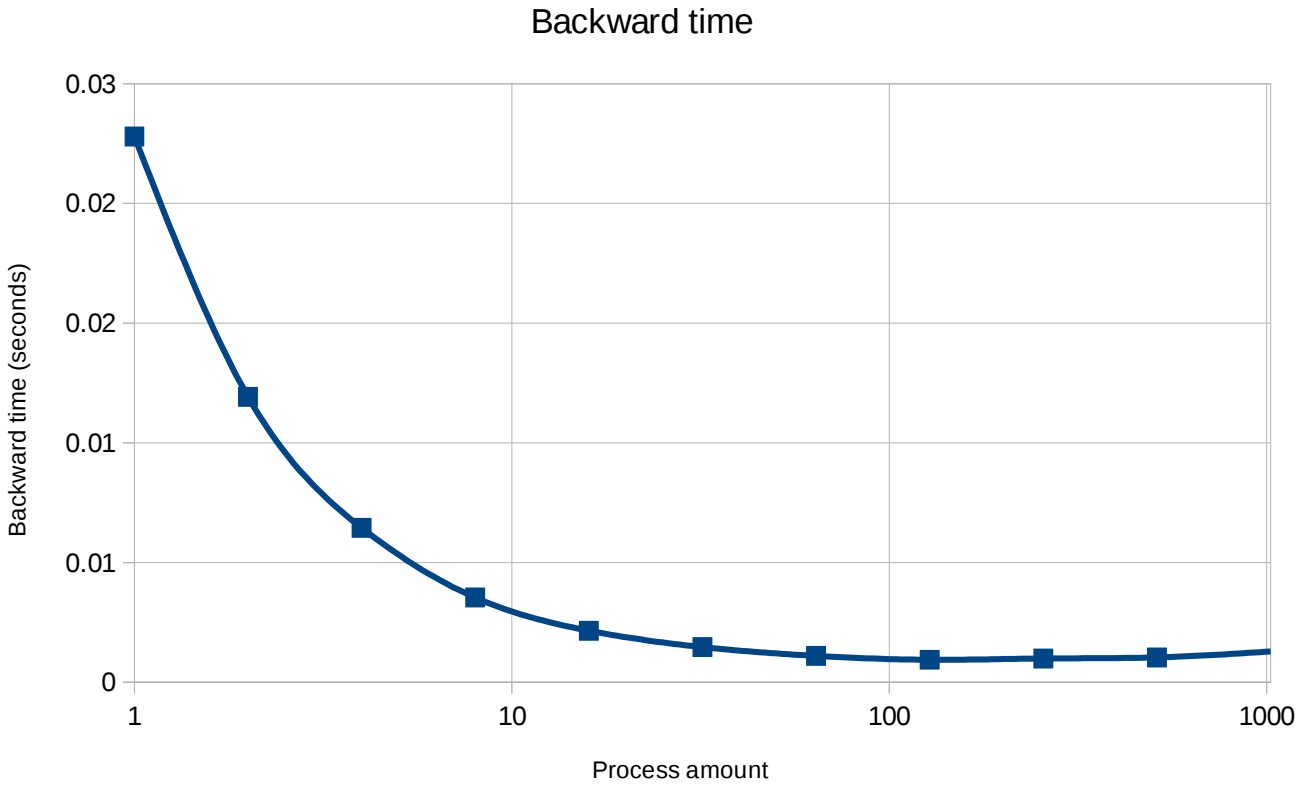
Для распараллеливания удобно делить матрицу по столбцам, чтобы после распространения преобразования одновременно обновлять столбцы в рамках своего процесса.
При прямом ходе в процессе итерации по столбцам тот процесс, на котором хранится текущий столбец, рассчитывает vecW и рассылает его, после чего параллельно производится изменение всех столбцов.
При обратном ходе итерация по столбцом происходит, соответственно, в обратном направлении. Предварительно всем процессам рассылается вектор значений правого столбца. Вначале вычисляется последний элемент вектора решения, в дальнейшем для вычисления соответствующего элемента решения процесс, содержащий соответствующий столбец, получает сумму произведений всех более правых элементов данной строки, умноженных на соответствующие элементы вектора решения, вычитает её из правого столбца и делит на соответствующий вычисляемому элементу решения элемент столбца.
Таким образом, в итоге мы получаем все элементы вектора решений.

