ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
основан в 1878 году

Сравнение различных способов декомпозиции сеточной области при численном решении уравнения переноса

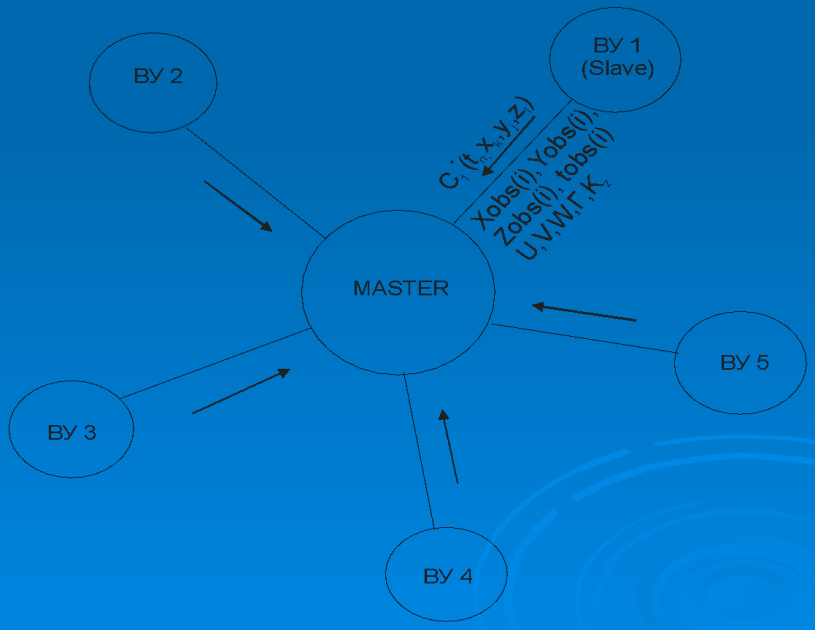
Е.А. Данилкин, А.В. Старченко
Томский государственный университет

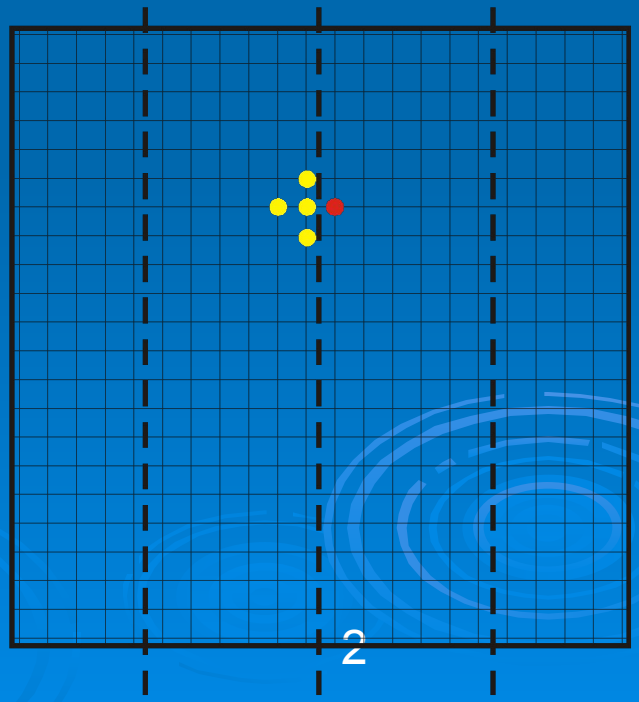
Цель работы

Целью данной работы было достижение максимальной эффективности при распараллеливании данного класса задач.

Две парадигмы параллельного программирования:

