Домашнее задание №2

Описать вычислительный узел

Наименование и краткая характеристика CPU :

- Модель процессора (Model name): Intel(R) Xeon(R) Gold 6248 CPU @ 2.50GHz
- Архитектура (Architecture): x86_64
- Число логических процессоров (CPU(s)): 80.

  CPU(s) = Core(s) per socket × Socket(s) × Thread(s) per core

- Число потоков (Thread(s) per core): 2
- Количество физических ядер в одном сокете (Core(s) per socket): 20
- Число сокетов (Socket(s)): 2
- Тактовая частота (CPU MHz):
  - *Максимальная частота (CPU max MHz)*: 3900.0000 MHz
  - *Минимальная частота (CPU min MHz)*: 1000.0000 MHz
- NUMA: Система поддерживает NUMA

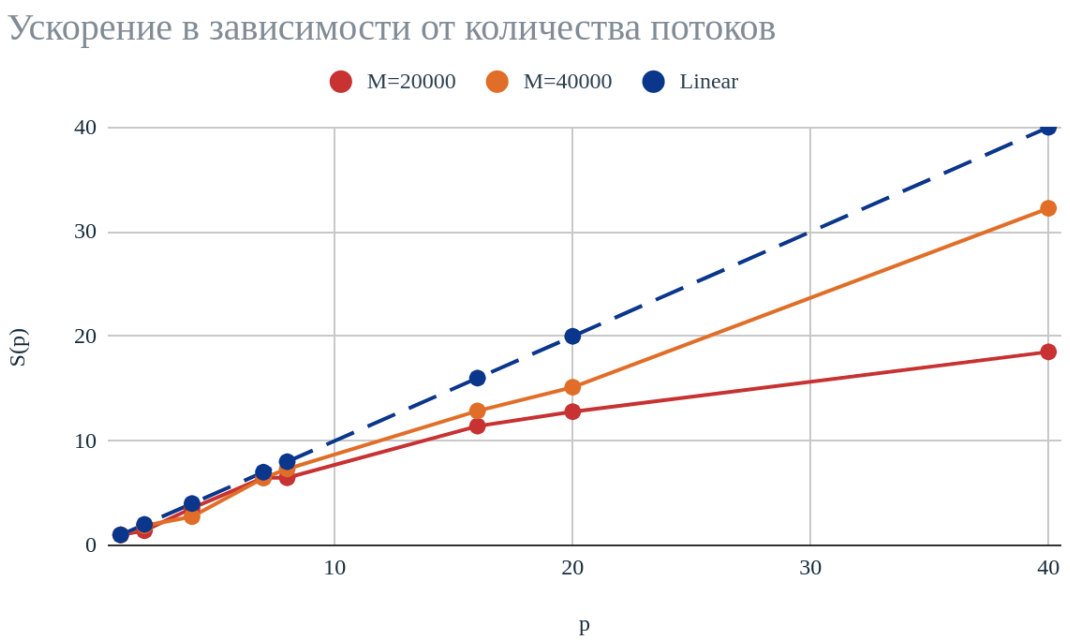
  Наименование сервера : ProLiant XL270d Gen10

- *NUMA node(s)* : 2
  - *node 0 size* : 385636 MB
  - *node 1 size* : 387008 MB
- *Операционная система* : Ubuntu 22.04.5 LTS

Задание 1

Умножение матрицы на вектор с параллельной инициализацией массивов

M=N	Количество потоков															
	1		2		4		7		8		16		20		40	
	T(1)	S(1)	T(2)	S(2)	T(4)	S(4)	T(7)	S(7)	T(8)	S(8)	T(16)	S(16)	T(20)	S(20)	T(40)	S(40)
20000 (~3GiB)	1,2697	1	0,9073	1,3994	0,3573	3,5536	0,1966	6,4583	0,1964	6,4649	0,1113	11,408	0,0993	12,787	0,0686	18,509
40000 (~12GiB)	5,0632	1	2,7016	1,8741	1,8435	2,7465	0,786	6,4417	0,6931	7,3052	0,3937	12,861	0,3347	15,128	0,157	32,250