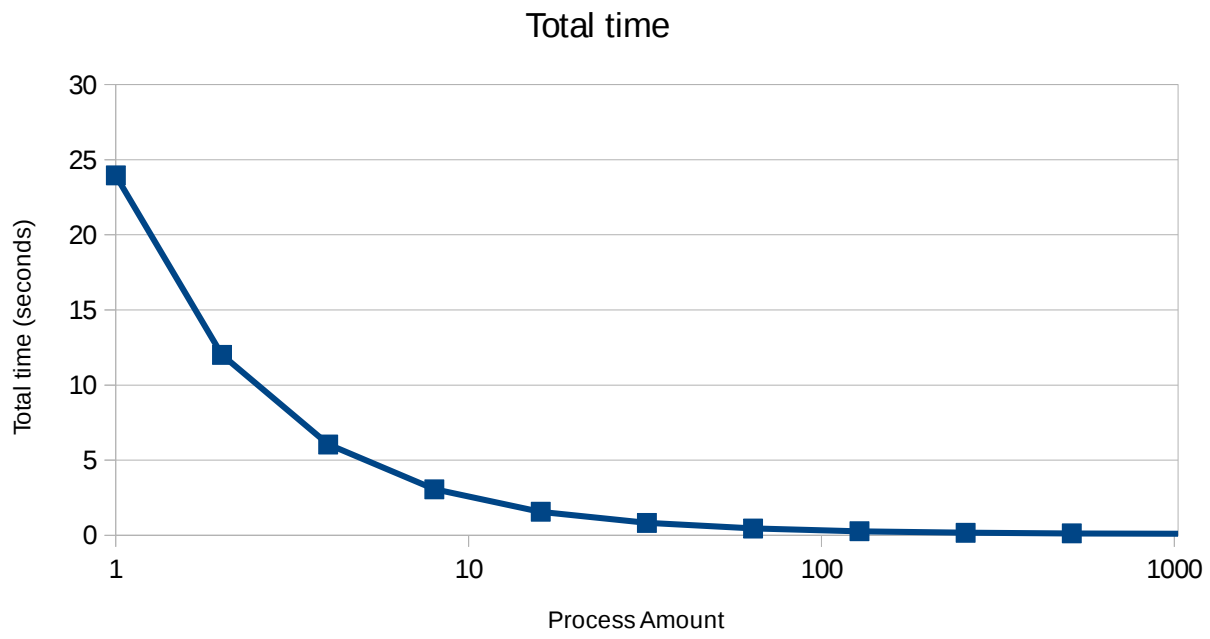
Результаты выполнения

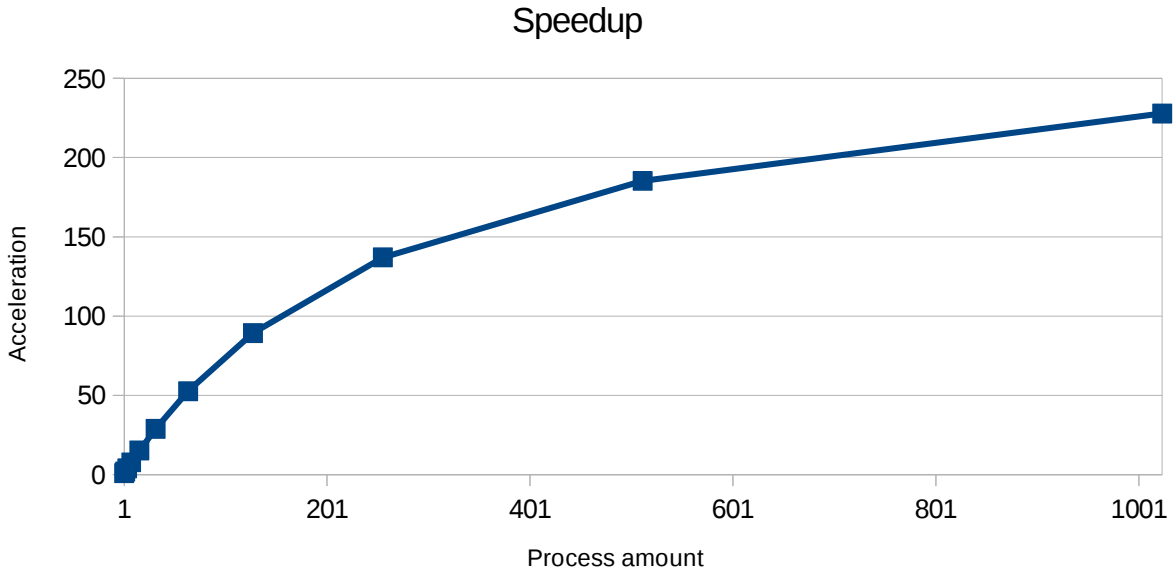
Размер матрицы 1000:

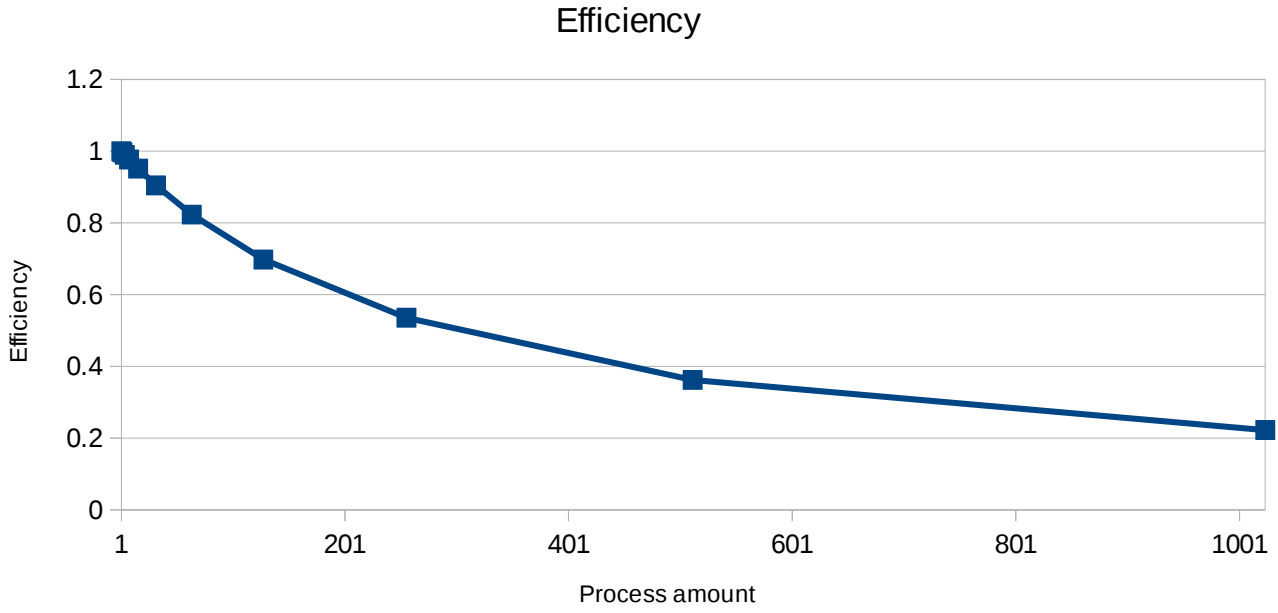
Process amount	Forward time	Backward time	Total time	Speedup	Efficiency
1	23.9220	0.0228	23.9448	1.0000	1.0000
2	11.9988	0.0119	12.0107	1.9936	0.9968
4	6.0431	0.0064	6.0496	3.9581	0.9895
8	3.0619	0.0035	3.0654	7.8112	0.9764
16	1.5709	0.0022	1.5730	15.2223	0.9514
32	0.8256	0.0015	0.8271	28.9506	0.9047
64	0.4535	0.0011	0.4546	52.6693	0.8230
128	0.2671	0.0009	0.2680	89.3496	0.6980
256	0.1738	0.0010	0.1748	137.0197	0.5352
512	0.1282	0.0010	0.1292	185.2846	0.3619
1024	0.1038	0.0013	0.1051	227.8023	0.2225





