Раздел I. Введение.

Тема 1. Коротко об операционной системе UNIX.

UNIX – это многозадачная, многопользовательская система, обладающая широкими возможностями. Её реализации существуют практически на всех распространенных компьютерных платформах.

LINUX – один из наиболее известных свободно распространяемых диалектов **UNIX**.

Тема 1. Коротко об операционной системе UNIX.

Все объекты в **UNIX** делятся на два типа:

файлы и *процессы*.

Все данные хранятся в файлах, доступ к периферийным устройствам осуществляется через специальные файлы.

Вся функциональность операционной системы определяется выполнением различных процессов.

Тема 1. Коротко об операционной системе UNIX.

Важнейшим пользовательским процессом является основной *командный интерпретатор (login shell).*

`passwd` - смена пароля

пароль должен хорошо запоминаться и быть трудным для подбора!

`exit` – выход из системы

`man` – получение справки о командах

`man man`

Тема 1. Коротко об операционной системе UNIX.

Идентификатор пользователя (**`UID`**), идентификаторы групп (**`GID`**).

Принадлежность к группе определяет дополнительные права пользователей.

Информация о пользователях и группах обычно хранится в системных файлах **`/etc/passwd`**, **`/etc/shadow`** и **`/etc/group`**

Тема 1. Коротко об операционной системе UNIX.

Файловая система, каталоги.

/ - корневой каталог

`/home/asa/myfile.txt`

`.` – текущий каталог

`..` – каталог на единицу более высокого уровня

С каждым пользователем ассоциируется его *домашний каталог*.

Тема 1. Коротко об операционной системе UNIX.

Атрибуты файлов. **`ls -l`**

```
   1          2  3     4     5     6       7     8
-rwxr-xr--  1 asa group 3422 Feb 28 13:30 test
```

`-` – обычный файл; **`d`** – каталог, **`l`** – ссылка и др.

Права доступа к файлу:

`-` – отсутствие права доступа, **`r`** – право на чтение, **`w`** – право на запись или удаление, **`x`** – право на выполнение файла.

Владелец-пользователь, владелец-группа и все остальные пользователи.

Тема 1. Коротко об операционной системе UNIX.

Смена прав доступа к файлу:

`chmod [u g o a][+ - =][r w x] file1…`

`u` – смена права доступа для пользователя,

`g` – для группы, **`o`** – для других пользователей,

`a` – для всех трёх категорий.

`+` – добавление соответствующего права,

`-` – удаление, а **`=`** – присвоение

`chmod g+w test`

`chown` и **`chgrp`** – смена владельца-пользователя и владельца-группы файла

Тема 1. Коротко об операционной системе UNIX.

- **`cd [dir]`** – переход в каталог **`dir`**

Если каталог не указан, то переход осуществляется в домашний каталог пользователя

- **`cp file1 file2`** – копирование файла
- **`mv file1 file2`** – перемещение (изменение имени) файла
- **`rm file1...`** – удаление файлов
- **`rmdir dir1...`** – удаление каталогов
- **`mkdir dir1...`** – создание каталога

Тема 1. Коротко об операционной системе UNIX.

- **pwd** – вывести имя текущего каталога
- **cat file, more file, less file** – утилиты просмотра содержимого файла
- **find dir** – поиск в файловой системе, начиная с каталога **dir**
- **grep <**рег_выражение**> file1...** – поиск в файлах вхождений регулярного выражения рег_выражение
- …

Тема 1. Коротко об операционной системе UNIX.

Процесс - программа в стадии её выполнения.

ps – список выполняющихся процессов

Уникальный идентификатор процесса (**PID**).

Сигналы.

Завершить выполнение процесса:

```
kill -9 PID
```

Список процессов, занимающих наибольшее количество процессорного времени или системных ресурсов:

```
top
```

Тема 1. Коротко об операционной системе UNIX.

