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

Курс: практикум на ЭВМ.

Отчет №1.

Реализация однокубитного квантового преобразования с использованием OpenMP

Работу выполнила

Килина М. Л.

Москва, 2020

Постановка задачи и формат данных.

Задача. Реализовать параллельную программу на C++ с использованием OpenMP, которая выполняет однокубитное квантовое преобразование над вектором состояний длины 2^n, где n – количество кубитов, по указанному номеру кубита k.

Параметры, передаваемые командной строке:

- Количество выделяемых нитей программы
- Количество кубитов n
- Номер кубита k
- Имя файла, в который записывается время работы программы

Описание алгоритма.

Имеется комплексный входной вектор (массив) размерности 2^n : $\{a_i\} = \{a_0, a_1, \dots, a_{2^n-1}\}$; n – параметр задачи (число кубитов). Над такими векторами нам необходимо производить так называемые однокубитные операции. Обе эти операции переводят вектор в новый вектор такой же размерности (длины массива). Однокубитная операция задается двумя параметрами: комплексной матрицей размера 2x2 и числом от 1 до n (данный параметр обозначает номер кубита, по которому проводится операция). Итак, дана комплексная матрица:

$$U = \begin{pmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{pmatrix}$$

и k - номер индекса от 1 до n (номер кубита). Такая операция преобразует вектор $\{a_{i_1,i_2,\dots i_n}\}$ в $\{b_{i_1,i_2,\dots i_n}\}$, где все 2 n элементов нового вектора вычисляются по следующей формуле:

$$b_{i_1 i_2 \dots i_k \dots i_n} = \sum_{j_k=0}^{1} u_{i_k j_k} a_{i_1 i_2 \dots j_k \dots i_n} = u_{i_k 0} a_{i_1 i_2 \dots 0_k \dots i_n} + u_{i_k 1} a_{i_1 i_2 \dots 1_k \dots i_n}$$

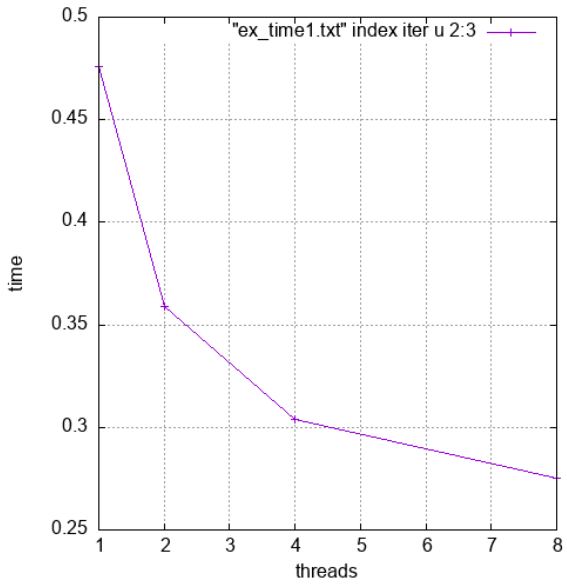
В программе используется преобразование Адамара.

Результаты выполнения.

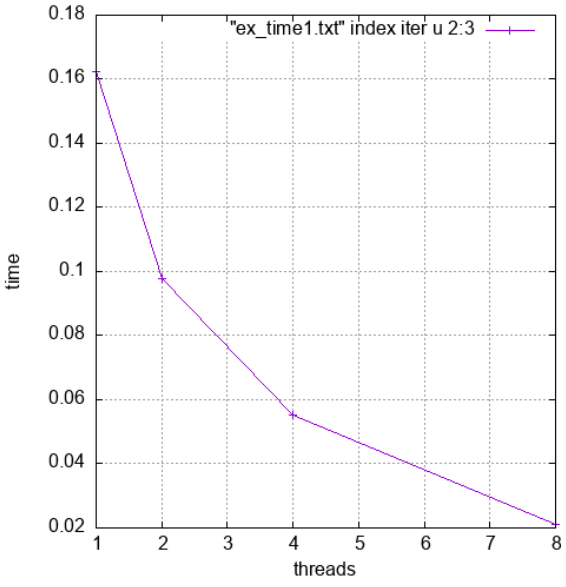
1) Резльтат для k = 1

Количество кубитов	Количество процессов	Полное время работы (сек)	Время выполнения преобразования (сек)	Ускорение (для полного времени)	Ускорение (для выполнения преобразования)
20	1	0,475981	0,162559	1	1
20	2	0,359147	0,0979161	1,32530969	1,66018662
20	4	0,30417	0,0551773	1,56485189	2,946120959
20	8	0,275047	0,0211252	1,73054423	7,695027739
24	1	7,95436	2,61853	1	1
24	2	6,06994	1,31886	1,312614	1,985449555
24	4	5,06994	0,795344	1,570118	3,292323825
24	8	4,60002	0,502652	1,729201	5,209429188
28	1	128,067	41,8127	1	1
28	2	95,7812	21,3569	1,337079	1,957807547
28	4	81,1047	12,4984	1,579033	3,345444217
28	8	72,7552	7,37942	1,760245	5,666122812
30	1	501,349	170,016	1	1
30	2	374,087	86,4554	1,340194	1,96651684
30	4	317,79	51,5037	1,577611	3,301044391
30	8	285,866	27,4862	1,75379	6,185503998

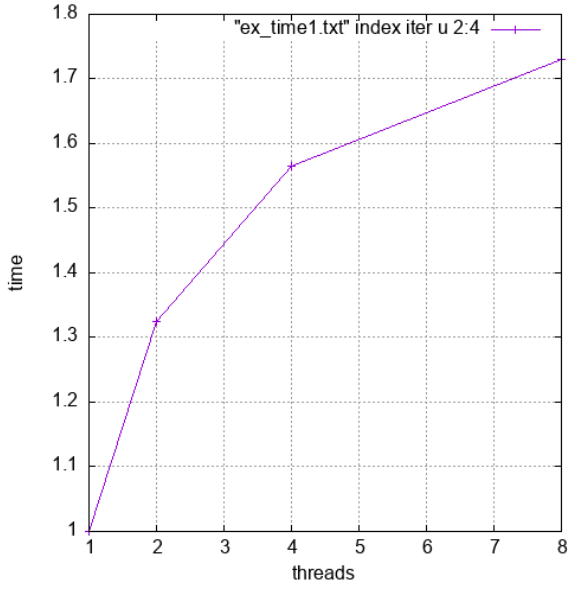
n = 20



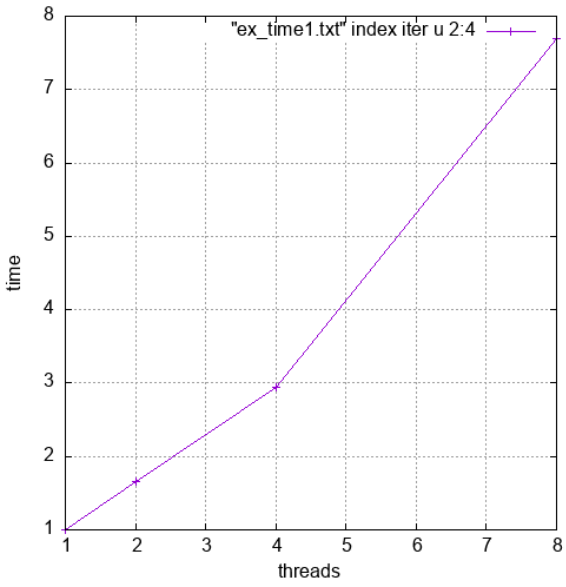
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей

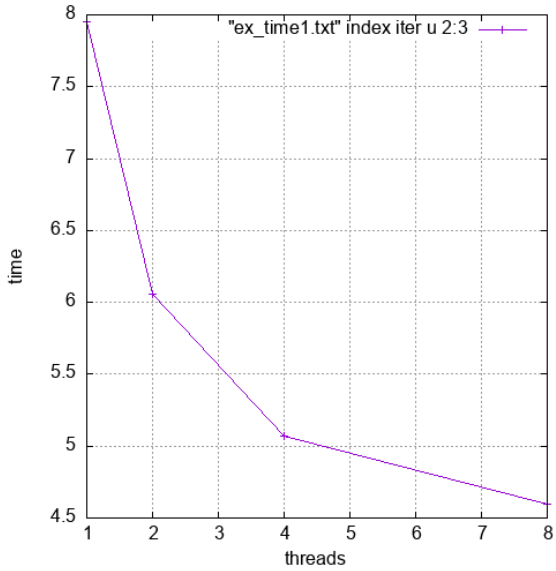


Ускорение (для полного выполнения программы)

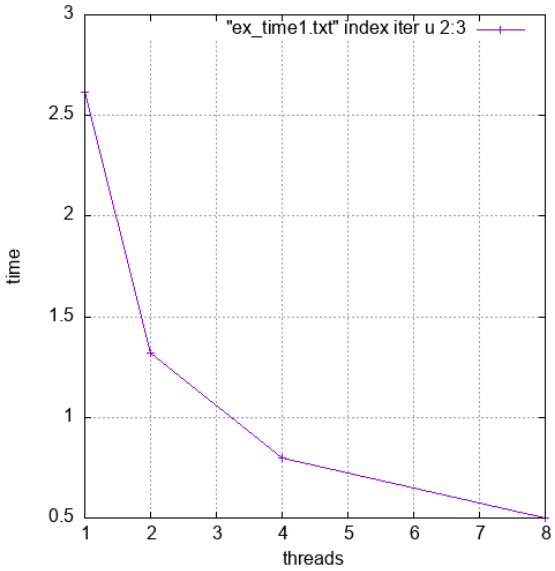


Ускорение (для выполнения преобразования)

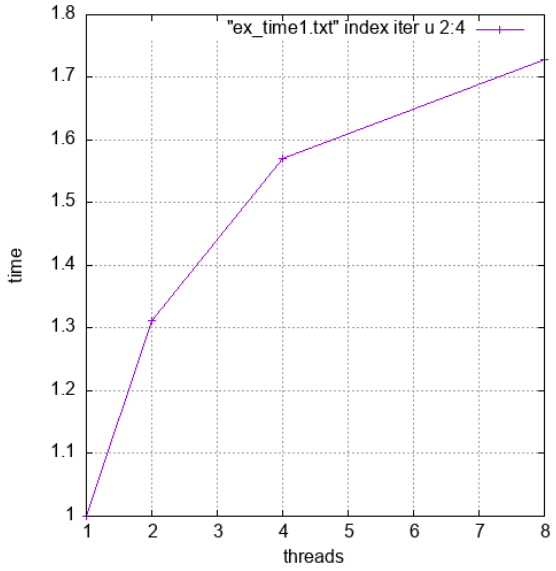
n = 24



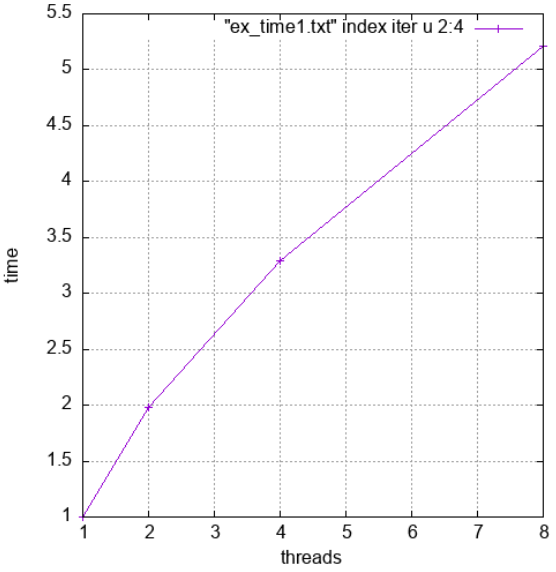
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей

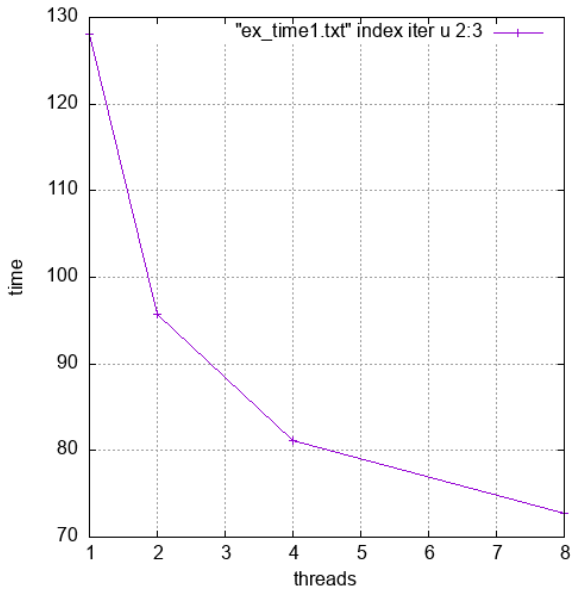


Ускорение (для полного выполнения программы)

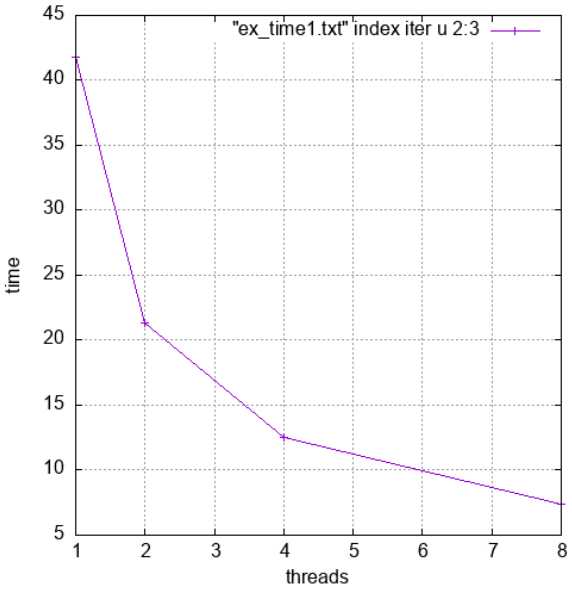


Ускорение (для выполнения преобразования)

