HPC:

суперкомпьютерные вычисления: краткий обзор курса

И. Н. Коньшин[1,2,3,4]

[1]Институт вычислительной математики им. Г.И. Марчука, РАН
[2]Московский физико-технический институт
[3]Сеченовский университет
[4]Университет Сириус

План: лекции и практика

Обзор лекций:

- суперкомпьютеры →
- OpenMP + MPI →
- линейная алгебра + матфизика + задачи оптимизации →
- статический + динамический //-зм →
- теория + возможности готовых пакетов →

Дополнительные лекции:

- Распараллеливание на OpenMP v.5.0 или MPI v.3.1 или C++11
- Распараллеливание на OpenCL или OpenACC или Cuda
- Распараллеливание на python

Практика по C++ и Py: (4 – 9)

Практика на кластере ИВМ РАН: →

Задания:

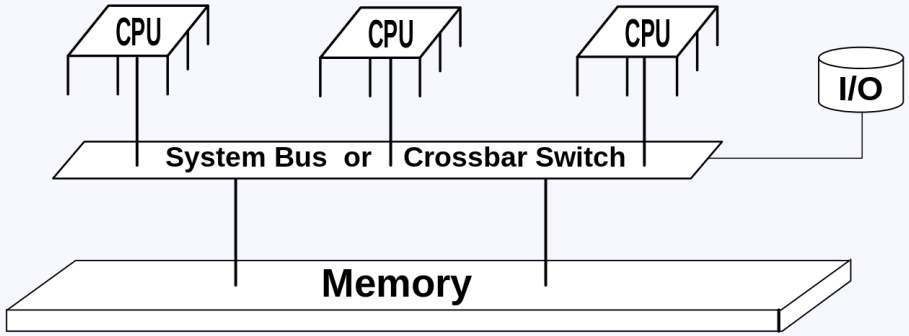
1) викторина →
2) теоретическая задача →
3) практическое задание: на выбор {1–9} mvm diff

Лекция 1: суперкомпьютеры

- История
- Закон Мура
- Советские и российские компьютеры
- Top-500
- Доклад Jack Dongarra, HPC
- HPL vs HPCG
- Будущее HPC
- Архитектура и классификация компьютеров
- Применение суперкомпьютеров

OpenMP = распараллеливание арифметики

Общая память:



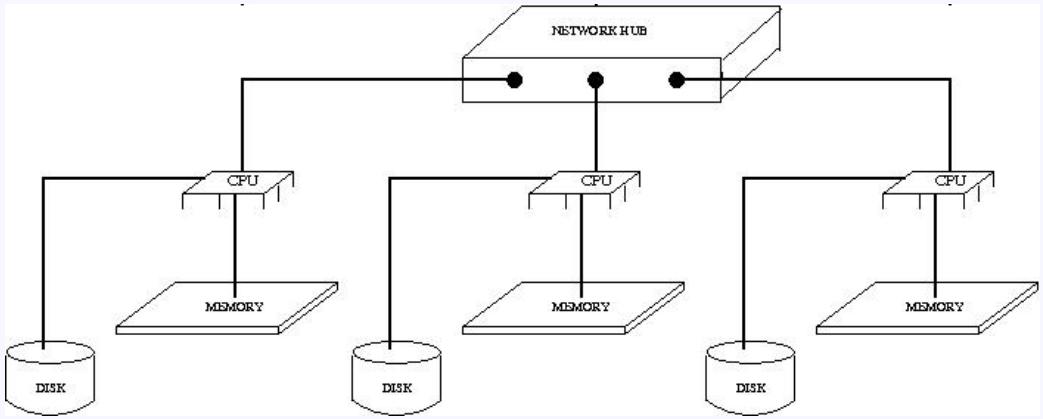
- макросы
- директивы препроцессора
- библиотечные функции
- переменные окружения

Оценка //-ной эфф-ти: Закон Амдала [Amdahl's law, 1967]

$$S(p) = \frac{p}{1 + \sigma(p-1)}$$

MPI = распараллеливание данных

Распределенная память



- Коммуникаторы
- Обмены точка-точка
- Коллективные обмены
- Специальные функции

Оценка //-ной эфф-ти: [ИНК, 2012]

$$S(p) = \frac{p}{1 + \tau L}, \qquad \tau = \frac{\tau_c}{\tau_a}, \quad L = \frac{L_c}{L_a}$$

Виды параллелизма и оценка //-ной эфф-ти

Статический //-зм

- линейная алгебра
- матфизика

Динамический //-зм

- задачи оптимизации

- Теоретическая оценка //-ной эфф-ти
- Измерение //-ной эфф-ти на практике
- Зависимость //-ной эфф-ти от компьютера

Практическая работа на кластере ИВМ РАН

- команды Unix
- редактор vim
- кластер ИВМ РАН: http://cluster2.inm.ras.ru
- работа на кластере ИВМ РАН
- запуск готовых тестов
- написание своих тестов
- mvm – исследование параллельной эфф-ти

http://dodo.inm.ras.ru/konshin/hpc

Параллельные методы суперкомпьютерных вычислений

И.Н.Коньшин (ИВМ РАН)

Занятия:

Дата: Тема... Задание... [иногда слайды]

Список файлов:

*) students_win.txt – список нашей объединенной группы

1) викторина – короткий тест из 10 вопросов

2) questions_win.txt – теор. задачи для зачета/экзамена

3) tasks_win.txt – практические задания

+) advanced_win.txt – список дополнительных лекций

Литература:

[1-2] – теория параллельных вычислений (Воеводин, Байдин)

[5-7] – оценка параллельной эффективности (ИНК)