- распараллеливание по физическим процессам
- распараллеливание по данным

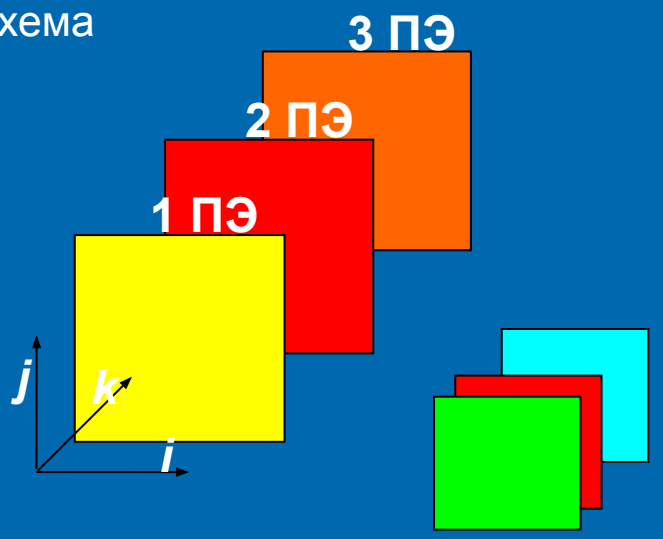


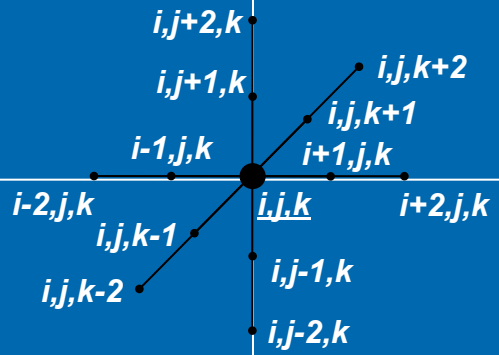


Виды декомпозиции

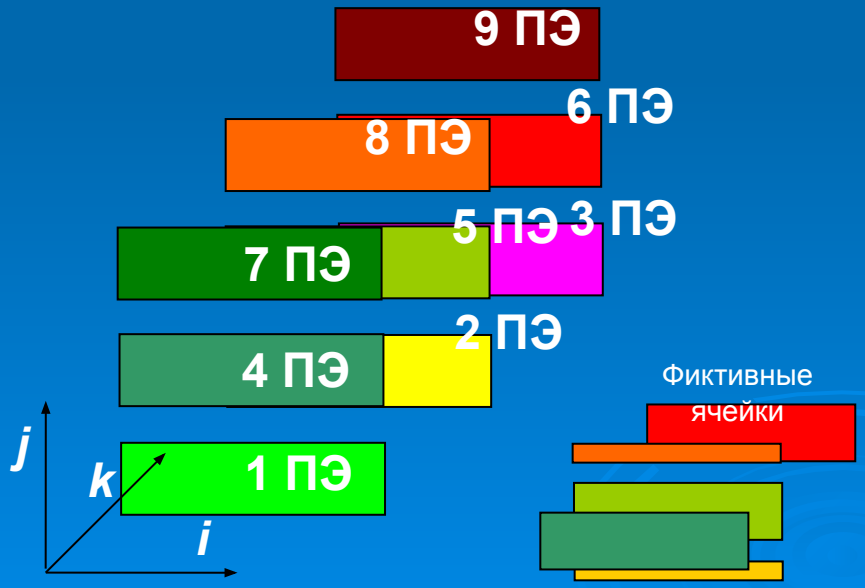
Наиболее общим подходом равномерного распределения вычислительной загрузки между вычислительными узлами при решении сеточных уравнений является распределение вычислительных областей на подобласти. Так называемый принцип геометрической декомпозиции.

1D-декомпозиция
ленточная схема

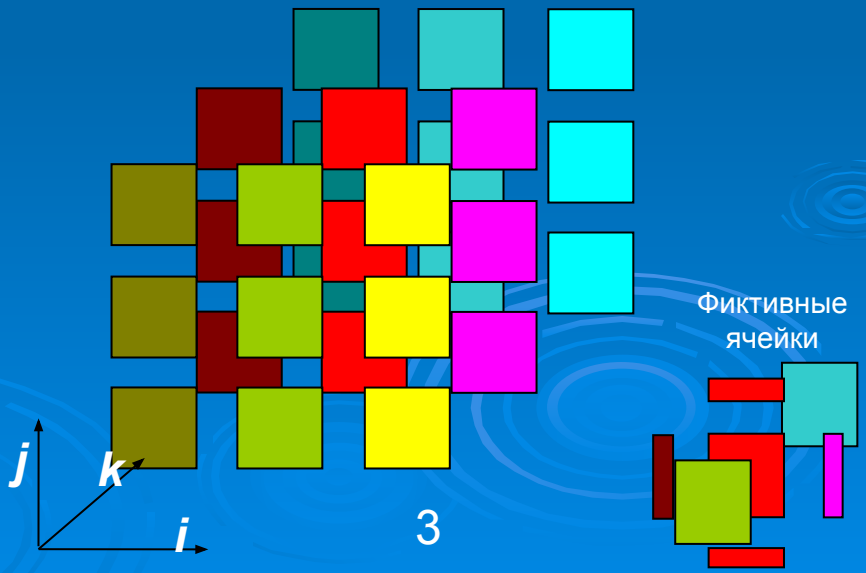




2D-декомпозиция *блочное* разбиение



3D-декомпозиция



Математическая постановка

Требуется решить трехмерное уравнение переноса в единичном кубе

$$\frac{\partial \Phi}{\partial t}+\frac{\partial (u\Phi)}{\partial x}+\frac{\partial (v\Phi)}{\partial y}+\frac{\partial (w\Phi)}{\partial z}=\frac{\partial}{\partial x}\left[\Gamma\frac{\partial \Phi}{\partial x}\right]+\frac{\partial}{\partial y}\left[\Gamma\frac{\partial \Phi}{\partial y}\right]+\frac{\partial}{\partial z}\left[\Gamma\frac{\partial \Phi}{\partial z}\right]+S$$

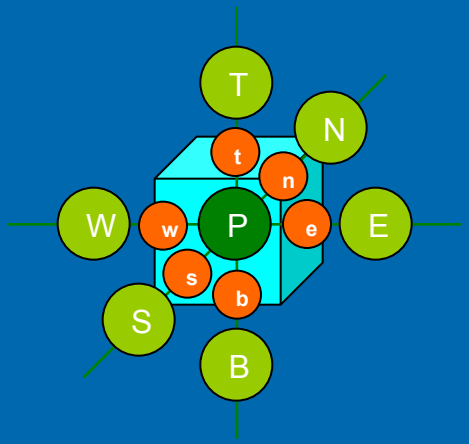
с граничными условиями второго рода и заданными начальными условиями

$$\left.\frac{\partial \Phi}{\partial x}\right|_{x=0}=0;\quad \left.\frac{\partial \Phi}{\partial y}\right|_{y=0}=0;\quad \left.\frac{\partial \Phi}{\partial z}\right|_{z=0}=0;\quad \Phi|_{t=0}=0;$$

$$\left.\frac{\partial \Phi}{\partial x}\right|_{x=1}=0;\quad \left.\frac{\partial \Phi}{\partial y}\right|_{y=1}=0;\quad \left.\frac{\partial \Phi}{\partial z}\right|_{z=1}=0;$$

Функции *u, v, w*, Γ>0 и *S* - определены в рассматриваемой области.