Потоки ввода/вывода: стандартный ввод, стандартный вывод и стандартный вывод ошибок. Для перенаправления стандартного ввода можно использовать символ **<**, для стандартного вывода – **>** или **>>** (с добавлением), для потока ошибок – **2>**

```
program > file.log
```

Конвейер команд:

```
program1 | program2 | program3…
```

Фоновый режим:

```
program &
```

Тема 1. Коротко об операционной системе UNIX.

who – список пользователей, работающих в данный момент в системе

uname – некоторые сведения о системе

Редактирование файлов: **vi, joe** и др., встроенный редактор файлового менеджера **Midnight Commander (mc)**, удалённое редактирование.

time program - время работы программы

Тема 1. Коротко об операционной системе UNIX.

Компилятор с языка Си – **`cc`** (**`CC`** для Си++), компилятор с языка Фортран – **`f77`** (**`f90`** для Фортрана 90). Для программ на MPI используются скрипты **`mpicc`** (для программ на языке Си), **`mpiCC`** (Си++), и **`mpif77/mpif90`** (Фортран 77/90).

Опции компиляции:

`-o <`имя**`>`** – задание имени выполняемого файла (по умолчанию **`a.out`**)

`-O{`N**`}`** – уровень оптимизации

Тема 1. Коротко об операционной системе UNIX.

Задания:

- Войти в систему, поменять пароль.
- Просмотреть содержимое домашнего каталога и структуру файловой системы.
- Просмотреть наличие других пользователей в системе.
- Получить информацию обо всех выполняющихся процессах и отдельно о процессах данного пользователя.
- Написать, откомпилировать и выполнить программу, печатающую "Hello, world!"

Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

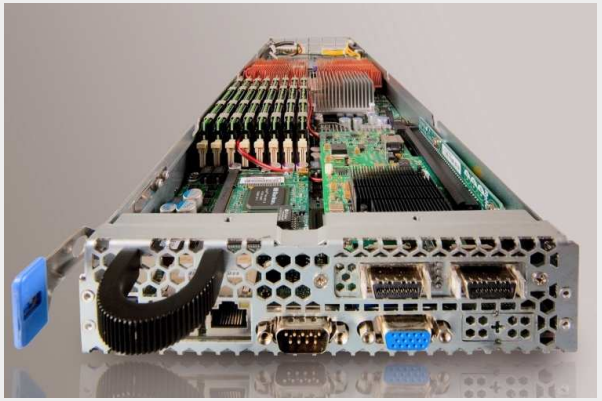
Суперкомпьютер СКИФ МГУ «Чебышёв»

Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

- Пиковая производительность: 60 TFlop/s
- Производительность на Linpack: 47.32 TFlop/s (79% пиковой), матрица 740000x740000
- 625 вычислительных узлов, 1250 процессоров, 5000 процессорных ядер
- 42 стойки: 14 вычислительных, 28 инфраструктурных
- Помещение 98 $м^2$
- Общий вес оборудования: более 30 тонн

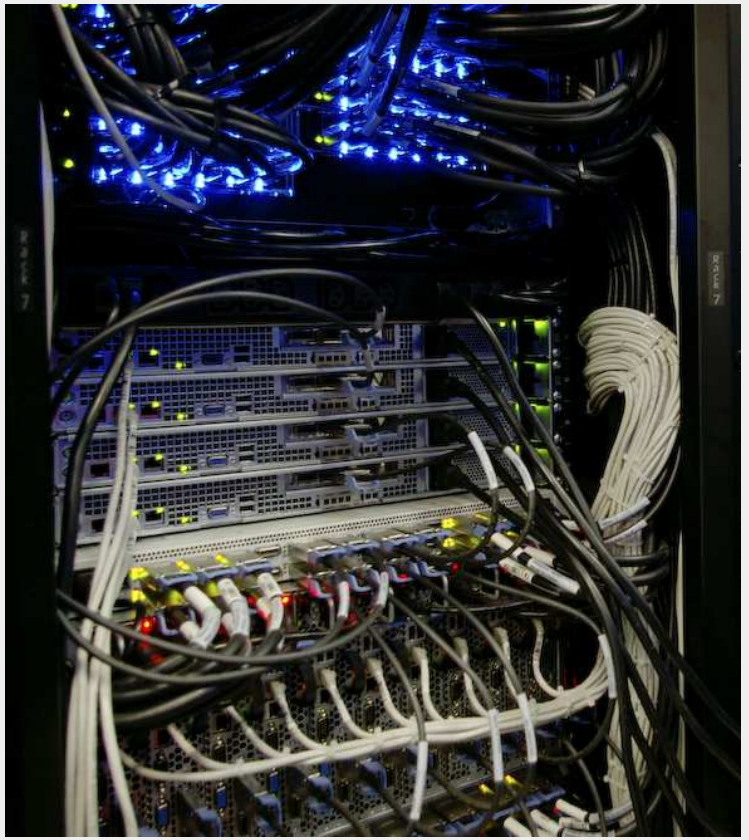
Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