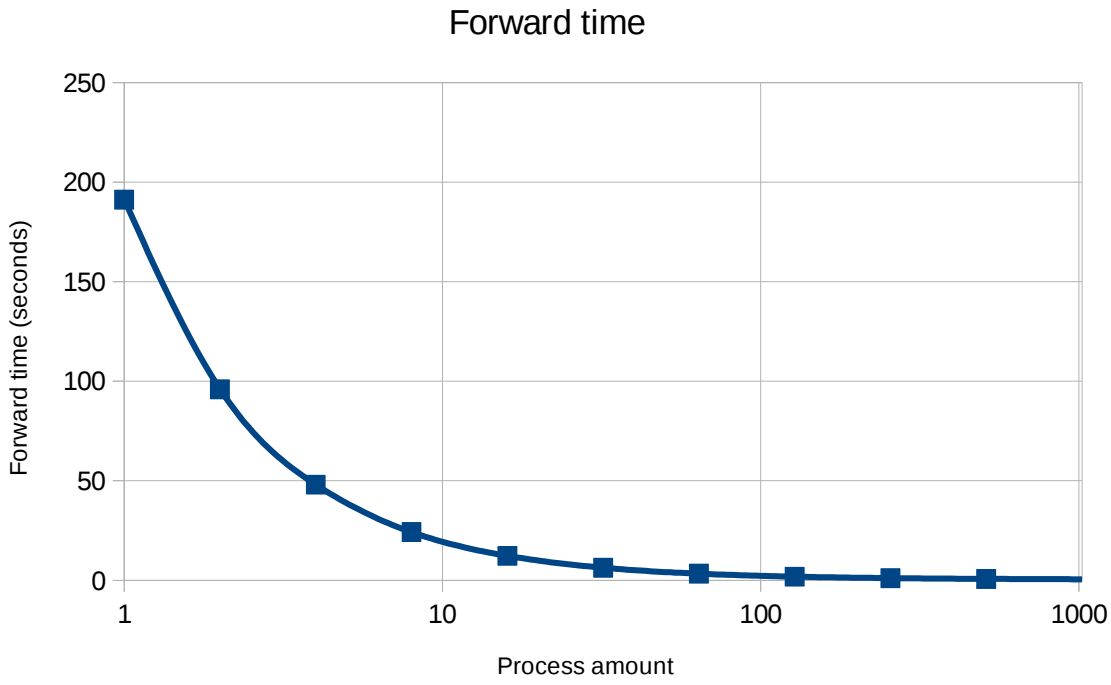

Total time
30
25
20
15
10
5
0
Total time (seconds)
1
10
100
1000
Process Amount

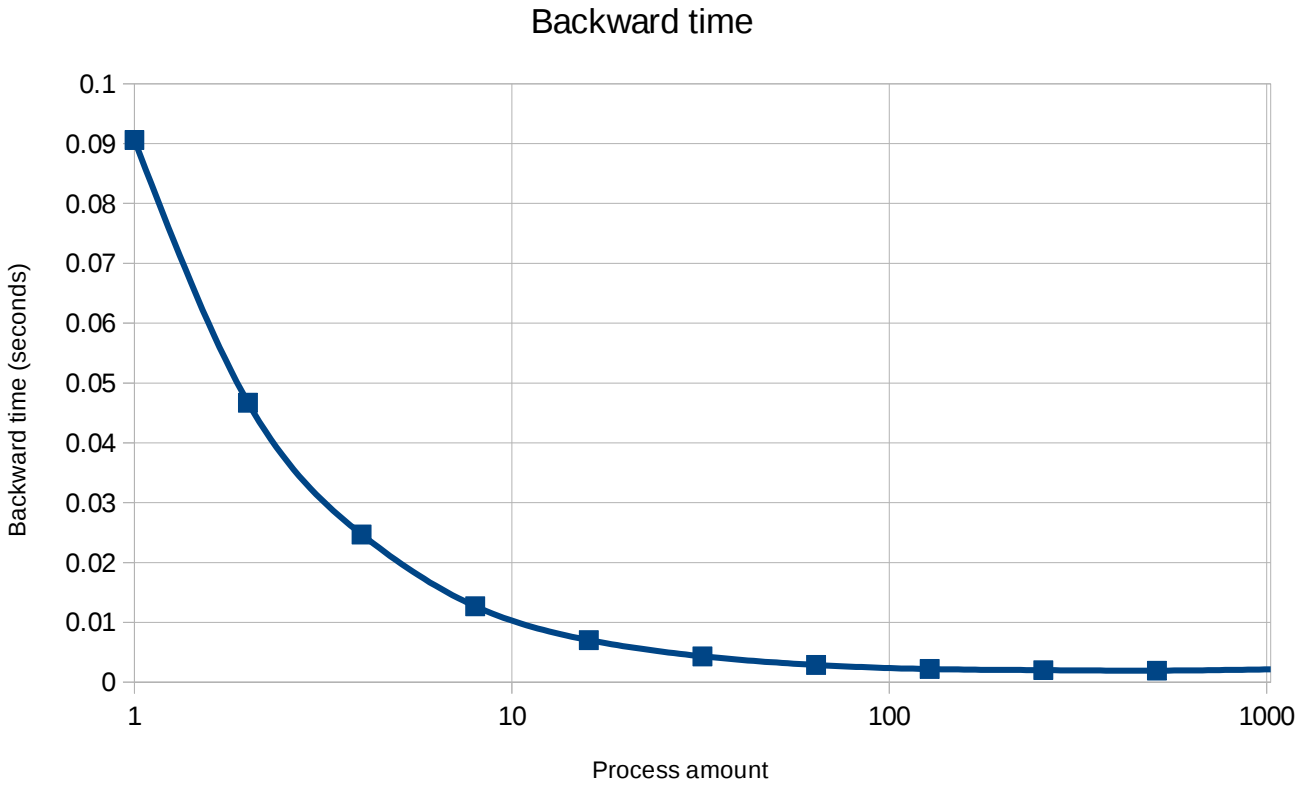

Speedup
250
200
150
100
50
0
Acceleration
1
201
401
601
801
1001
Process amount

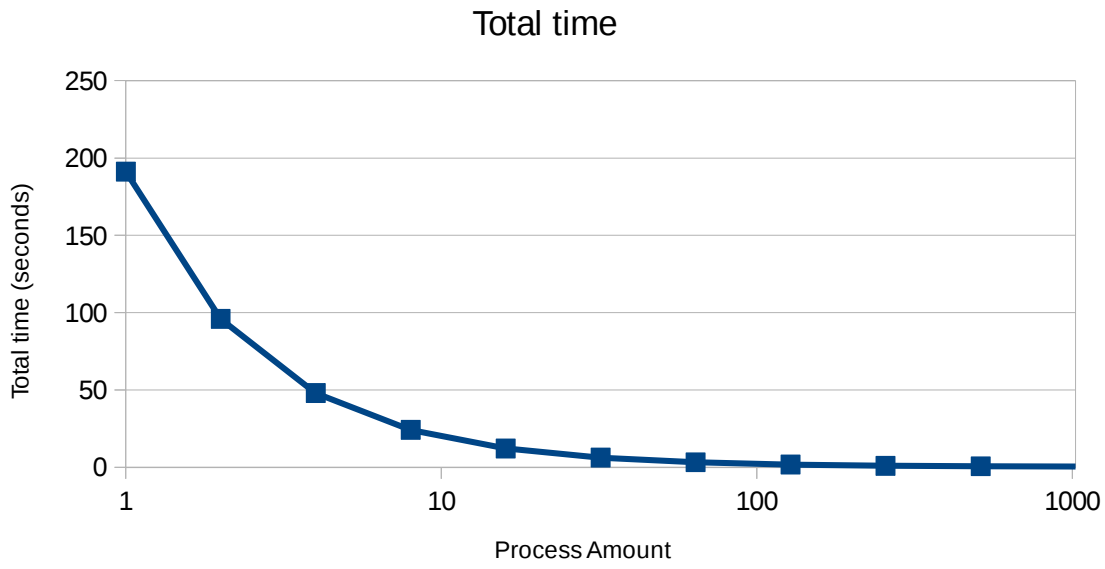

Efficiency
1.2
1
0.8
0.6
0.4
0.2
0
Efficiency
1
201
401
601
801
1001
Process amount