[10-12] – описания некоторых примеров из практического задания

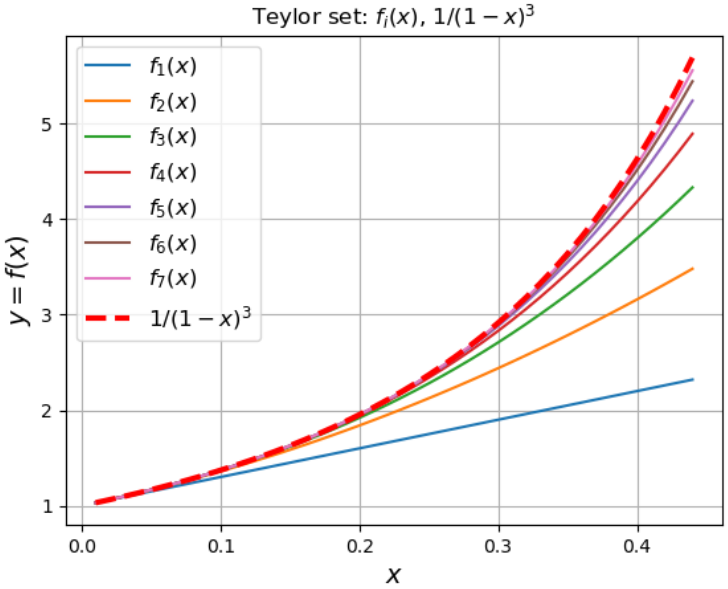
Все справочные файлы для работы на кластере:

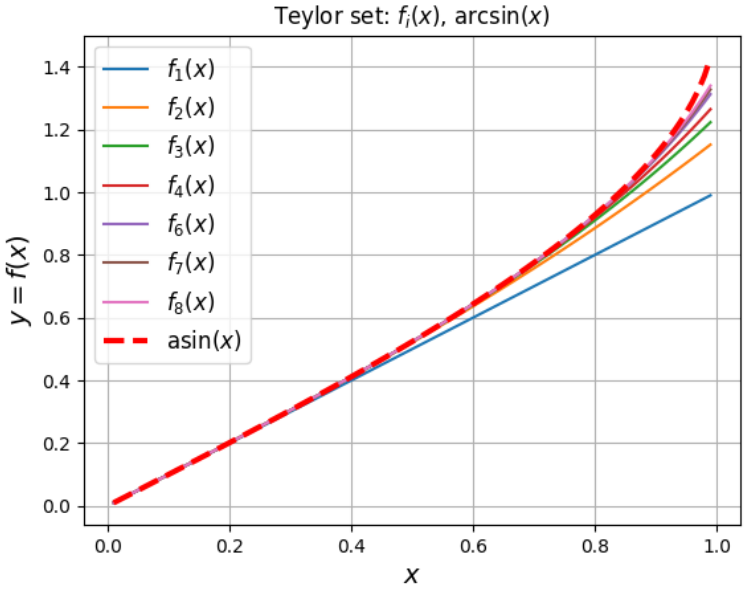
http://cluster2.inm.ras.ru/~student/2x-ooo

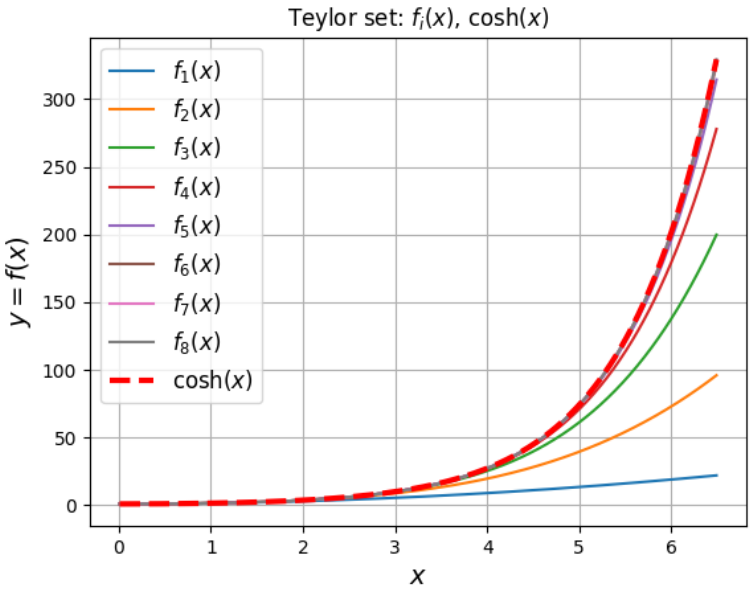
*) список нашей группы: [students_win.txt]

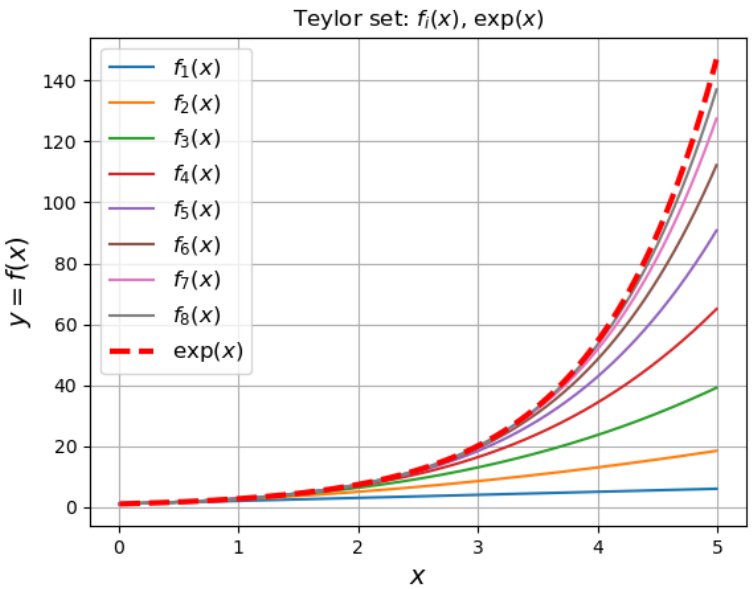
```
    СПИСОК СТУДЕНТОВ ОБЪЕДИНЕННОЙ ГРУППЫ
    "ПАРАЛЛЕЛЬНЫЕ МЕТОДЫ СУПЕРКОМПЬЮТЕРНЫХ ВЫЧИСЛЕНИЙ"
    преподаватель: Коньшин Игорь Николаевич igor.konshin@gmail.com

NM) Фамилия Имя                                  ....
NM) Фамилия Имя                                  ....

v - викторина из 10 вопросов                    v...
s - теоретическая задача на оценку ускорения    s..
t - практическое вычисление tau на кластере      t.
n - количество выполненных практ. заданий         n
o - итоговая оценка                                ooo(oo)
p - практическая задача (лекция)                          ppp
```