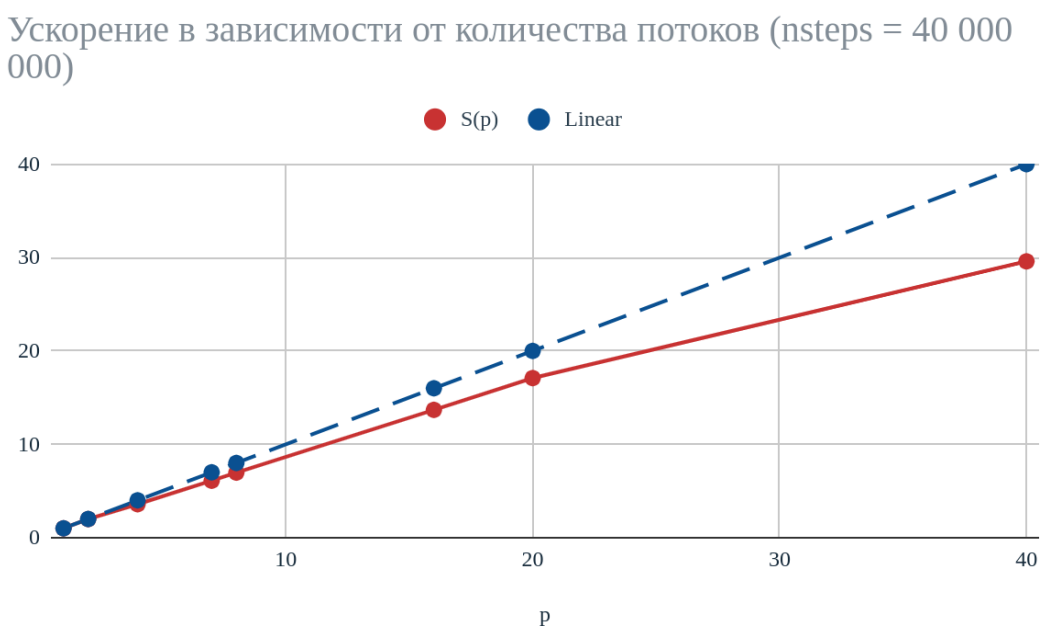


Ускорение увеличивается с количеством потоков, параллельная обработка эффективно сокращает время вычислений.
Для обоих наборов данных (малого и большого размера) ускорение увеличивается с увеличением числа потоков.
Ускорение не является линейным, что указывает на наличие накладных расходов, связанных с параллельным выполнением.
Для большего размера массива ускорение выше, чем для меньшего, особенно при большом количестве потоков.

Задание 2

Параллельная версия программы численного интегрирования

nsteps	Количество потоков															
	1		2		4		7		8		16		20		40	
	T(1)	S(1)	T(2)	S(2)	T(4)	S(4)	T(7)	S(7)	T(8)	S(8)	T(16)	S(16)	T(20)	S(20)	T(40)	S(40)
40 000 000	0,6392	1	0,3212	1,9900	0,1784	3,5829	0,1048	6,0992	0,0917	6,9706	0,0467	13,687	0,0374	17,091	0,0216	29,592



Ускорение значительно увеличивается с ростом числа потоков. Например, при использовании 40 потоков ускорение достигает почти 30, что свидетельствует о значительном улучшении производительности по сравнению с однопоточным выполнением.

Система эффективно масштабируется с увеличением числа потоков, что позволяет лучше распределять вычислительные задачи и уменьшать время выполнения.

Задание 3

Параллельная реализация решения системы линейных алгебраических уравнений с помощью OpenMP

M=N=40000	Количество потоков																	
	1		2		4		7		8		16		20		40		80	
	T(1)	S(1)	T(2)	S(2)	T(4)	S(4)	T(7)	S(7)	T(8)	S(8)	T(16)	S(16)	T(20)	S(20)	T(40)	S(40)	T(80)	S(80)
static	173,5491	1	199,7685	**0,86875108**	117,9625	**1,471222634**	114,2679	**1,518791367**	98,6608	**1,75904818**	64,8333	**2,67681248**	59,1801	**2,93251158411**	38,4718	**4,51107305**	38,617	**4,49411140**
auto	169,3684	1	142,5223	**1,188364207**	102,8767	**1,6463242**	110,1387	**1,537773734**	100,1674	**1,6985351**	64,2094	**2,63774850859**	57,6567	**2,93751931978**	38,4159	**4,4080989**	38,9026	**4,3536565246**
guided	189,7631	1	151,5091	**1,252486484**	151,2278	**1,2541624**	118,7527	**1,59796872**	98,4146	**1,928200694**	63,2708	**2,99922081**	57,6678	**3,2906262492**	38,1191	**4,978163178**	38,7605	**4,8957878566**
dynamic	168,5931	1	128,9215	**1,307719038**	120,3048	**1,4013838299**	112,1419	**1,5033939079**	97,4564	**1,729933591**	65,533	**2,57264431732**	57,6972	**2,92203232612**	38,3823	**4,3924696997**	38,8825	**4,3359696348**
#pragma omp parallel	175,0463	1	154,2753	**1,13463594**	58,3614	**2,999350598**	39,6475	**4,41506526**	34,6536	**5,05131646**	14,1802	**12,3444169**	16,3526	**10,7044935**	9,5968	**18,2400696**	9.9500	**12.189**

Вариант 1: Отдельные параллельные секции для каждого цикла (#pragma omp parallel for)
В этом варианте наблюдается значительное ускорение при увеличении числа потоков. Например, при использовании 80 потоков ускорение достигает примерно 4.5 раза по сравнению с однопоточным выполнением.
Различные стратегии распределения нагрузки (static, auto, guided, dynamic) показывают схожие результаты, но static и dynamic демонстрируют немного лучшие показатели ускорения.
Вариант 2: Одна параллельная секция для всего алгоритма (#pragma omp parallel)
Ускорение также наблюдается, но оно менее значительное по сравнению с первым вариантом. При использовании 80 потоков ускорение составляет около 12.2 раза.
Этот вариант может быть менее эффективным из-за возможных накладных расходов на создание и управление одной большой параллельной секцией.

Первый вариант (#pragma omp parallel for) оказывается более целесообразным для данной задачи, так как он обеспечивает лучшее ускорение и более эффективное использование ресурсов.