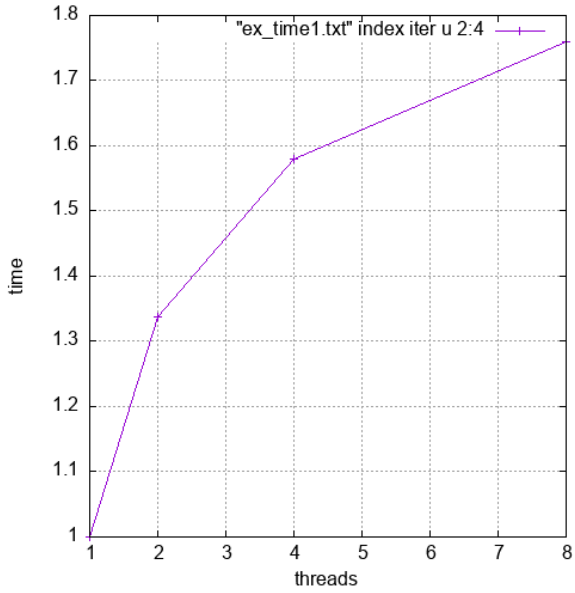
n = 28



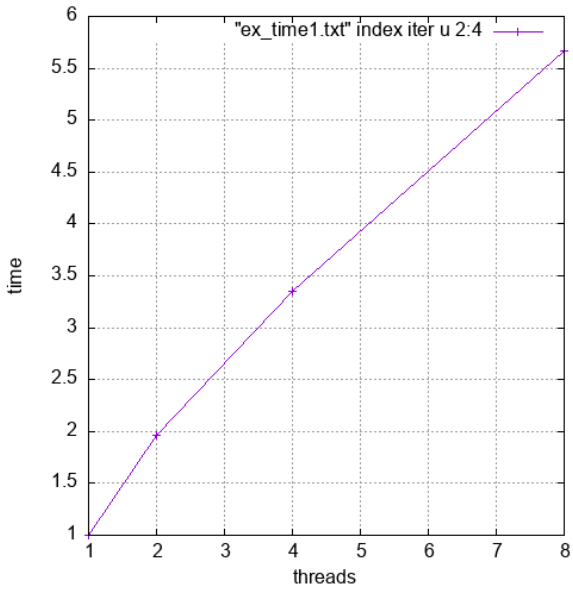
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей

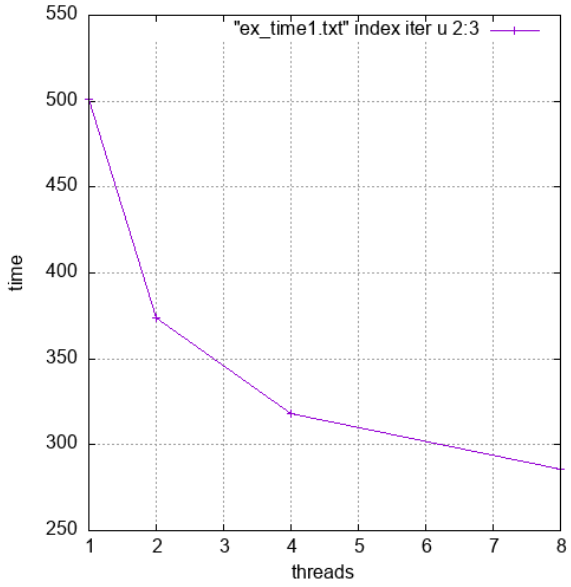


Ускорение (для полного выполнения программы)

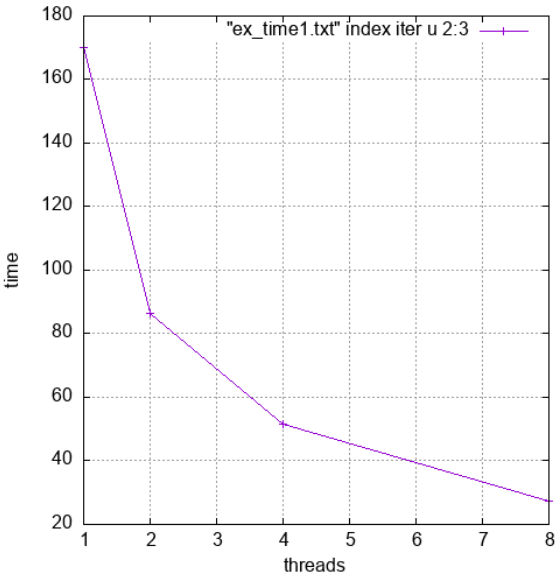


Ускорение (для выполнения преобразования)

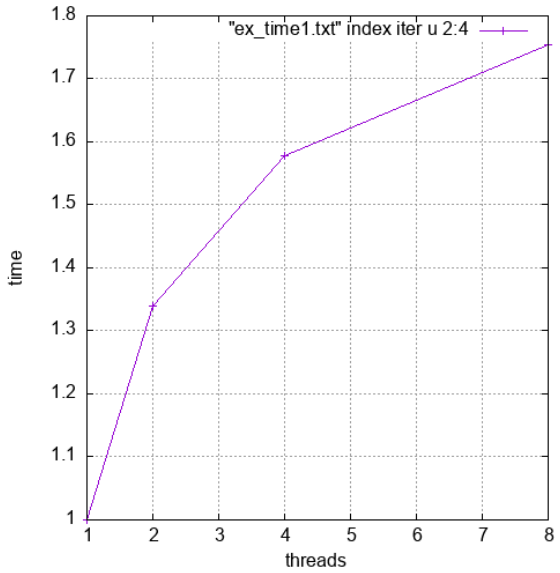
n = 30



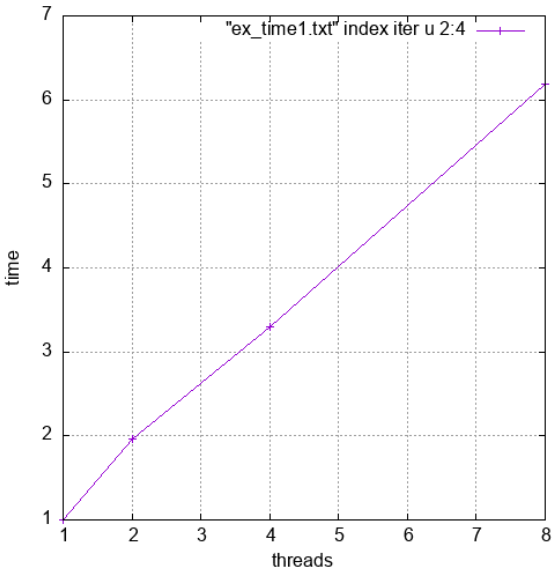
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей



Ускорение (для полного выполнения программы)

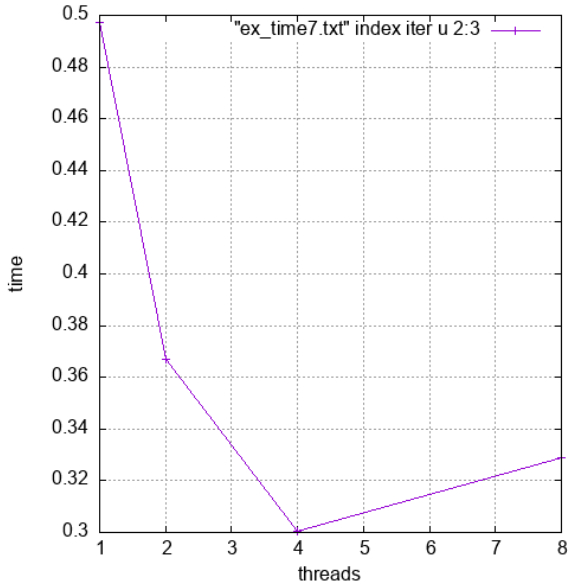


Ускорение (для выполнения преобразования)