Практика по C++ и Py:

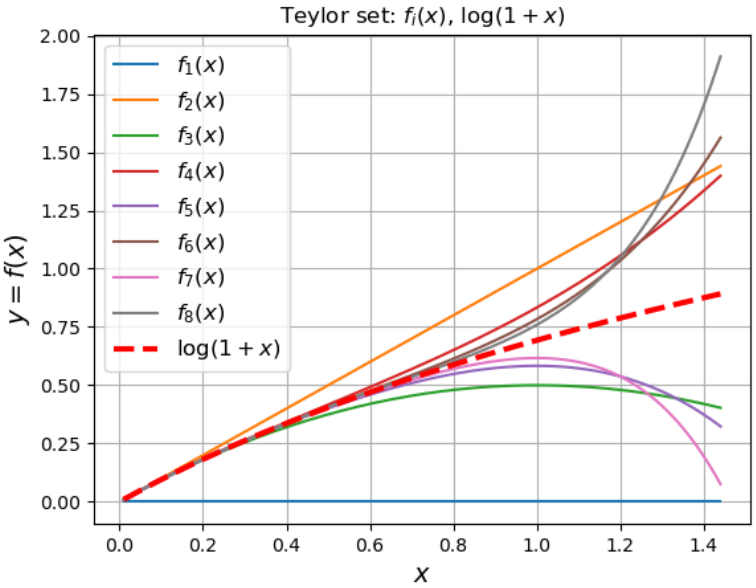


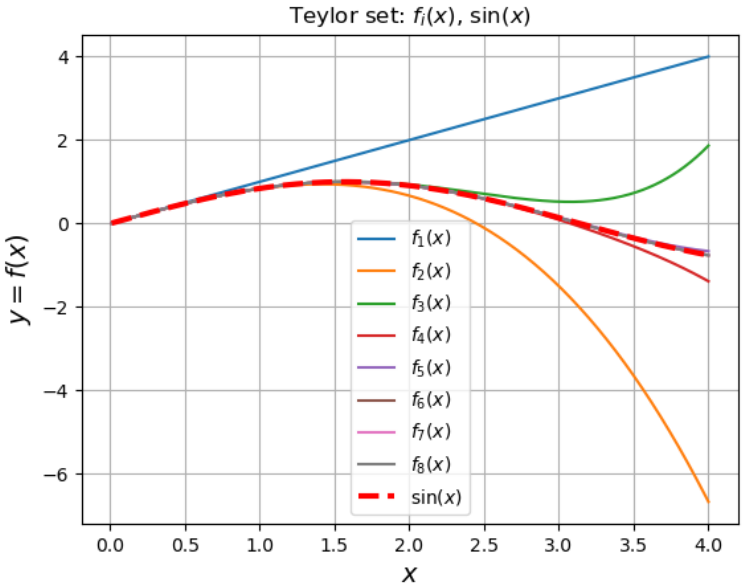


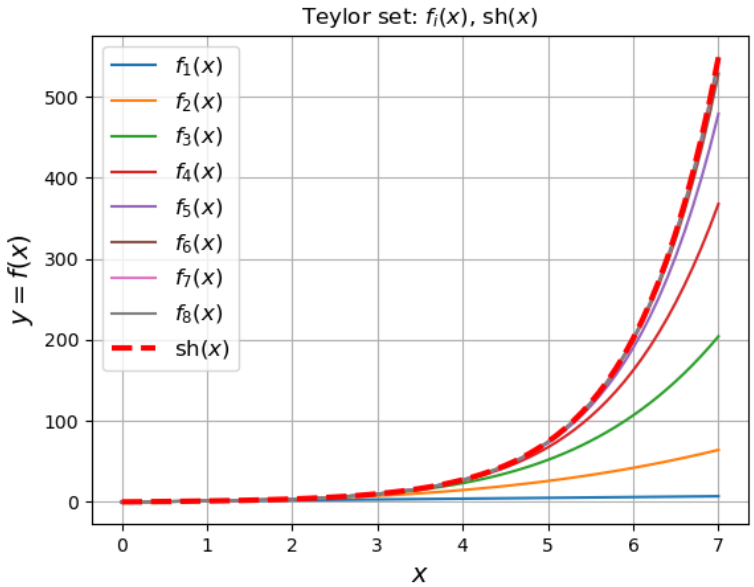


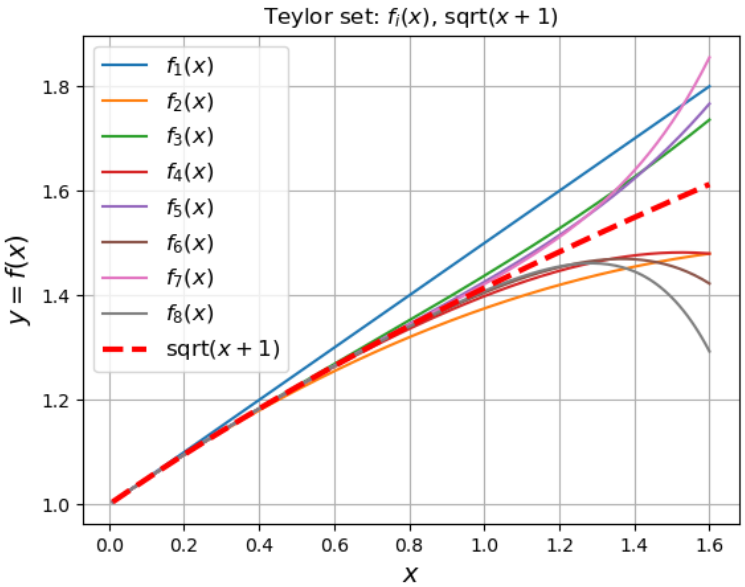


Практика по C++ и Py:



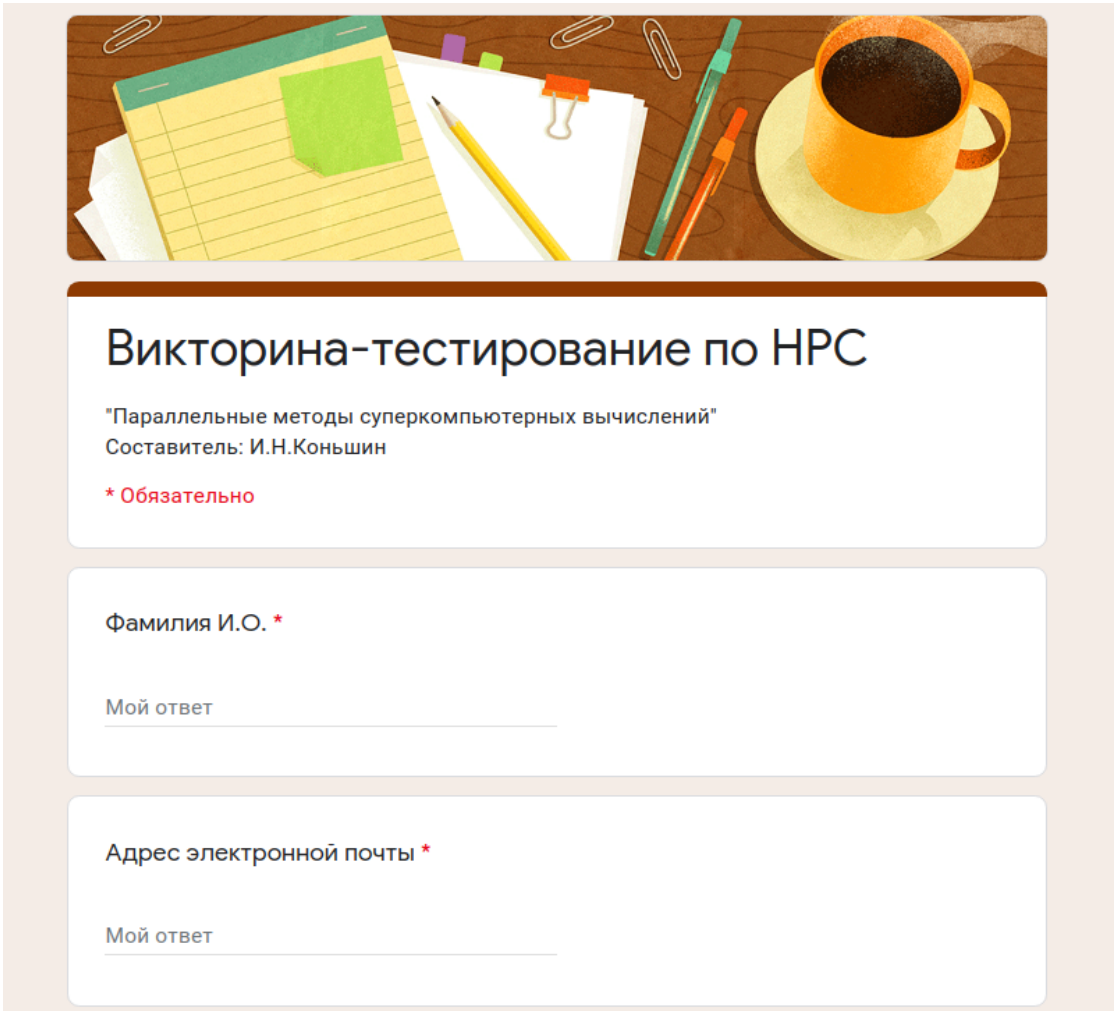






Задание 1 — викторина

- 10 вопросов: выбор от 1 до 4 вариантов ответа
- ...ссылка появится позже...



Задание 2 — теоретическая задача

- Цель теоретической задачи – оценить ускорения для конкретного алгоритма
- ...задачи появятся позже...