Аппроксимация

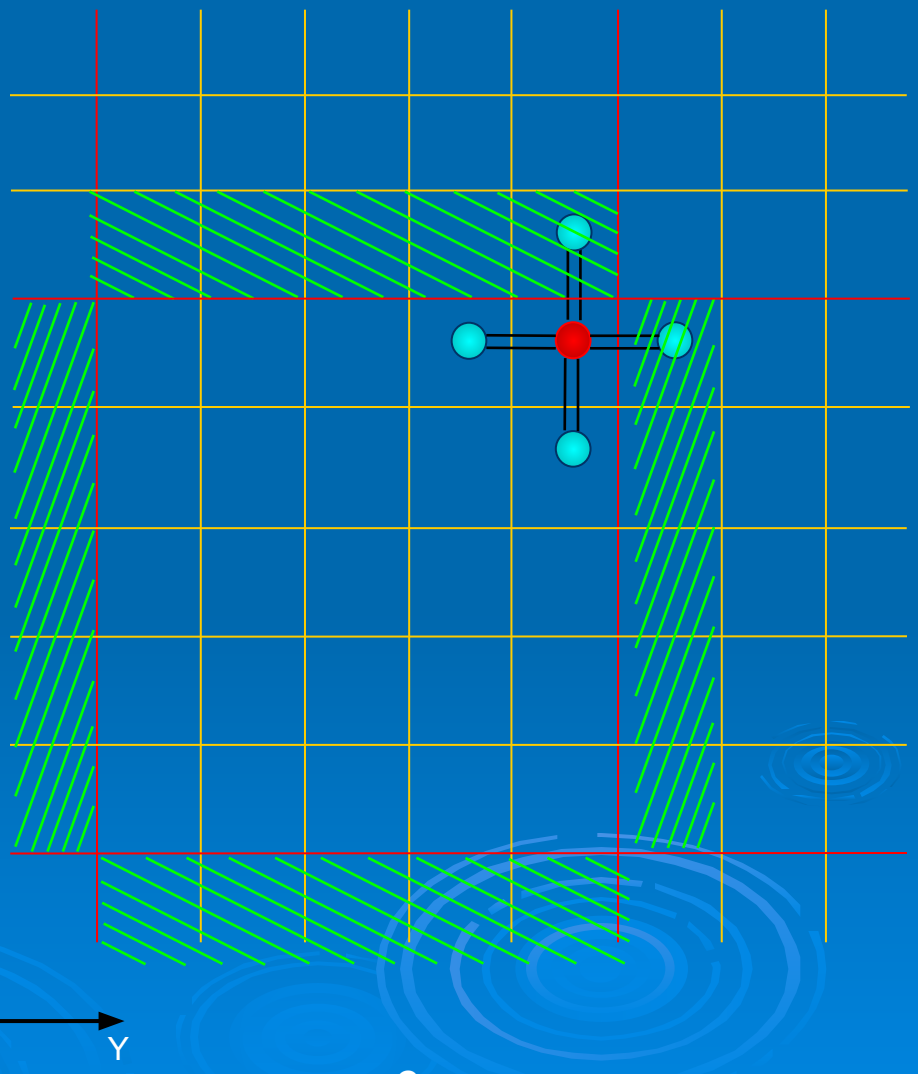


Аппроксимация дифференциальной задачи осуществляется на основе метода конечного объема. Аппроксимация конвективных членов уравнений переноса выполняется с использованием противопотоковой схемы. Для расчета применялась явная схема, поэтому результатом приближенного интегрирования по одному конечному объему является готовая для вычислений формула:

$$ap^0\Phi^{n+1}_{i,j,k} = ap_{i,j,k}\Phi^{n}_{i,j,k} + ae_{i,j,k}\Phi^{n}_{i+1,j,k} + an_{i,j,k}\Phi^{n}_{i,j+1,k} + at_{i,j,k}\Phi^{n}_{i,j,k+1}$$
$$+ aw_{i,j,k}\Phi^{n}_{i-1,j,k} + as_{i,j,k}\Phi^{n}_{i,j-1,k} + as_{i,j,k}\Phi^{n}_{i,j,k-1} + b_{i,j,k}$$

Планирование коммуникаций

Данные распределены, следующий этап построения параллельной программы – это планирование коммуникаций. В силу используемого шаблона схемы, для вычисления очередного приближения в приграничных узлах подобласти необходимо знать значения с соседнего процессора. Это процедура хорошо известна и поэтому долее детально остановимся на результатах сравнения различных способов декомпозиции.