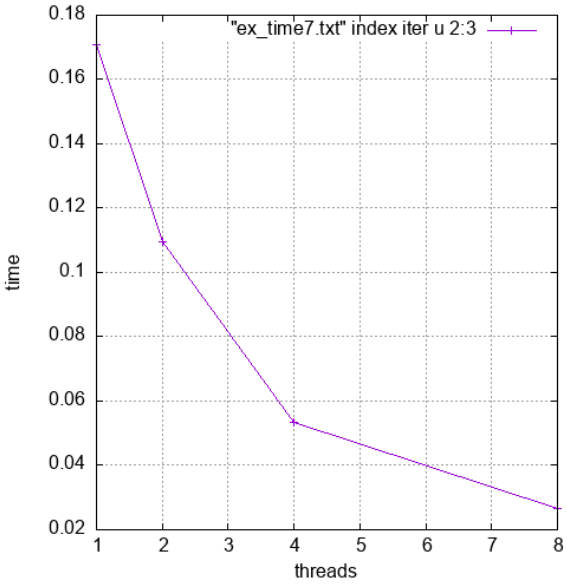
2) k = 7

Количество кубитов	**Количество процессов**	**Полное время работы (сек)**	**Время выполнения преобразования (сек)**	**Ускорение (для полного времени)**	**Ускорение (для выполнения преобразования)**
20	1	0, 49747	0,17096	1	1
20	2	0,366909	0,109386	1,35584	1,562906
20	4	0,300463	0,053405	1,655678	3,201204
20	8	0,329041	0,026453	1,511878	6,462881
24	1	7,99336	2,64922	1	1
24	2	5,93673	1,33647	1,346425	1,982252
24	4	4,87726	0,757791	1,639804	3,495977
24	8	4,41824	0,41404	1,809173	6,398464
28	1	125,584	41,8819	1	1
28	2	93,3483	10,6335	1,345327	3,938675
28	4	78,1869	21,0088	1,606203	1,993541
28	8	70,2742	9,43548	1,787057	4,438767
30	1	501,552	170,48	1	1
30	2	376,322	90,8316	1,332774	1,87688
30	4	312,286	51,7253	1,606066	3,295873
30	8	280,657	28,1575	1,787064	6,054515

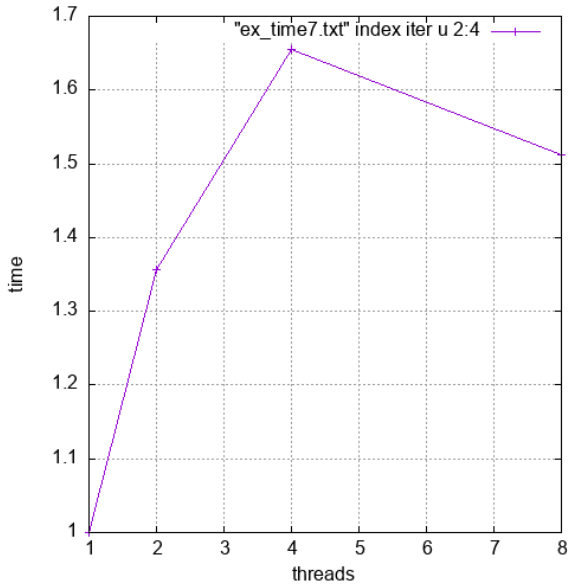
n = 20



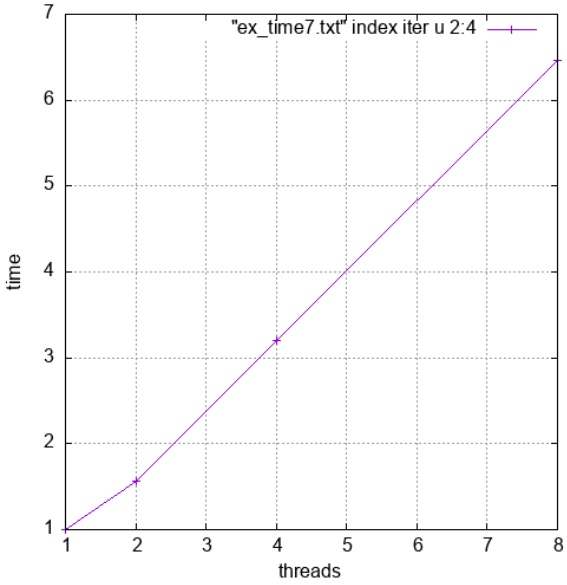
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей

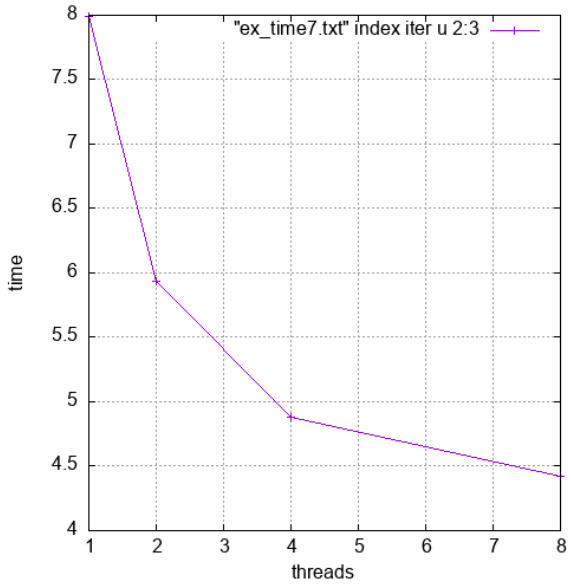


Ускорение (для полного выполнения программы)

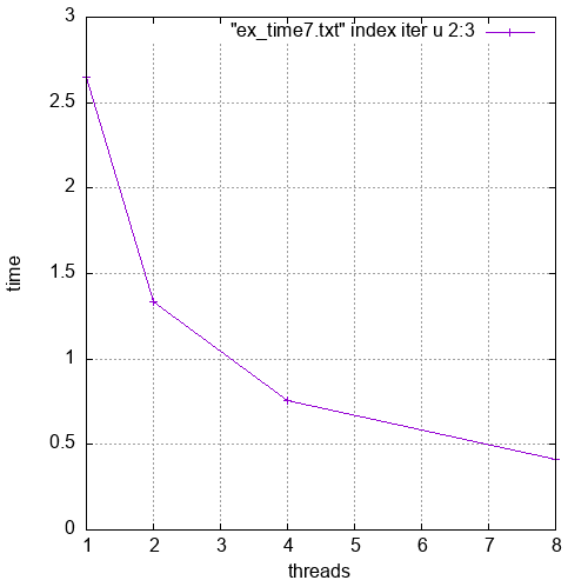


Ускорение (для выполнения преобразования)