Задание mvm: $\vec{y} = A \cdot \vec{x}$ — анализ эфф-ти

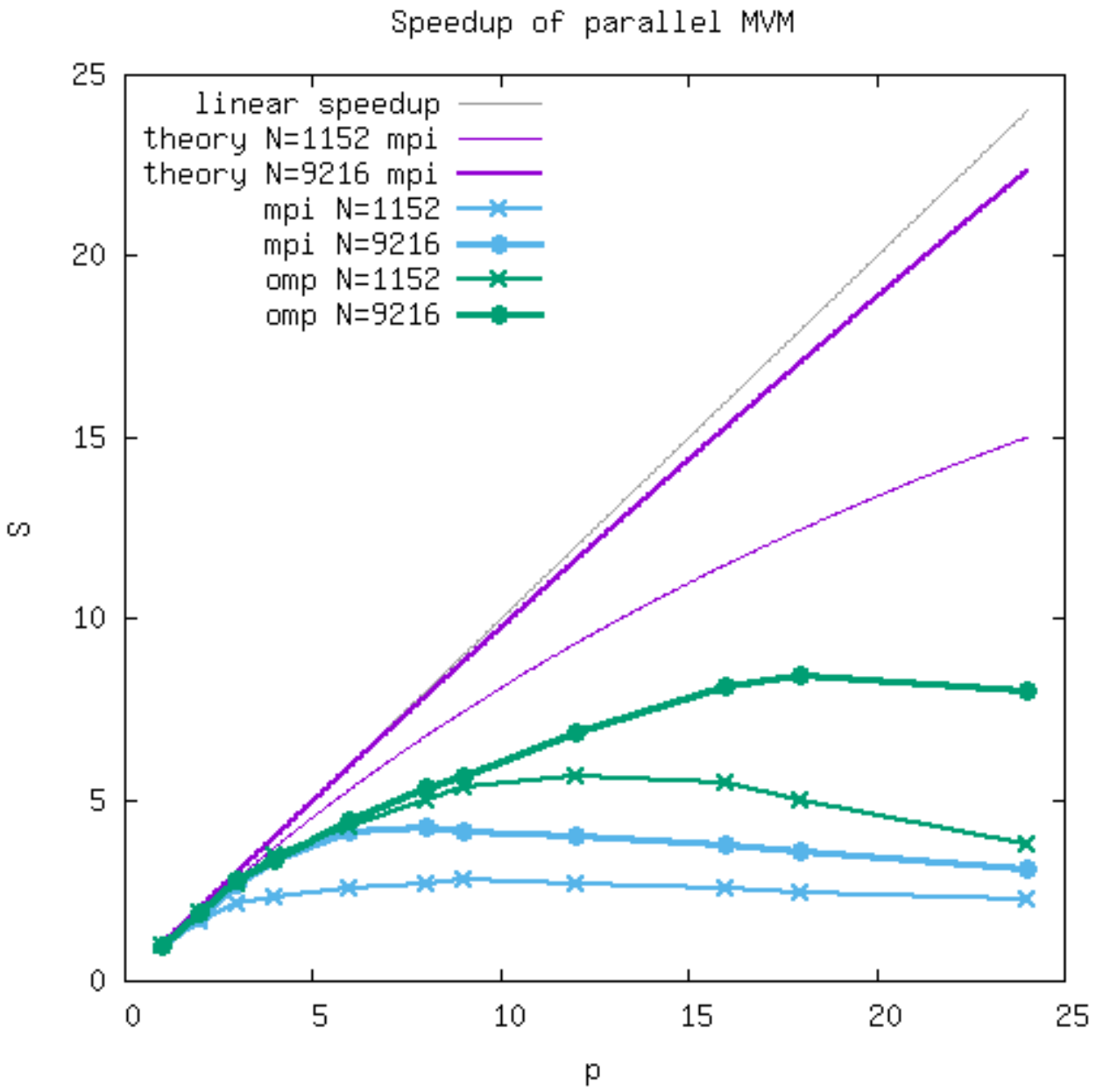
```
 y          A          x
[:]    [== == ==]    [:]
---    ----------    ---
[:] =  [== == ==] *  [:]
---    ----------    ---
[:]    [== == ==]    [:]
```

$$L_a = n^2/p$$

$$L_c = (n/p)(p-1)$$

$$L = L_c/L_a = (p-1)/n$$

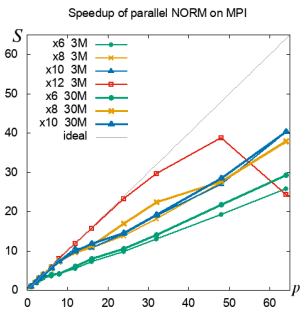
$$S = \frac{p}{1+\tau L} = \frac{p}{1+\tau\frac{p-1}{n}}$$

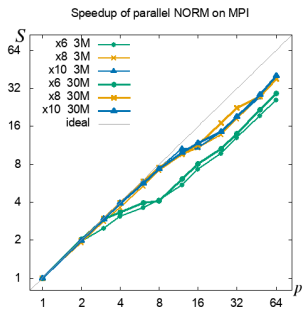


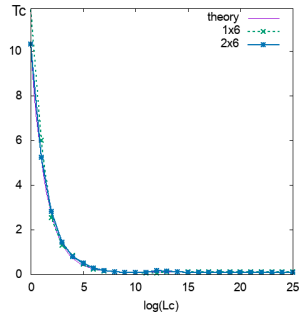
Задание 3 — практические задания [tasks_win.txt]

- требуется ***выбрать и запрограммировать*** один из предложенных алгоритмов (или представить свою собственную параллельную программу в рамках работы со своим научным руководителем, рассказать о параллельных методах, которые вы использовали);
- проверить ***корректность*** получаемого решения;
- замерить ***время счета*** на 1,2,4,8,16,32 процессорах и вычислить реальное ускорение;
- ***теоретически оценить*** ускорение через вычислительные/коммуникационные затраты и машиннозависимую характеристику τ (отношение скорости коммуникаций к скорости арифметики), которую нужно измерить самому в зависимости от используемых в программе операций;
- построить ***графики*** ускорения и сравнить теорию с практикой;
- написать краткий ***отчет*** о работе, указав адрес вашей программы на кластере.

(0) Тонкие характеристики обменов данными





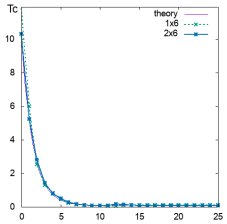


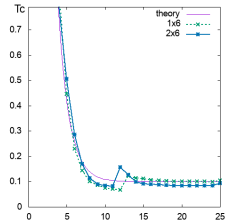
Исследовать зависимость времени инициализации обмена и скорости передачи данных от всех возможных параметров.