Теоретический анализ

$$T_p = \frac{T_{calc}}{p} + T_{com};$$

n³ – размерность задачи

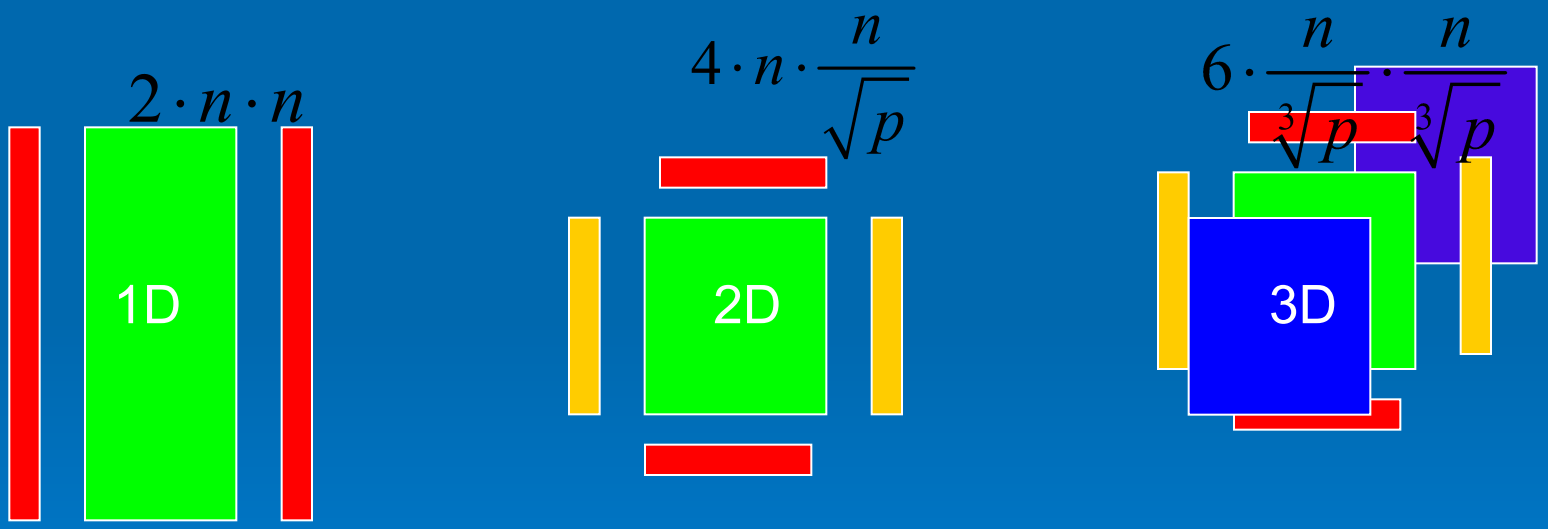
p – количество процессоров

***t*send** – время пересылки одного числа

$$T_{com}^{1D} = t_{send} \cdot n^2 * 2;$$

$$T_{com}^{2D} = t_{send} \cdot n^2 * \frac{4}{\sqrt{p}};$$

$$T_{com}^{3D} = t_{send} \cdot n^2 * \frac{6}{\sqrt[3]{p^2}};$$



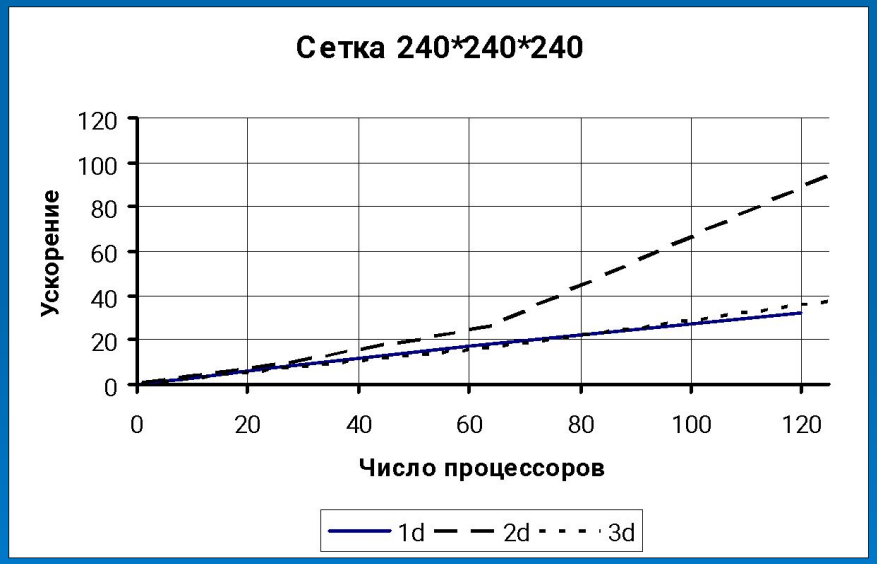
Число процессов	3	4	7	10	13	16	64
1d декомпозиция	2	2	2	2	2	2	2
2d декомпозиция	2.31	2.00	1.51	1.26	1.11	1.00	0.50
3d декомпозиция	2.88	2.38	1.64	1.29	1.09	0.94	0.38

Вычислительный кластер имеет 283 вычислительных узла, один узел включает два двухъядерных процессора Intel 5150, 2,66ГГц, 4 Гб оперативной памяти, 80 Гб жесткий диск, дополнительно 10 Тб быстрой памяти, сеть - Infiniband

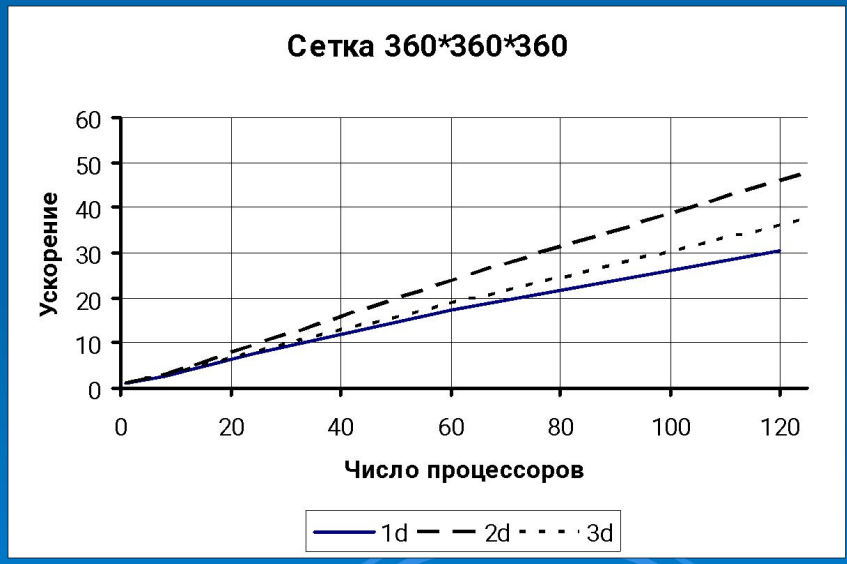
Распараллеливание

Основное внимание уделялось сравнению времени пересылок и времени вычислений при различных способах декомпозиции. Несмотря на теоретический анализ, практика показала, что лучших результатов можно достичь, используя 2D декомпозицию.

a)