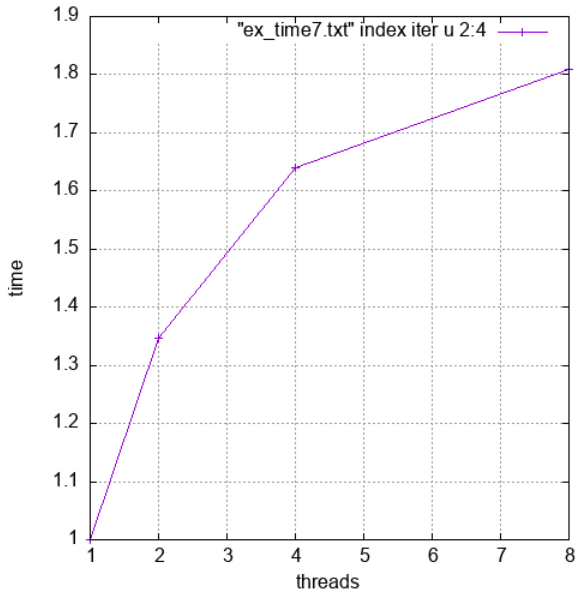
n = 24



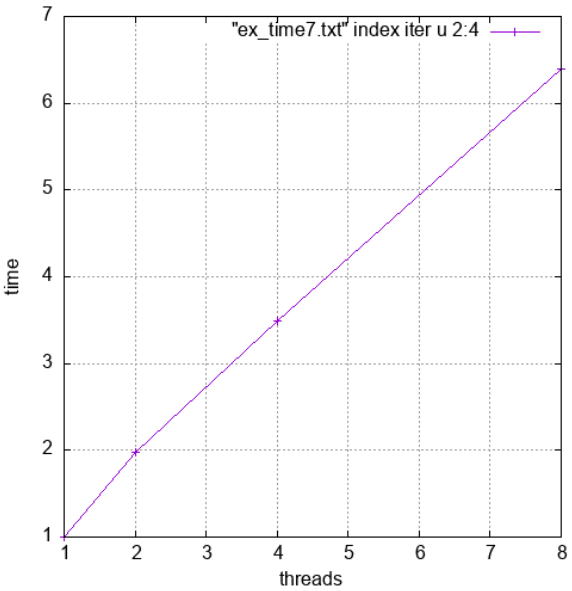
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей

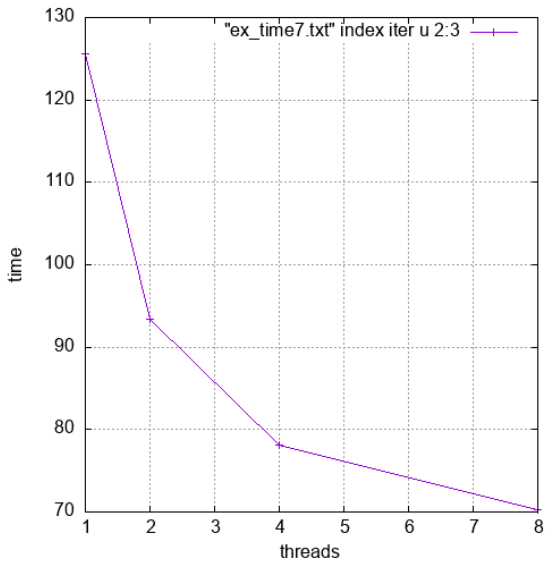


Ускорение (для полного выполнения программы)

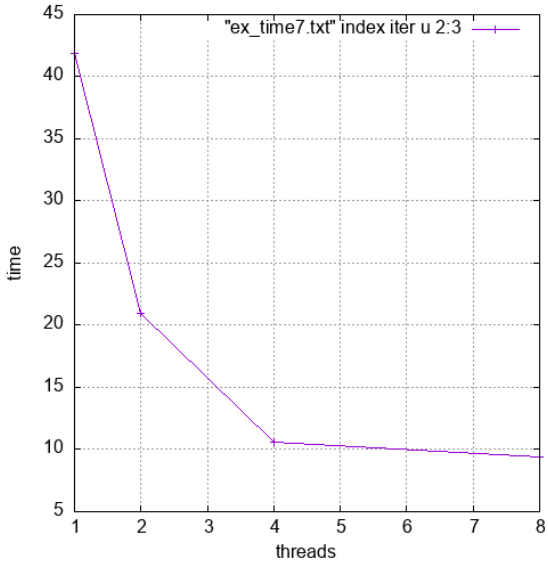


Ускорение (для выполнения преобразования)

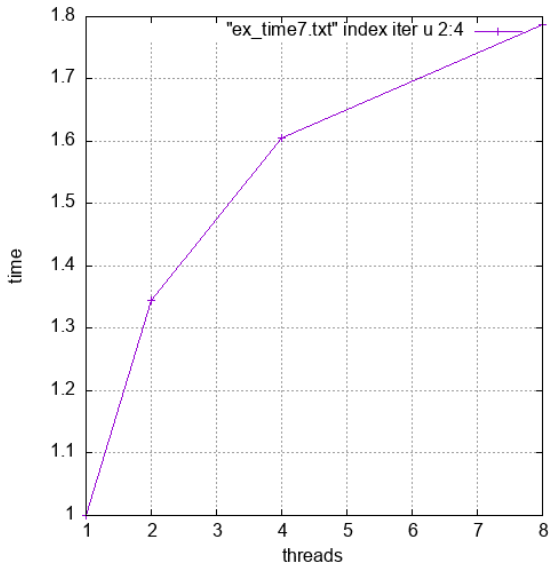
n = 28



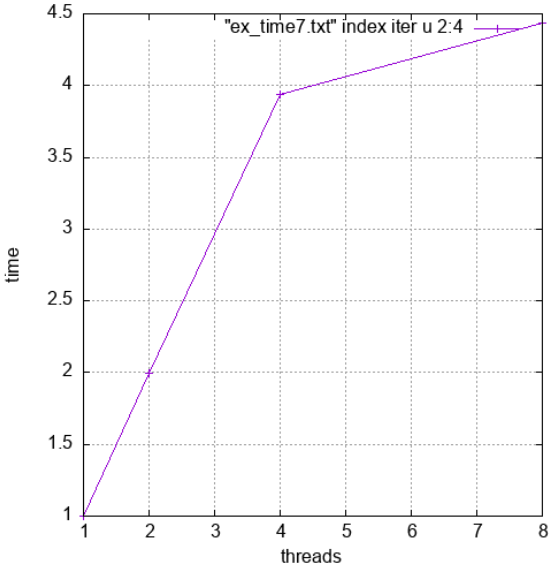
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей

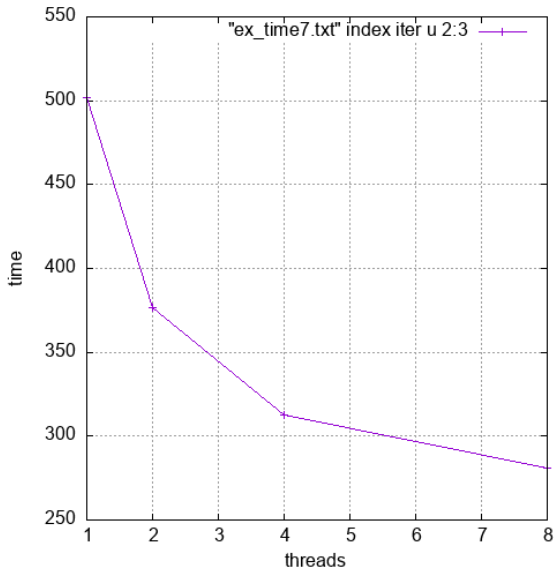


Ускорение (для полного выполнения программы)