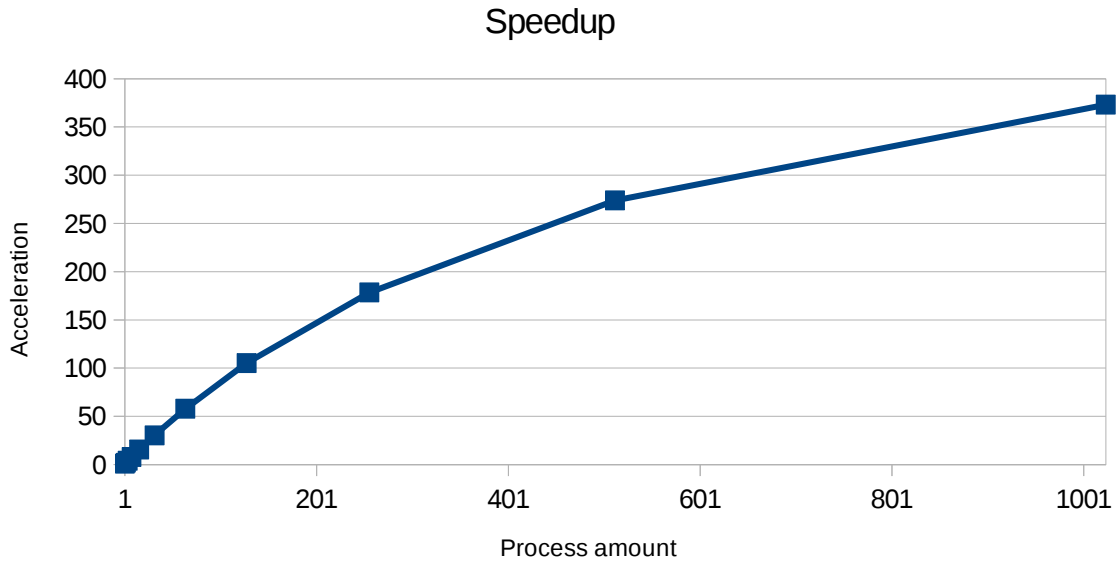
Размер матрицы 2000:

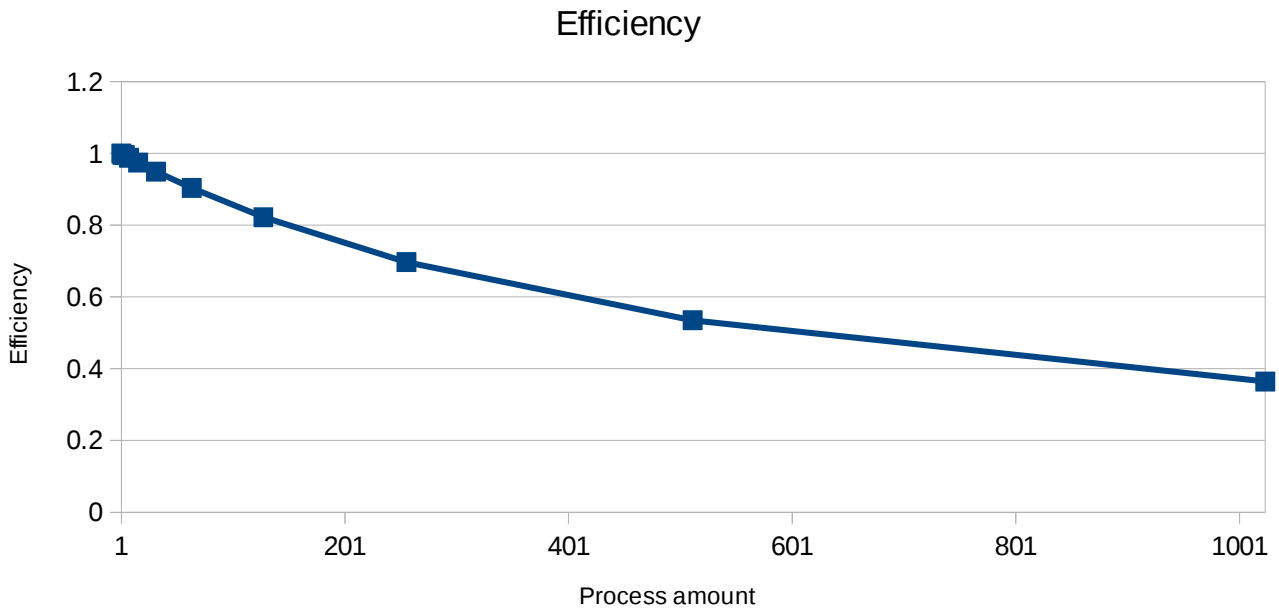
Process amount	Forward time	Backward time	Total time	Speedup	Efficiency
1	191.1280	0.0906	191.2186	1.0000	1.0000
2	95.9502	0.0467	95.9969	1.9919	0.9960
4	48.0171	0.0247	48.0418	3.9803	0.9951
8	24.1883	0.0127	24.2010	7.9013	0.9877
16	12.2560	0.0070	12.2630	15.5931	0.9746
32	6.2903	0.0043	6.2946	30.3781	0.9493
64	3.3033	0.0029	3.3062	57.8372	0.9037
128	1.8145	0.0022	1.8167	105.2561	0.8223
256	1.0694	0.0020	1.0714	178.4683	0.6971
512	0.6963	0.0019	0.6983	273.8521	0.5349
1024	0.5103	0.0022	0.5125	373.1323	0.3644