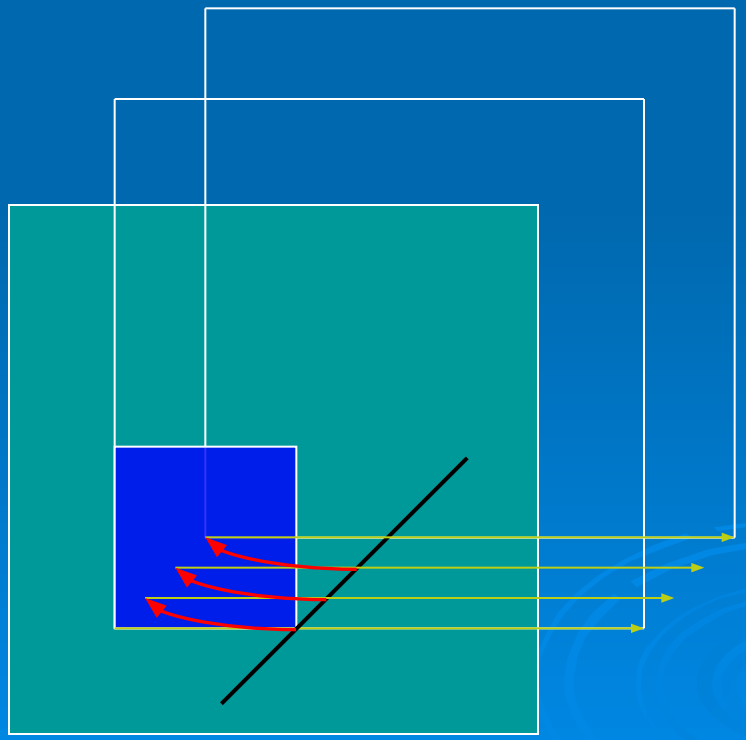


b)



Для объяснения полученных результатов необходимо обратить внимание на допущения, которые были сделаны при предварительном теоретическом анализе поставленной задачи.

- Во-первых, предполагалось, что независимо от способа распределения данных на одном процессорном элементе выполняется одинаковый объем вычислительной работы, который должен приводить к идентичным временным затратам.
- Во-вторых, принималось, что время, затраченное на межпроцессорную пересылку любой последовательности одинакового количества данных, не зависит от способа их выборки из оперативной памяти.

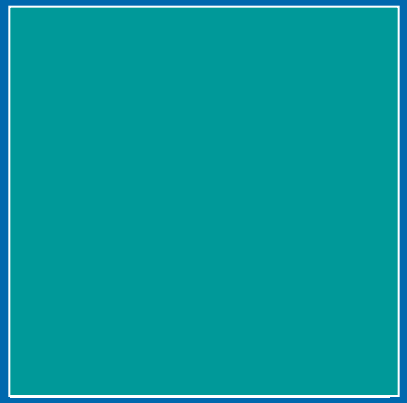
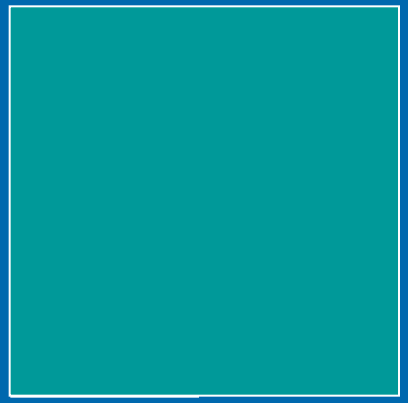
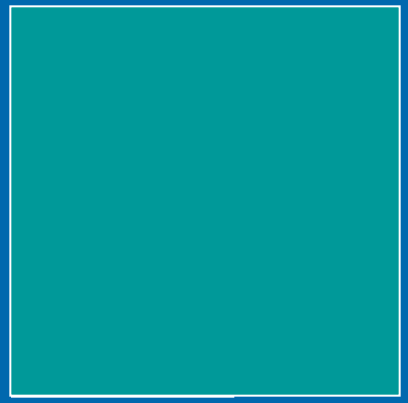


```
Do k=1,n          a1 1 1
  Do j=1,n        a2 1 1
    Do i=1,n      a3 1 1
```

Для оценки состоятельности первого допущения был рассмотрен случай, когда программа запускалась в однопроцессорном варианте, и при этом имитировались различные способы геометрической декомпозиции данных при одинаковом объеме вычислений, выполняемом каждым процессором.