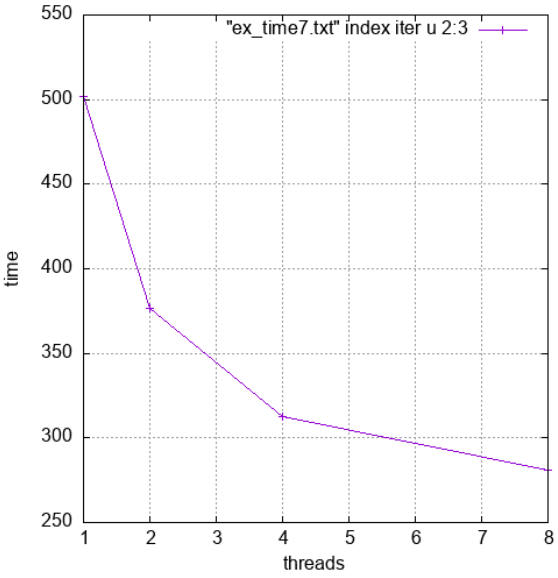


Ускорение (для выполнения преобразования)

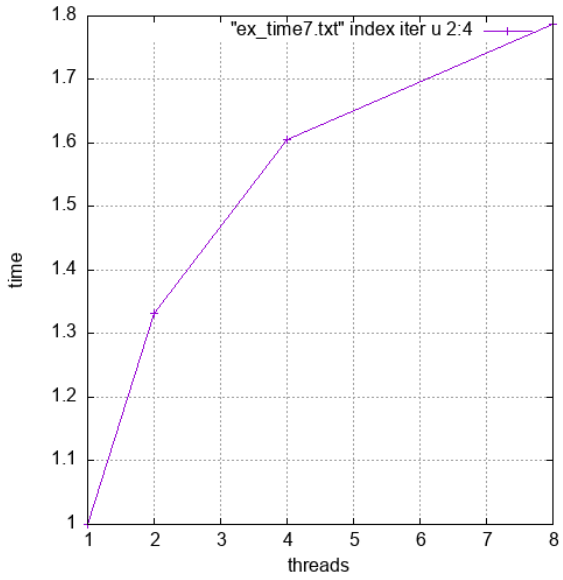
n = 30



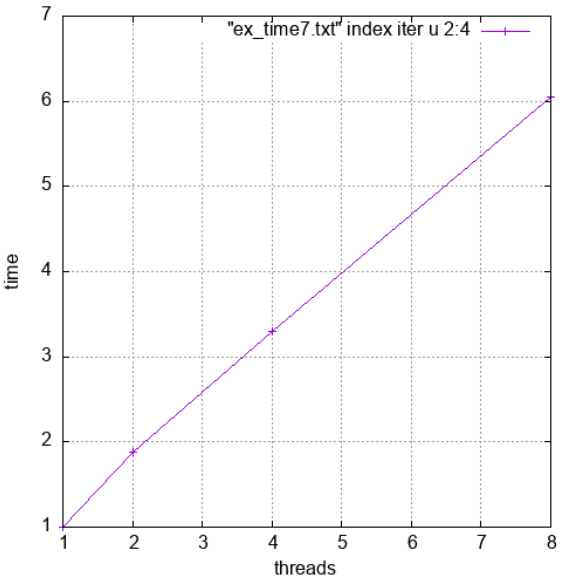
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей



Ускорение (для полного выполнения программы)

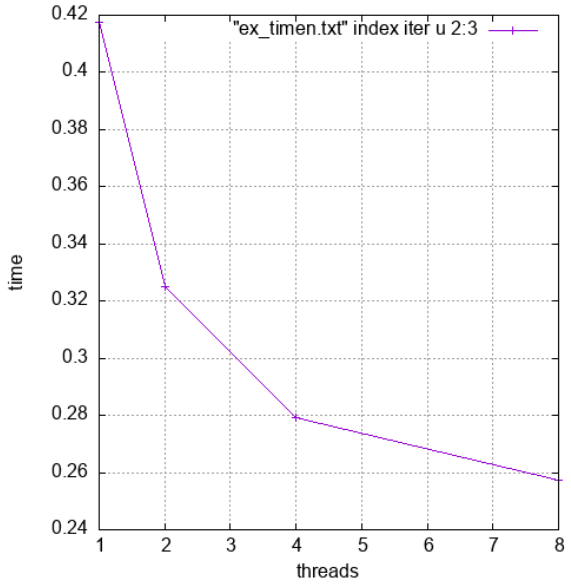


Ускорение (для выполнения преобразования)

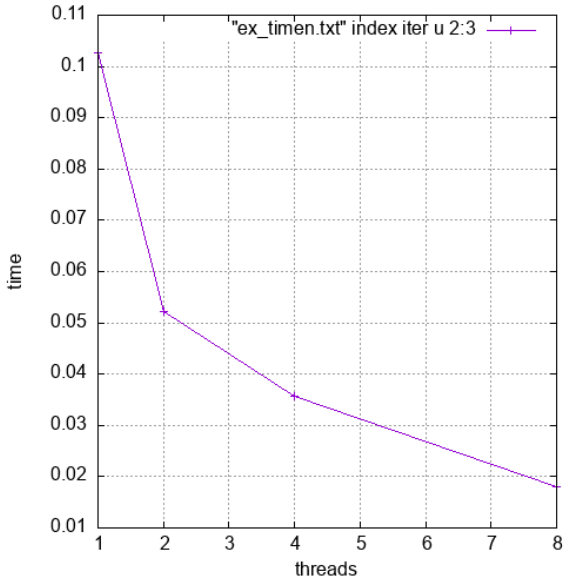
3) k = n

Количество кубитов	**Количество процессов**	**Полное время работы (сек)**	**Время выполнения преобразования (сек)**	**Ускорение (для полного времени)**	**Ускорение (для выполнения преобразования)**
20	1	0,417485	0,102719	1	1
20	2	0,325147	0,0522	1,283988	1,967804
20	4	0,279092	0,03567	1,495869	2,879711
20	8	0,257516	0,018087	1,6212	5,679225
24	1	6,846	1,61945	1	1
24	2	5,45611	0,812562	1,25474	1,993017
24	4	4,64035	0,418145	1,47532	3,872939
24	8	4,23184	0,277617	1,617736	5,833396
28	1	110,189	26,119	1	1
28	2	86,2942	13,0205	1,276899	2,005991
28	4	73,5132	6,69577	1,498901	3,900821
28	8	68,5235	4,02771	1,608047	6,484826
30	1	440,56	104,477	1	1
30	2	344,609	54,3915	1,278434	1,920833
30	4	296,505	31,5026	1,485843	3,316456
30	8	273,242	19,502	1,612344	5,357245

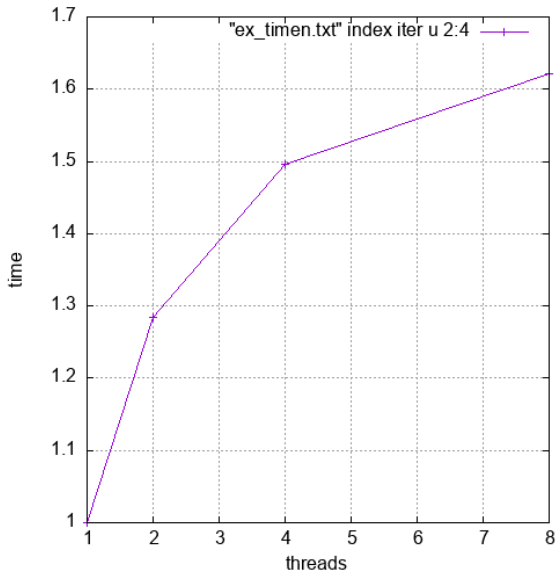
n = 20



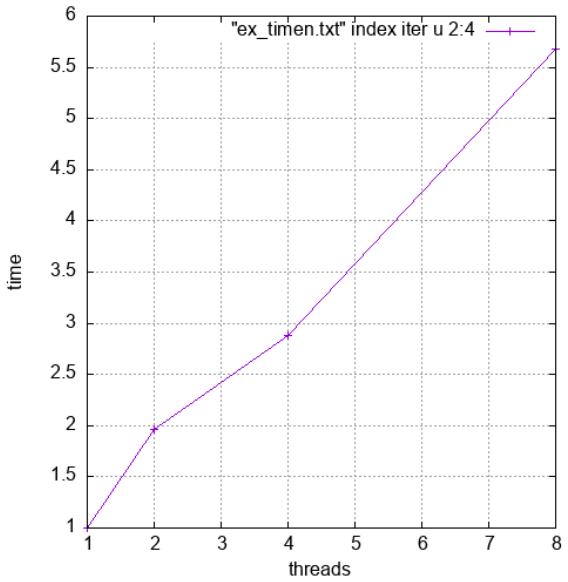
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей

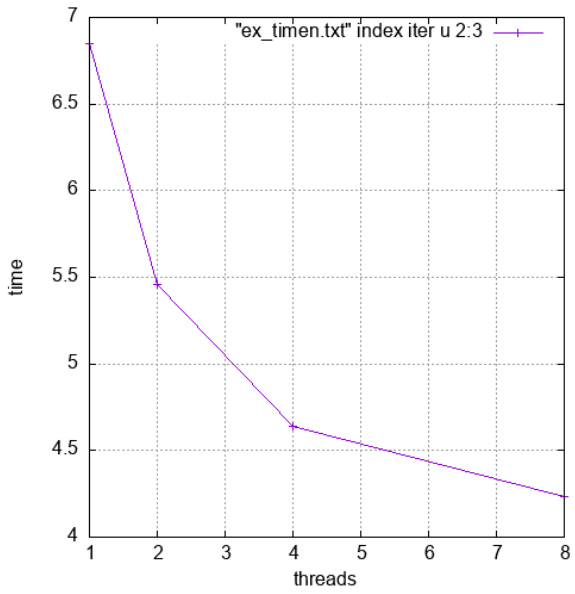


Ускорение (для полного выполнения программы)

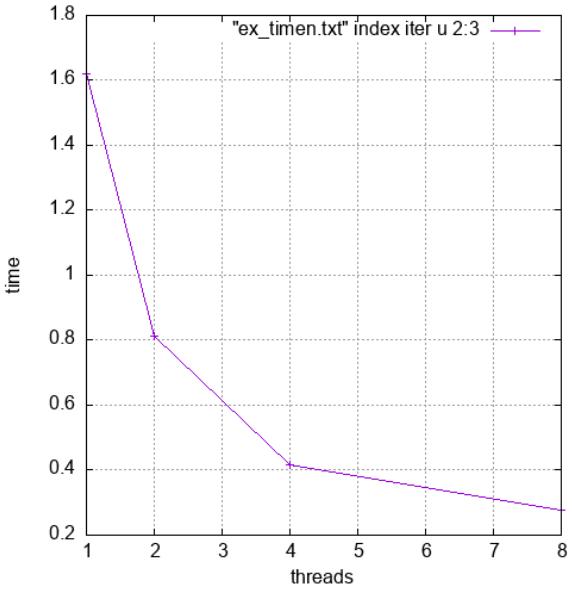


Ускорение (для выполнения преобразования)

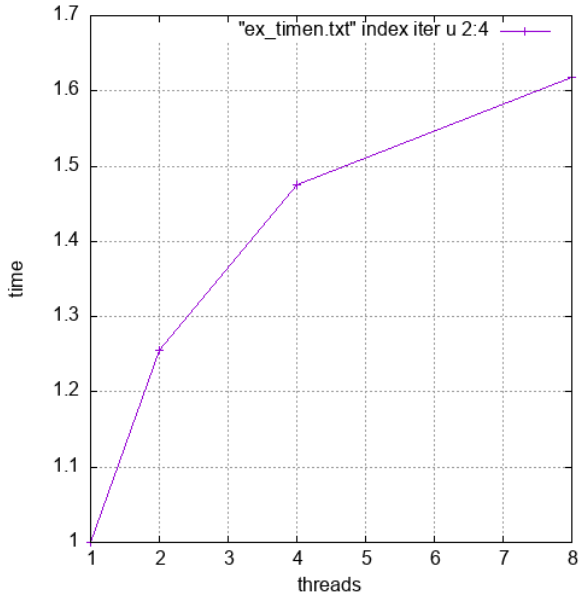
n = 24



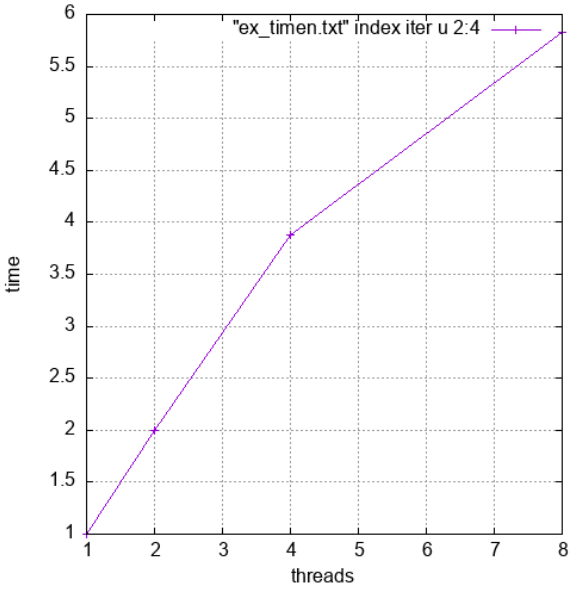
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей

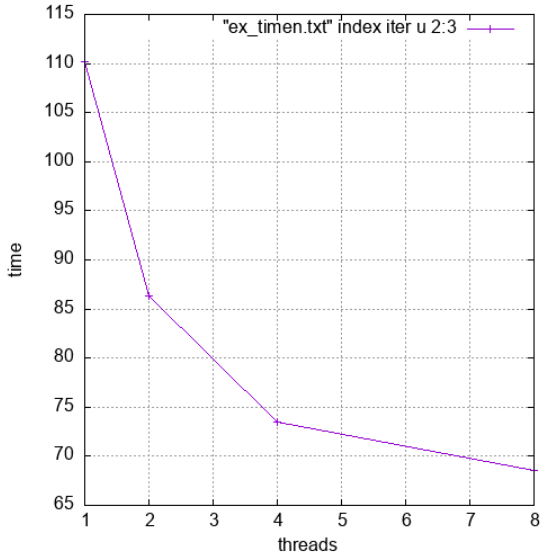


Ускорение (для полного выполнения программы)

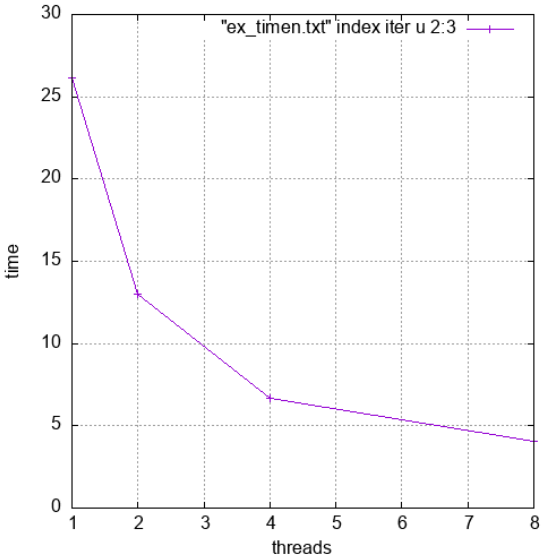


Ускорение (для выполнения преобразования)