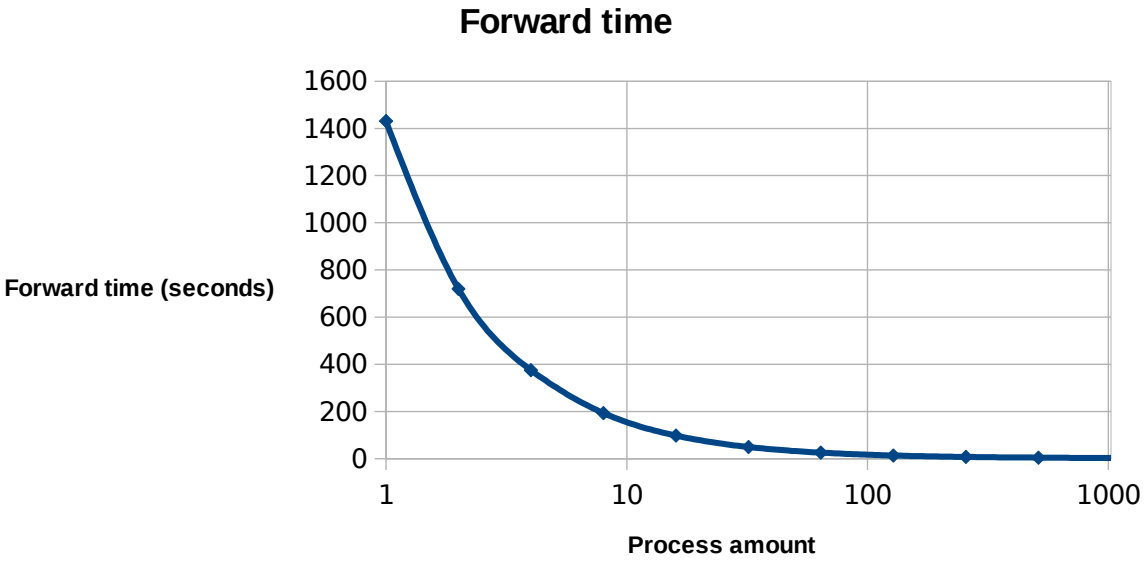

Total time
250
200
150
100
50
0
Total time (seconds)
1
10
100
1000
Process Amount