Зависимость от:

- **a** длины пересылки (см. [Байдин], тест 2),
- **b** типа обмена (синхронного/асинхронного, блокирующего/неблокирующего),
- **c** количества одновременных обменов,
- **d** количества процессоров.

(1) "Стереотипы" параллельного программирования

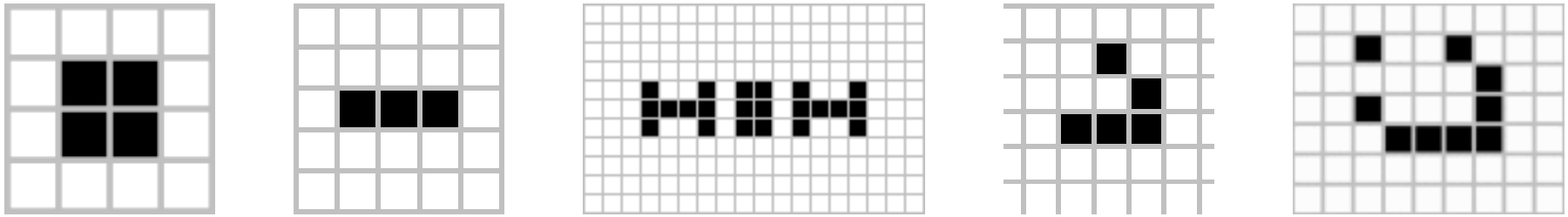




1. ***Выгодно ли проводить обмены на фоне вычислений?*** Сравнить время выполнения программы в случае блокирующих и неблокирующих обменов.
2. ***Как зависит скорость передачи сообщения от его длины?*** К чему стремится время передачи сообщения при уменьшении его длины (к τ_0)? При увеличении длины (к τ_c)? Есть ли максимум скорости где-то посередине (это зависит от реализации MPI)?
3. ***Что работает быстрее:*** коллективный обмен (например, `MPI_Bcast`) или последовательные блокирующие пересылки (например, через `MPI_Sendrecv`)?

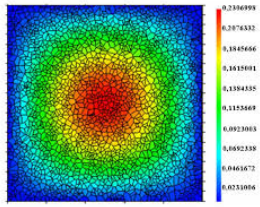
Г.В. Байдин, О некоторых стереотипах параллельного программирования, *ВАНТ, Матем. модел. физ. проц.*, 2008, No.1, 67-75. Bajdin-Stereotipy-2008.pdf (имеются фрагменты кодов программ)

(2) 2D игра “Жизнь”



- Случайное начальное распределение живых клеток.
- Периодические условия на границе с замыканием квадратной области в тор.
- Контроль стационарности/периодичности состояния.
- Исследовать несколько правил “оживания/умирания” клетки.
- Интересуют правила, дающие наиболее длительные вариации состояния поля.
- 2D разбиение исходного поля по процессорам.
- *Самые смелые могут попробовать реализовать 3D вариант игры* (см. ссылки на https://en.wikipedia.org/wiki/3D_Life).

(3) 3D ур-е теплопроводности: $\partial u/\partial t - a^2 \Delta u = f$



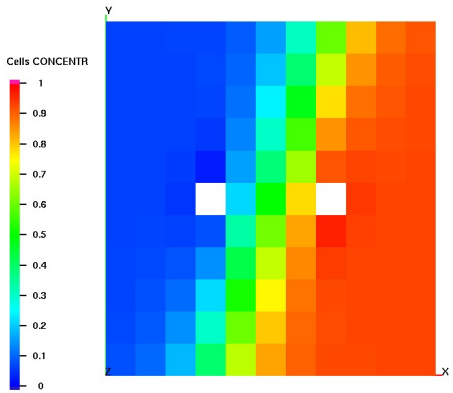
- Холодные стенки фиксированной температуры, в центре – постоянный источник тепла.
- Ввиду выполнения большинством из вас этой задачи в другом курсе, просьба рассмотреть оптимизированный вариант, когда передача данных соседям происходит не на каждом временном шаге, а через несколько (q) шагов (при этом обмены будут примерно в q раз длиннее, но зато в q раз реже – экономия на времени инициализации обменов).

(3a) 2D разбиение исходной 3D сетки по процессорам, оптимизированные обмены ($q > 1$).

(3b) Если не получилось отладить оптимизированные обмены, то сделать 3D разбиение исходной сетки по процессорам; обычные, неоптимизированные обмены ($q = 1$).

[Корнеев]: §3.4, стр. 49–53; §9.3, стр. 192–198

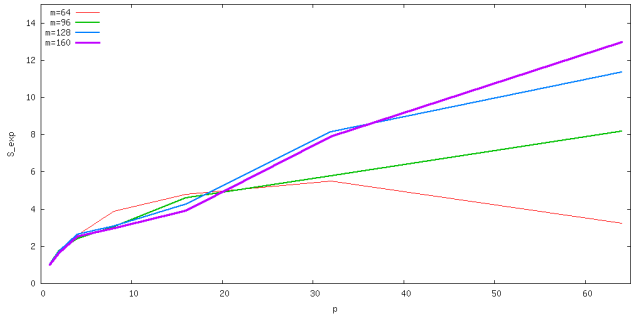
(4) 3D ур-е диффузии: $\partial c/\partial t = \operatorname{div}(D \operatorname{grad} c) + f$

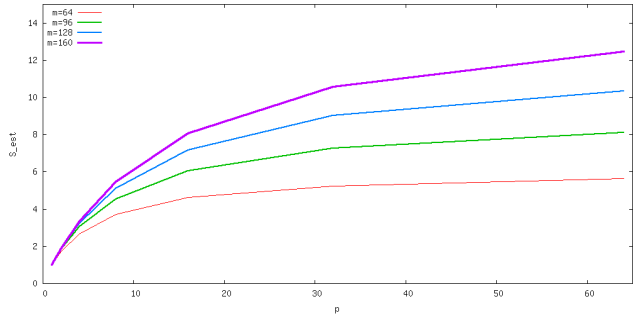


- 3D разбиение исходной сетки по процессорам.
- Один из вариантов, аналогичных (3a) или (3b).