Выборка данных из памяти

1D 2D 3D

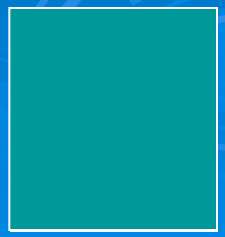
Без учета размерности массивов

p	1d	2d	3d
8	9,3	8,1	10,4
24	3,1	2,6	3,7
60	1,2	1,0	1,6

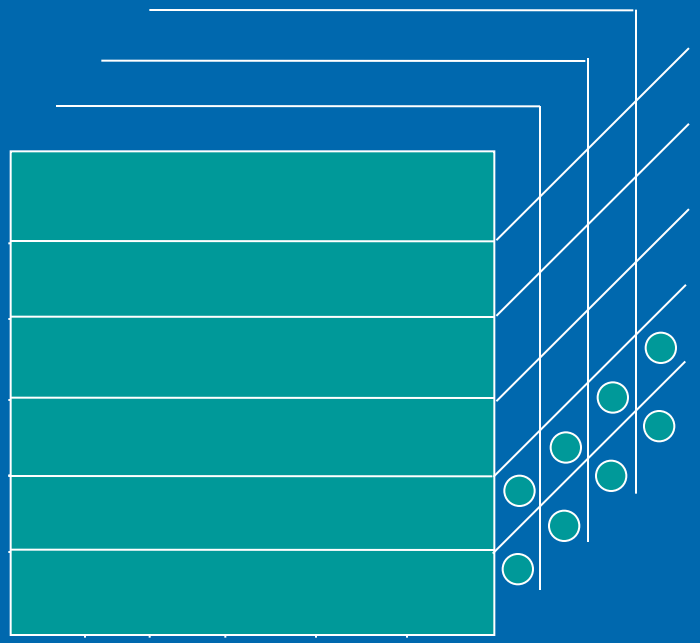
С учетом размерности массивов

p	1d	2d	3d
8	8,8	7,5	7,3
24	3,0	2,5	2,5
60	1,2	0,9	1,0





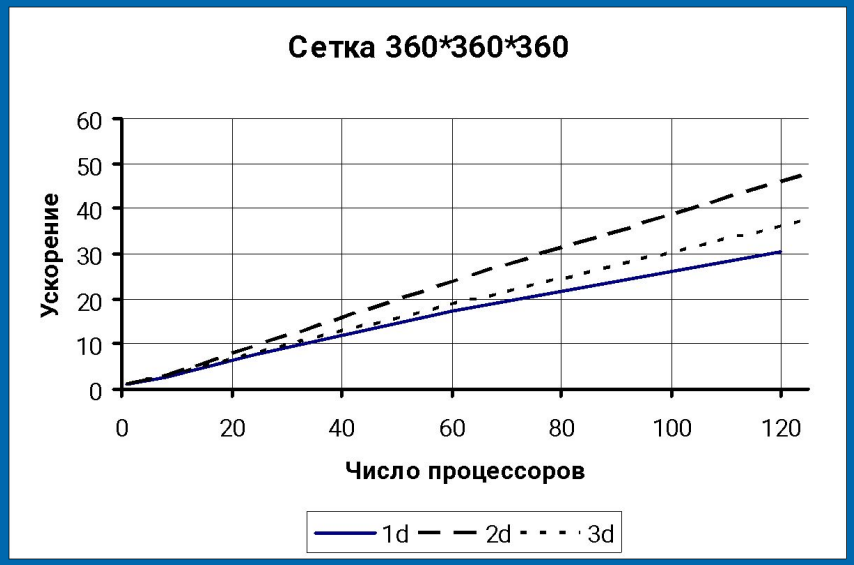
Выборка данных из памяти



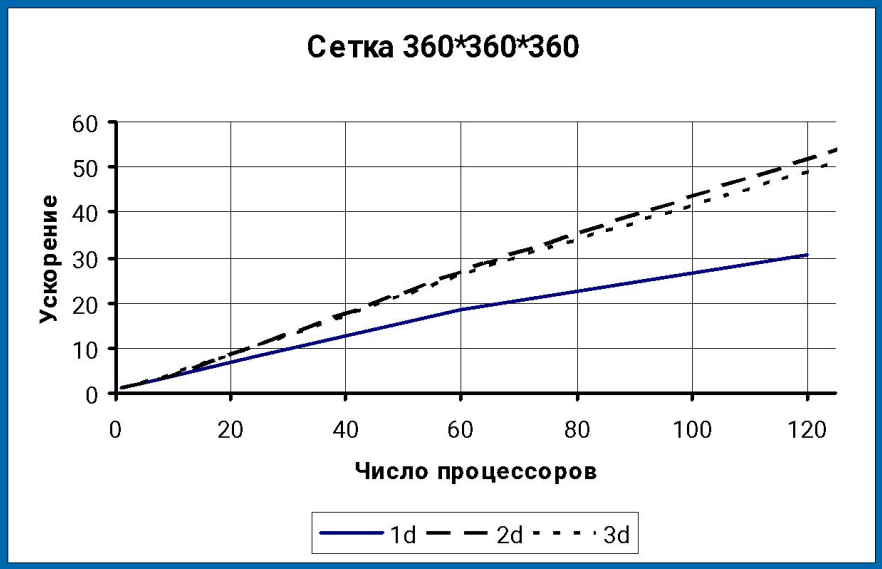
Для оценки реализуемости второго допущения (Tcom = tsend·N2 *k(p)) был рассмотрен тест, в котором измерялось время MPI-пересылки различных сечений массива с размерностью 3. Заметим, что решение этой задачи (пересылки, скажем, i-j, i-k, j-k – сечений массива, элементы которого имеют индексы i, j, k) в стандарте MPI может осуще ся различными способами в зависим ти от того, какие сечения массивов рассматриваются.

Количество элементов в массиве\ сечение	i-j	i-k	j-k
30^3	0,00189	0,00192	0,00265
60^3	0,00402	0,00556	0,00884
120^3	0,01601	0,02151	0,03563