- **Процессоры:**
  - 1250 Intel E5472 3.0 ГГц Harpertown
- **Блэйд-шасси:** T-Blade («Т-Платформы»), форм-фактор 5 U, до 10 вычислительных узлов
- **Оперативная память:**
  - 529 x 8 ГБ, бездисковые
  - 64 x 8 ГБ, 160 ГБ HDD
  - 32 x 16 ГБ, 160 ГБ HDD
  - 8 x 32 ГБ, 160 ГБ HDD



Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

- **DDR InfiniBand**
  - Mellanox MT25418 NIC
  - FatTree
  - SilverStorm 9120 – базовые коммутаторы
  - Flextronix F-X430046 – листовые коммутаторы
- **Характеристики**
  - 1.3 – 1.95 μs латентность
  - 1.7 ГБ/с пропускная способность



Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

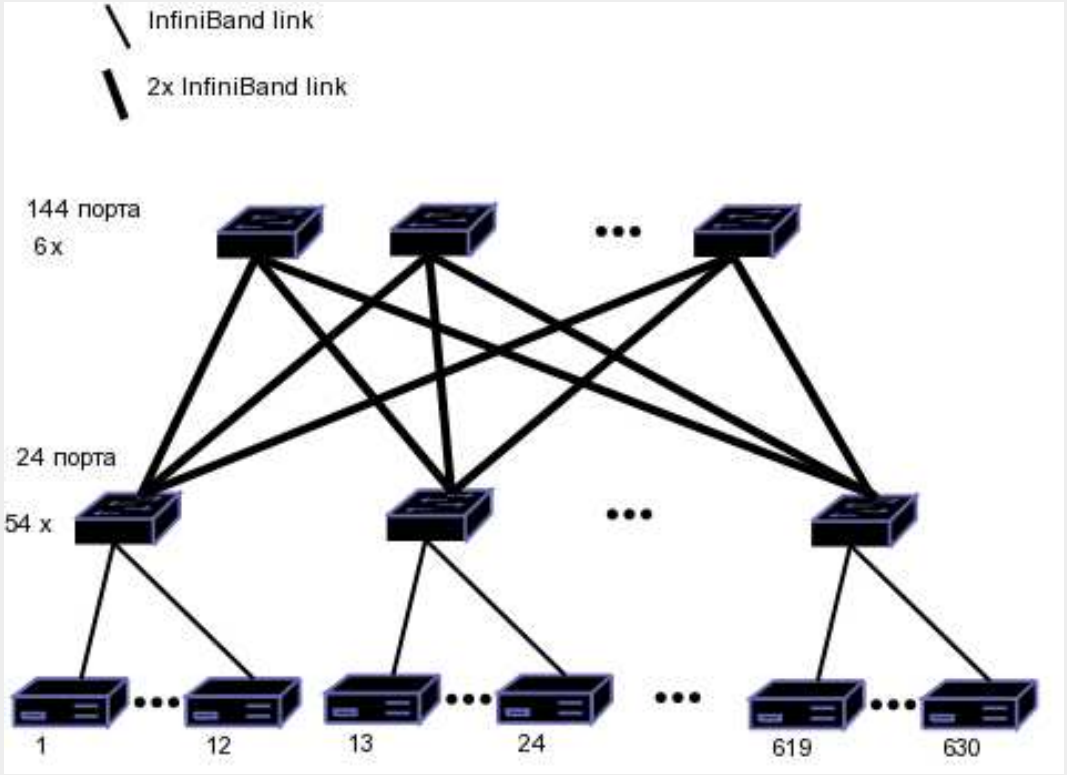


Схема построения Fat Tree в СКИФ МГУ «Чебышёв»

Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

- **Вспомогательные сети:**
  - Gigabit Ethernet: коммутаторы Force10 C300 и Force10 S2410
  - Управляющая сеть ServNet + IPMI
- **Хранилище данных:**
  - 60 ТБ распределённое отказоустойчивое сетевое хранилище T-Platforms ReadyStorage ActiveScale Cluster
  - 15 ТБ локальных дисков на узлах
  - Ленточное хранилище Quantum Scalar i500

Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

- **Операционная система**
  - ALT Linux HPC
- **Параллельная среда**
  - mvapich
- **Система управления**
  - Cleo
- **Разработка программ**
  - Компиляторы GCC, Intel, PGI, PathScale
  - Intel Cluster and Development Toolkit

Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

- **Вход на головную машину**

  Вход по **ssh** версии 2 на адрес **skif-mgu.parallel.ru**, IP-адрес: **212.192.244.31**

  ssh, SSH Secure Shell Client, PuTTY, Teraterm

  На вычислительные узлы вход запрещён.

  Передача файлов: протокол **SFTP**

- **Хранение файлов**

  /home/<имя пользователя> - на системе хранения данных, доступны по сети на всех узлах, ограничено квотой.

Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

- **Компиляторы**
  - Intel Compilers 11.0. (C,C++,Fortran77/90,95) Команды: **`icc`**, **`ifort`**
  - Portland Group Inc. Compilers 7.2-3 (C,C++,Fortran77/90,95) Команды: **`pgcc`**, **`pgCC`**, **`pgf77`**, **`pgf90`**, **`pgf95`**
  - PathScale Compiler Suite: Version 3.2 (C,C++,Fortran90/95) Команды: **`pathcc`**, **`pathCC`**, **`pathf90`**, **`pathf95`**
  - GNU 4.1.2 (C,C++,Fortran) Команды: **`gcc`**, **`gfortran`**

Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

- **Компиляция программ**

  команды **`mpicc/mpicxx`** (C и C++) и **`mpif77/mpif90`**

  (Фортран 77/90). Автоматически подключают заголовочные файлы и библиотеки MPI.

- **Выбор компилятора и реализации MPI**

  утилита **`mpi-selector`**:

  `--list`

  `--set <name>`

  По умолчанию - компилятор Intel и mvapich

Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

- **Опции компиляции**

  `-o <имя>`

  `-O3`

  при компиляции **`mpicc/mpicxx`** и **`mpif77/mpif90`** не должны использоваться опции **`-static`** и **`-fast`**

- **Компиляция с OpenMP**

  gcc/gfortran: **`-fopenmp`**

  icc/ifort: **`-openmp`**

  pgcc/pgCC/pgf77/pgf90: **`-mp`**

  `export OMP_NUM_THREADS=8`

Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

- **Запуск приложений (Cleo)**

  `mpirun -np N program <параметры программы>`

  `mpirun -np 1 -as single program.e`

  `-q <очередь>` –название очереди

  `-maxtime <время>` - максимальное время работы задачи в минутах

- **Просмотр состояния задачи**

  `tasks`

  `-q <очередь>`

  `-l` просмотр расширенной информации о задачах

Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

Очереди на СКИФ МГУ «ЧЕБЫШЁВ»

`regular` (4152 ядра) - без локальных дисков, 8 ГБ ОП,

`hdd` (520 ядер) - с локальными дисками, 8 ГБ ОП,

`hddmem` (256 ядер) - с локальными дисками, 16 ГБ ОП,

`bigmem` (64 ядра) - с локальными дисками, 64 ГБ ОП,

`test` (80 ядер) - без локальных дисков, 8 ГБ ОП.