[Корнеев]: §3.4, стр. 49–53; §9.3, стр. 192–198

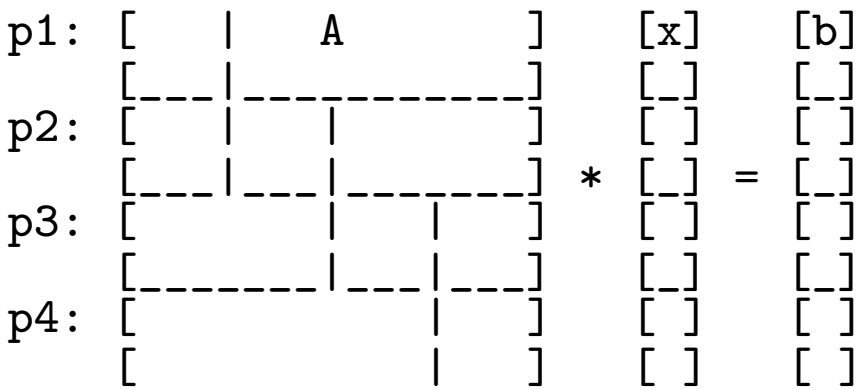
(5) Метод сопряженных градиентов для ур-я Пуассона





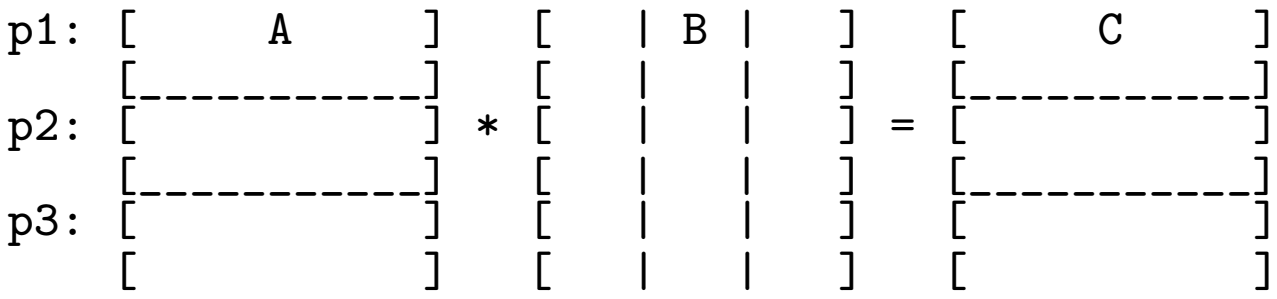
- 2D сетка узлов, 5-и точечный шаблон дискретизации (разреженная матрица с элементами: -1 -1 4 -1 -1);
- предобусловливатель без перекрытий (блочный метод Якоби);
- внутри процессора разложение IC0 (разложение в структуру исходной матрицы);
- точное решение $x^* = 1$;
- начальное приближение $x_0 = 0$;
- до точности 10^{-6}.

(6) М.С.Г. для системы с плотной матрицей: $Ax = b$



- Небольшое диагональное преобладание (симметричной!) матрицы;
- предобусловливатель без перекрытий (блочный метод Якоби);
- внутри процессора разложение Холецкого для своего центрального блока матрицы;
- точное решение $x^* = 1$;
- начальное приближение $x_0 = 0$;
- до точности 10^{-6}.

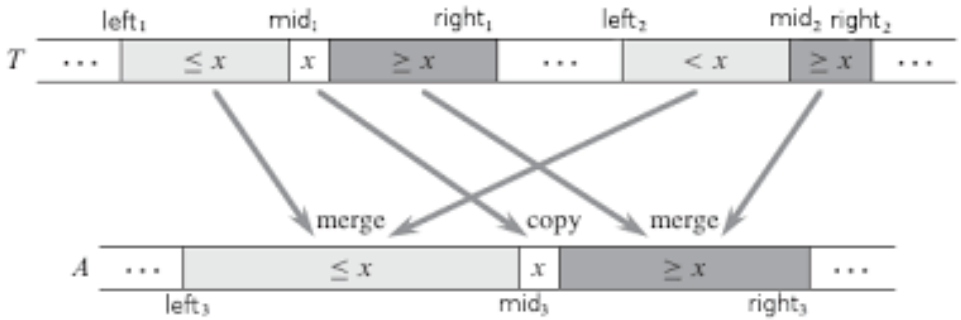
(7) Умножение двух плотных матриц: $A \cdot B = C$



- Плотные квадратные матрицы со случайными элементами.
- Строчно-столбцовое распределение данных по процессорам без дублирования начального и конечного распределения данных, например, A и C по блочным строкам, B – по столбцам.
- Дополнительно реализовать совместное MPI+OpenMP распараллеливание (см. `README.txt`, `mix_daxpy.c`, `qs_mix`) и для фиксированного общего количества ядер (например, 36 на normal) найти оптимальное соотношение процессов MPI и нитей OpenMP.