Распараллеливание



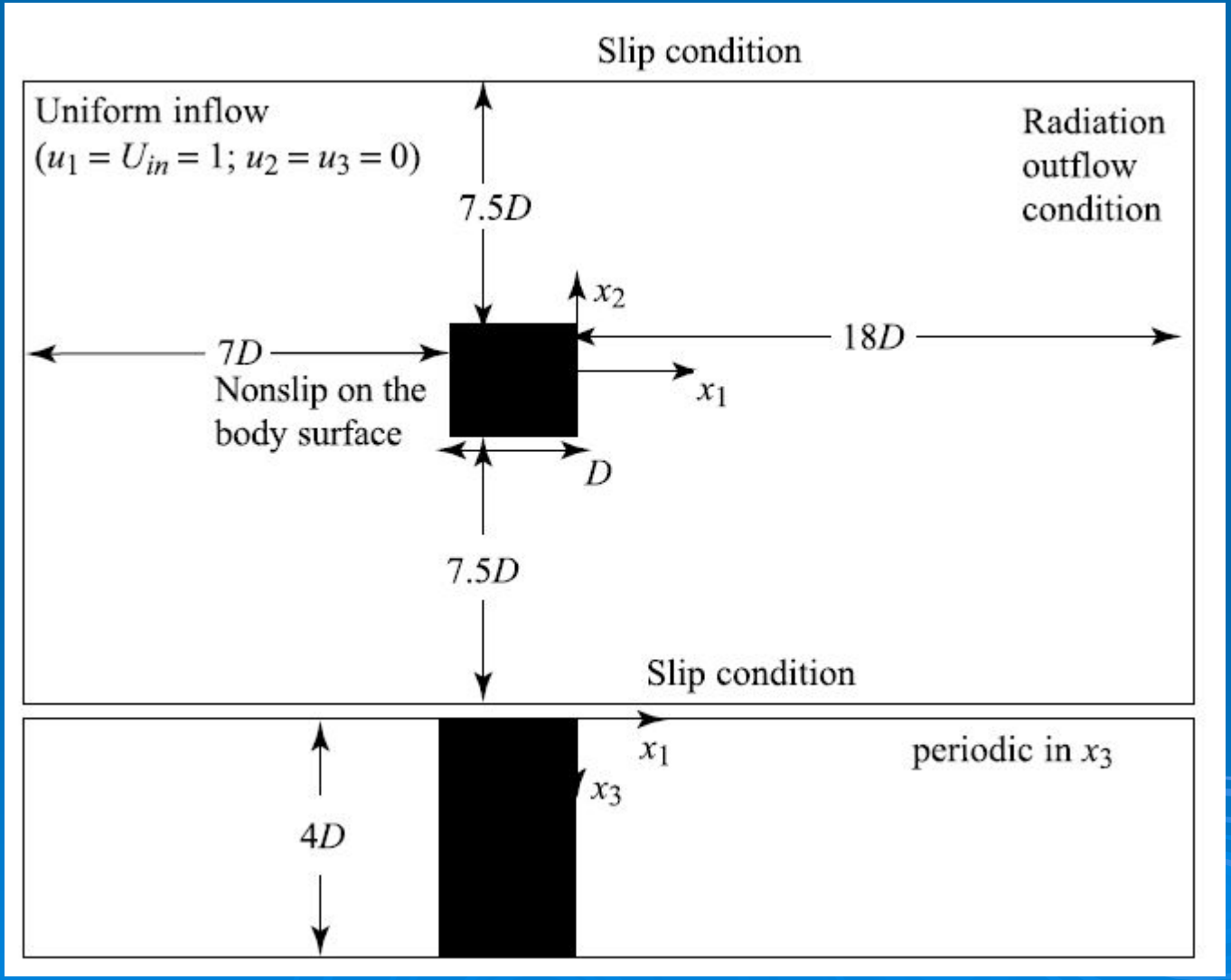
До оптимизации



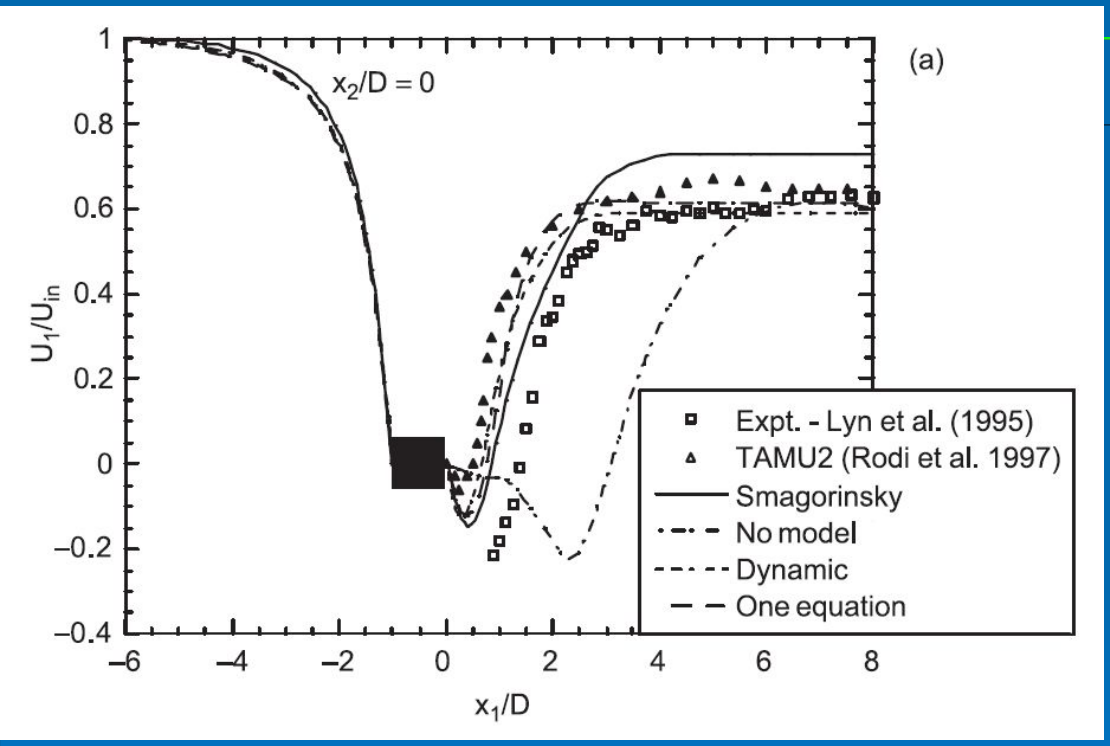
После оптимизации

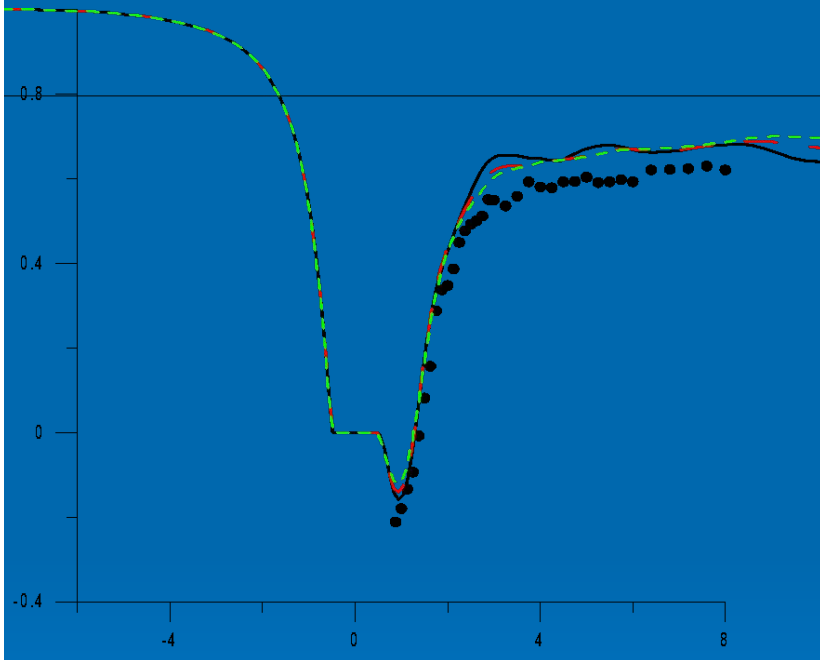
Видно что 2D декомпозиция является оптимальным вариантом для распараллеливания, сохраняя возможность масштабирования и простору реализации. При этом для 3D декомпозиции можно добиться