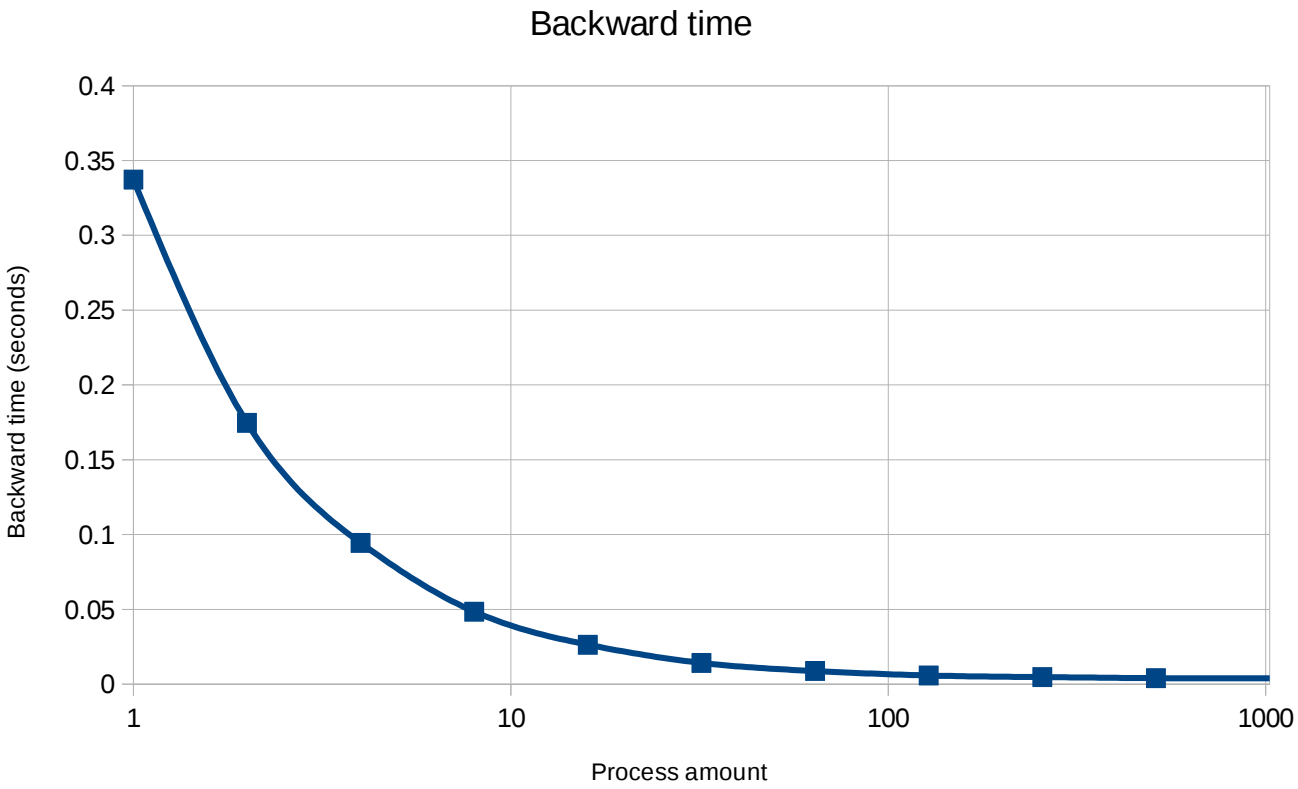

Speedup
400
350
300
250
200
150
100
50
0
Acceleration
1
201
401
601
801
1001
Process amount

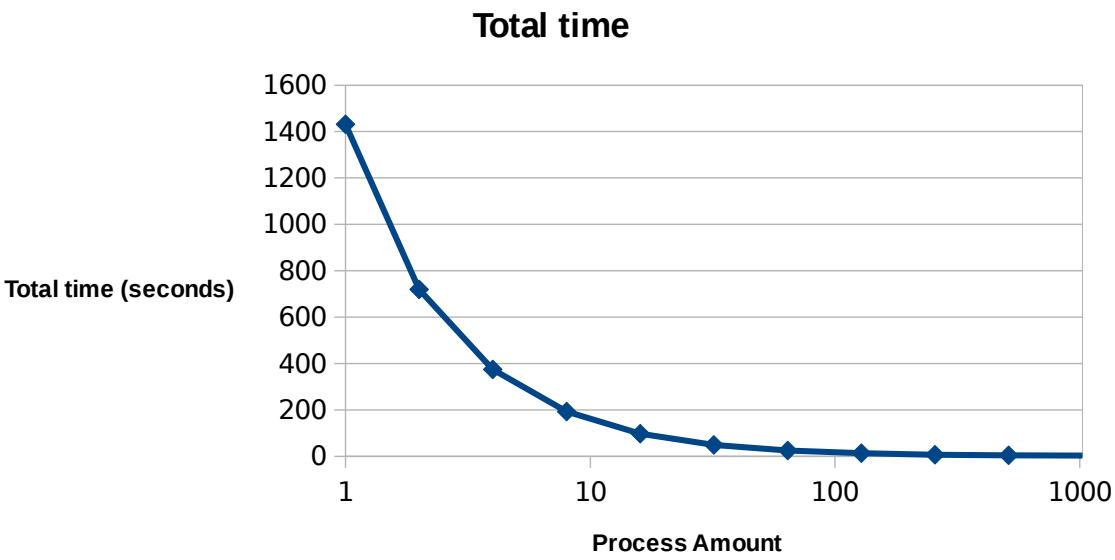

Efficiency
1.2
1
0.8
0.6
0.4
0.2
0
Efficiency
1
201
401
601
801
1001
Process amount

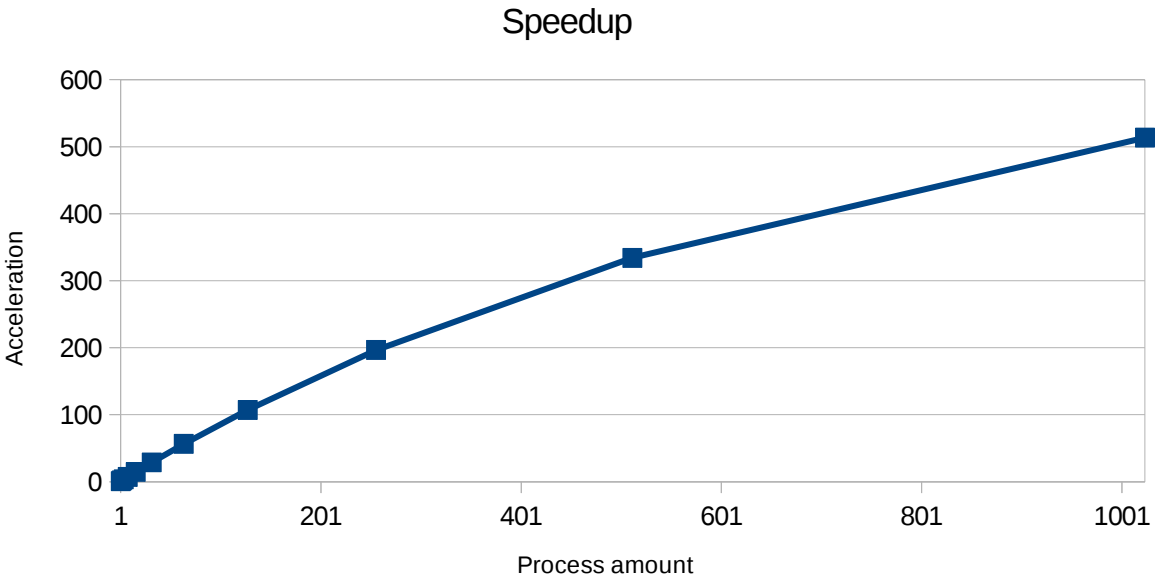
Размер матрицы 4000:

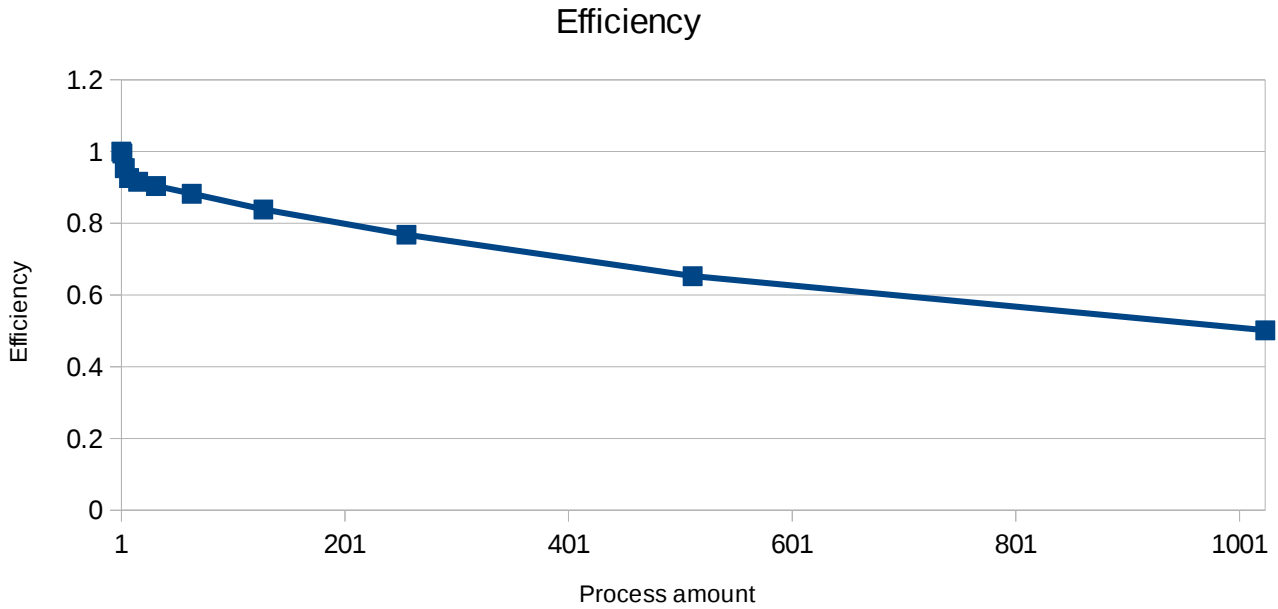
Process amount	Forward time	Backward time	Total time	Speedup	Efficiency
1	1431.0384	0.3372	1431.3756	1.0000	1.0000
2	719.4763	0.1746	719.6509	1.9890	0.9945
4	375.1180	0.0944	375.2124	3.8148	0.9537
8	193.1520	0.0485	193.2005	7.4088	0.9261
16	97.6830	0.0264	97.7094	14.6493	0.9156
32	49.4782	0.0143	49.4925	28.9211	0.9038
64	25.3368	0.0088	25.3456	56.4743	0.8824
128	13.3315	0.0058	13.3373	107.3211	0.8384
256	7.2771	0.0048	7.2819	196.5666	0.7678
512	4.2806	0.0040	4.2846	334.0715	0.6525
1024	2.7835	0.0040	2.7875	513.4989	0.5015






Total time
Total time (seconds)
1600
1400
1200
1000
800
600
400
200
0
1
10
100
1000
Process Amount


Speedup
Acceleration
600
500
400
300
200
100
0
1
201
401
601
801
1001
Process amount


Efficiency
Efficiency
1.2
1
0.8
0.6
0.4
0.2
0
1
201
401
601
801
1001
Process amount

Выводы

Распараллеливание задачи поиска решения СЛАУ методом отражений достаточно имеет заметную эффективность.

Также можно заметить, что эффективность, особенно на большом числе процессоров, повышается с увеличением размера матриц, причём с коэффициентом около 1,5.