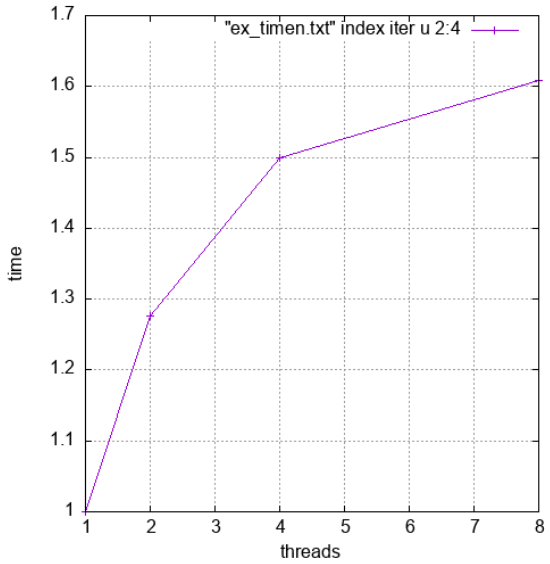
n = 28



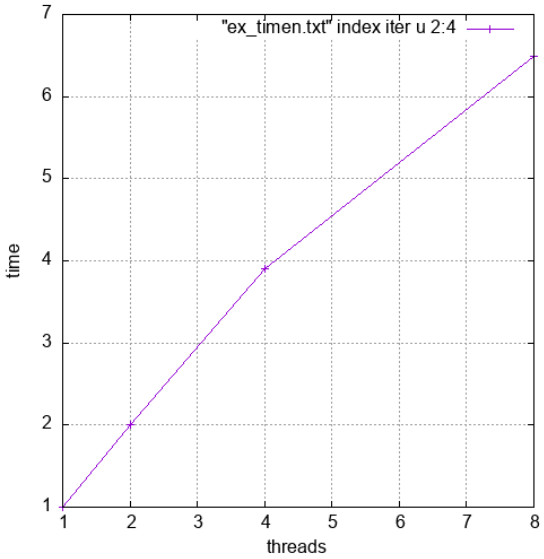
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей

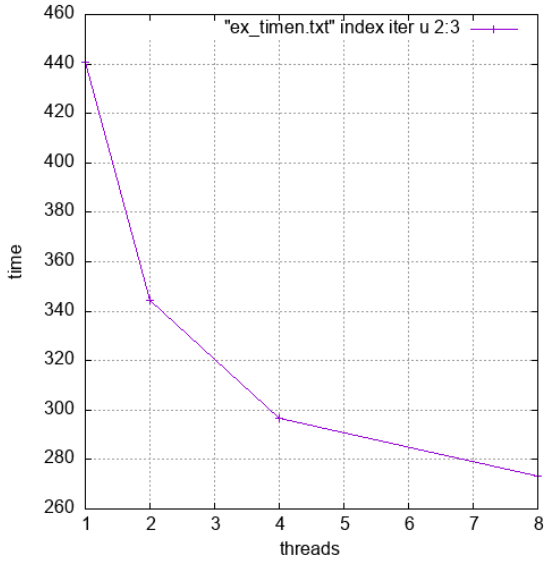


Ускорение (для полного выполнения программы)

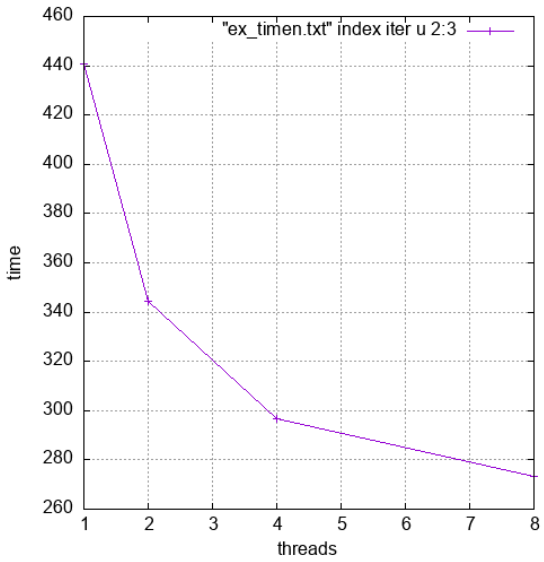


Ускорение (для выполнения преобразования)

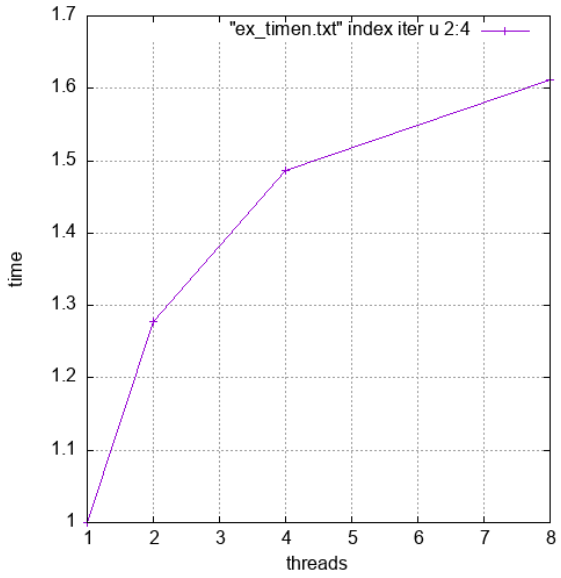
n = 30



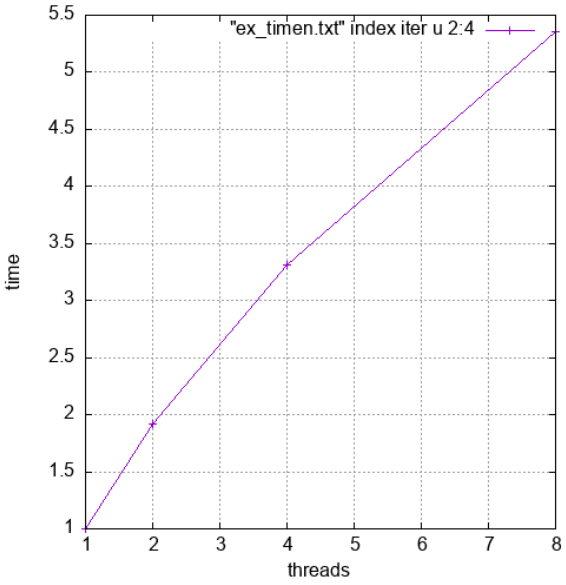
Зависимость полного времени работы от числа нитей



Зависимость времени выполнения преобразования от числа нитей



Ускорение (для полного выполнения программы)



Ускорение (для выполнения преобразования)

Ускорение полного выполнения программы существенно меньше ускорения выполнения только преобразования вектора в силу того, что инициализация вектора реализована последовательно. Такая реализация объясняется тем, что функция инициализации использует обращение к функции rand() в цикле, что приводит к замедлению параллельной программы с увеличением числа нитей.