аналогичных результатов по эффективности. Потратив некоторое время на оптимизацию программы.

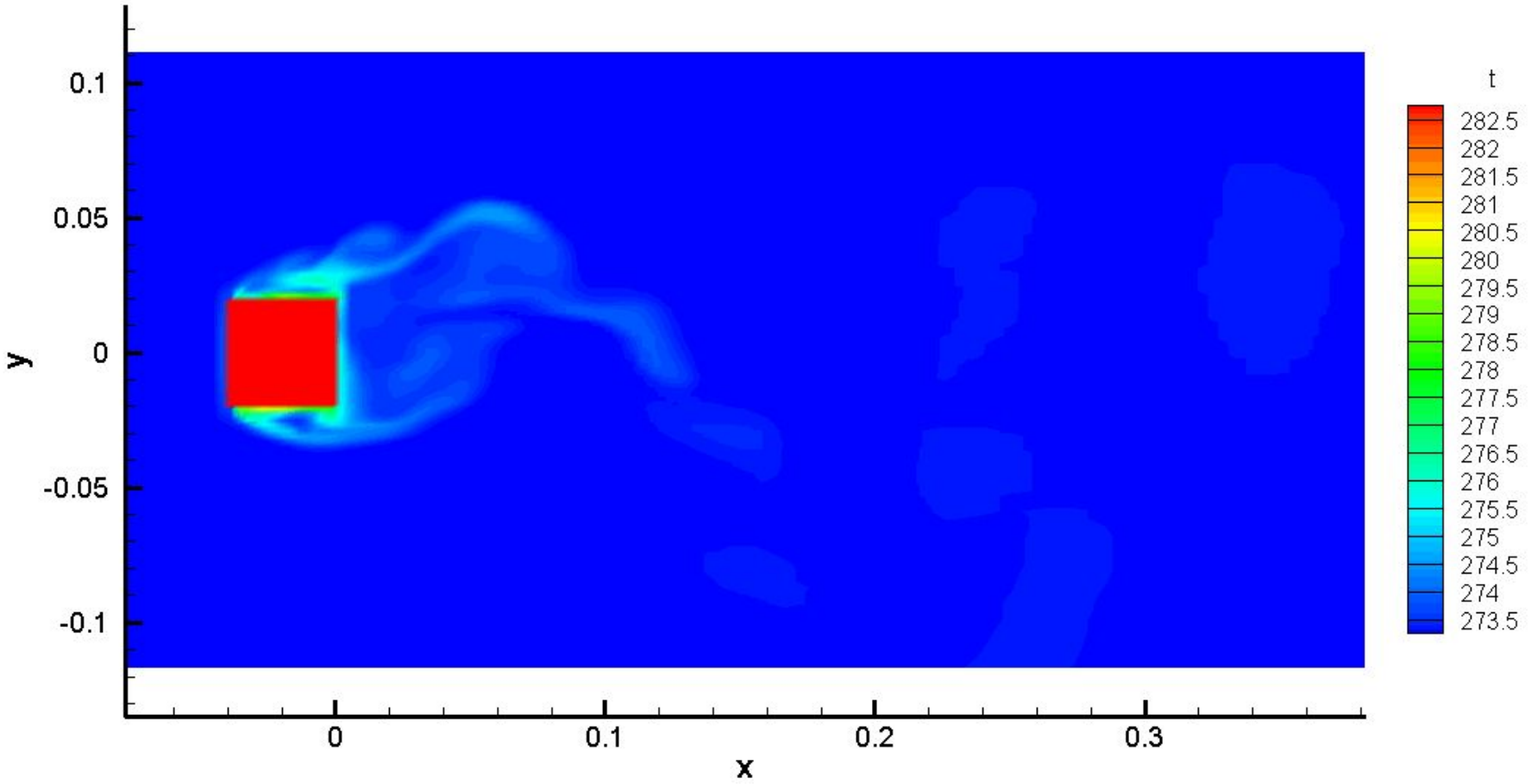
Тестовая задача, Lyn (1995). NAKAYAMA



Сравнение результатов.






0.1
0.05
0
-0.05
-0.1
y
0
0.1
0.2
0.3
x
t
282.5
282
281.5
281
280.5
280
279.5
279
278.5
278
277.5
277
276.5
276
275.5
275
274.5
274
273.5

Спасибо за внимание.