[Федотов]: §1.2.5, стр. 51–59
[Корнеев]: §2.1.1, стр. 14–18; §§9.2.1–9.2.3, стр. 178–192

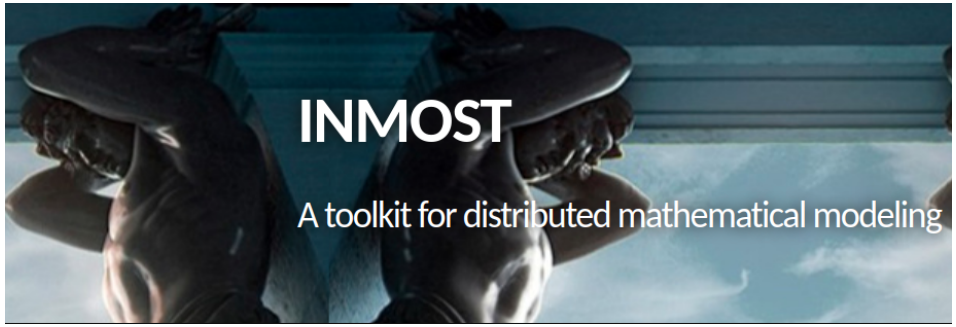
(8) Параллельная сортировка при ограниченной памяти



- Каждый процессор сортирует свою часть случайного вектора, после чего рассылает порции вектора на другие процессоры и собирает от других процессоров порции, лежащие в его интервале сортировки.
- Общая суммарная рабочая память – не более 3-х суммарных длин вектора.

[Тулебаев]: §3.4, стр. 49–53
[Корнеев]: §9.6, стр. 206–210
[М.В.Якобовский]

(9) INMOST



Integrated
Numerical
Modelling and
Object-oriented
Supercomputing
Technologies

- Примеры заданий для параллельной платформы INMOST могут быть предоставлены по запросу.
- *Только для желающих серьезно заниматься параллельными вычислениями, математическим моделированием и одновременно экспертов в C++.*

http://inmost.org

Ресурс параллелизма

В каждой из предложенных задач имеется достаточный ресурс параллелизма для получения хорошего ускорения:

(2-5) арифметические затраты пропорциональны количеству внутренних узлов, а обмены – количеству граничных узлов;

(6) арифметические затраты $\sim N^2$, а обмены $\sim N$;

(7) арифметические затраты $\sim N^3$, а обмены $\sim N^2$;

(8) арифметические затраты $\sim N \log N$, а обмены $\sim N$,

где N обозначает размерность вектора.

¡ Особенно, если размерность взять побольше !

Литература для практических заданий

- ***Г.В. Байдин***, О некоторых стереотипах параллельного программирования, *ВАНТ, Матем. модел. физ. проц.*, 2008, No.1, 67-75
  http://dodo.inm.ras.ru/konshin/HPC/bib/Bajdin-Stereotipy-2008.pdf
- ***С.Д. Тулебаев***, Параллельное программирование с использованием технологии MPI, 2010, 60 с.
  http://dodo.inm.ras.ru/konshin/HPC/bib/Tulebaev-ParProg.pdf
- ***И.Е. Федотов***, Некоторые приемы параллельного программирования, МГИРЭА, Москва, 2008, 188 с.
  http://dodo.inm.ras.ru/konshin/HPC/bib/Fedotov-Priemy.pdf
- ***В.Д. Корнеев***, Параллельное программирование в MPI, Новосибирск, 2002, 209 с.
  http://dodo.inm.ras.ru/konshin/HPC/bib/Korneev-ParProg.pdf

¡ Литература по теории приведена на страничке курса http://dodo.inm.ras.ru/konshin/hpc/ !

+) дополнительные лекции: [advanced_win.txt]

```
    ДОПОЛНИТЕЛЬНЫЕ ЛЕКЦИИ ПО КУРСУ
    "ПАРАЛЛЕЛЬНЫЕ МЕТОДЫ СУПЕРКОМПЬЮТЕРНЫХ ВЫЧИСЛЕНИЙ"

Вместо выполнения практического задания студентам разрешается
прочитать лекцию (~60 мин.) по своему выбору:

- дополнительные возможности OpenMP v.5.0 2018 vs. v.1.0 1997
- дополнительные возможности MPI v.3.1 2015 vs. v.1.0 1994
- распараллеливание по нитям средствами языка C++11
- распараллеливание на GPGPU с помощью Cuda
- распараллеливание посредством OpenCL
- распараллеливание посредством OpenACC
- распараллеливание на python
- ...предложите свою тему...

Необходимо:
- дать обзор возможностей выбранного средства распараллеливания
- привести примеры (фрагменты) программ
- желателен опыт работы
```

Литература для дополнительных лекций

¡ Дополнительная литература приведена на страничке курса
http://dodo.inm.ras.ru/konshin/hpc/ !

Решение задачи диффузии (командный конкурс)

$$\partial U/\partial t - \nabla(D \cdot \text{grad}\, U) = f(x, y, z, t)$$

$$U = U(x, y, z, t), \quad (x, y, z) \in \Omega = [0; 1]^3, \quad t \in [0; T], \quad T = 1$$

$U(x, y, z, t) = g(x, y, z)$ – граничные условия

$U(x, y, z, 0) = 0$ – начальные условия на $t_0 = 0$

$$D = \begin{bmatrix} d_x & 0 & 0 \\ 0 & d_y & 0 \\ 0 & 0 & d_z \end{bmatrix}, \qquad d_x = 0.25, \quad d_y = 0.15, \quad d_z = 0.1$$

$$g(x, y, z) = 0$$

$$f(x, y, z) = (d_x + d_y + d_z) \cdot \pi^2 \cdot \sin(\pi x) \sin(\pi y) \sin(\pi z)$$

$$U^* = \sin(\pi x) \sin(\pi y) \sin(\pi z) \cdot (1 - \exp(-(d_x + d_y + d_z) \cdot \pi^2\, t))$$

Решение задачи диффузии (командный конкурс)