Для очереди **`test`** лимит времени на одну задачу – 15 минут. Разрешено не более 10 задач одного пользователя в очереди, включая запущенные.

Тема 2. Краткие сведения о целевой высокопроизводительной вычислительной системе.

- **Просмотр результатов**

  По окончании работы - сообщение на терминал.

  В рабочей директории создаются файлы:

  <задача>.out-<номер> и **<задача>.rep-<номер>**

- **Удаление задачи**

```
tasks [-q <очередь>] -d ID
tasks [-q <очередь>] -d all
```

Тема 3. Параллелизм и его использование.

Конвейерность и *параллельность*.

Параллельная обработка. Одна операция - за единицу времени, то 1000 - за тысячу единиц. Пять устройств 1000 операций выполнит за 200 единиц времени. N устройств ту же работу выполнит примерно за 1000/N единиц времени.

Тема 3. Параллелизм и его использование.

Конвейерная обработка. Операция разбивается на ряд подопераций, выполняемых последовательно и независимо. Пусть 5 микроопераций, каждая из которых выполняется за единицу времени. Последовательное устройство 100 пар аргументов обработает за 500 единиц. Конвейерное устройство: первый результат через 5 единиц времени, каждый следующий – через одну единицу после предыдущего, а весь набор из ста пар будет обработан за 5+99=104 единицы времени.

Тема 3. Параллелизм и его использование.

Задача распараллеливания программы обычно сводится к нахождению в ней достаточного количества информационно независимых операций, распределению их между вычислительными устройствами, обеспечению синхронизации и необходимых коммуникаций.

Две операции программы называются *информационно зависимыми*, если результат выполнения одной операции используется в качестве аргумента в другой.

Тема 3. Параллелизм и его использование.

Если операция B информационно зависит от операции A (то есть, использует какие-то результаты операции A в качестве своих аргументов), то операция B может быть выполнена только по завершении операции A.

Если операции A и B не являются информационно зависимыми, то алгоритмом не накладывается никаких ограничений на порядок их выполнения, в частности, они могут быть выполнены одновременно.

Графовые модели программ.

Тема 3. Параллелизм и его использование.

Крупноблочное распараллеливание:

```
    if (MyProc == 0){
/* операции, выполняемые 0-ым процессом */
    }
    ...
    if (MyProc == K){
/* операции, выполняемые K-ым процессом */
    }
```

Тема 3. Параллелизм и его использование.

Наибольший ресурс параллелизма в программах сосредоточен в циклах!

Распределение итераций циклов:

```
    for (i=0; i<N; i++)
        if (i ~ MyProc) {
/* операции i-й итерации цикла для
 для выполнения процессом MyProc */
        }
    }
```

Тема 3. Параллелизм и его использование.

Примеры способов распределения итераций циклов:

- *Блочное распределение* – по $\lceil N/P \rceil$ итераций.

- *Блочно-циклическое распределение* – размер блока меньше, распределение продолжается циклически.

- *Циклическое распределение* – циклически по одной итерации.

Тема 3. Параллелизм и его использование.

Рассмотрим простейший цикл:

```
for (i=0; i<N; i++)
     a[i] = a[i] + b[i];
```

Тема 3. Параллелизм и его использование.

Блочное распределение:

```
/* размер блока итераций */
k = (N-1)/P + 1;
/* начало блока итераций
 процесса MyProc */
ibeg = MyProc * k;
/* конец блока итераций
 процесса MyProc */
iend = (MyProc + 1) * k;
```

Тема 3. Параллелизм и его использование.

```
/* если не досталось итераций */
if (ibeg >= N)
   iend = ibeg – 1;
else
/* если досталось меньше итераций */
   if (iend >= N) iend = N;
for (i=ibeg; i<iend; i++)
   a[i] = a[i] + b[i];
```

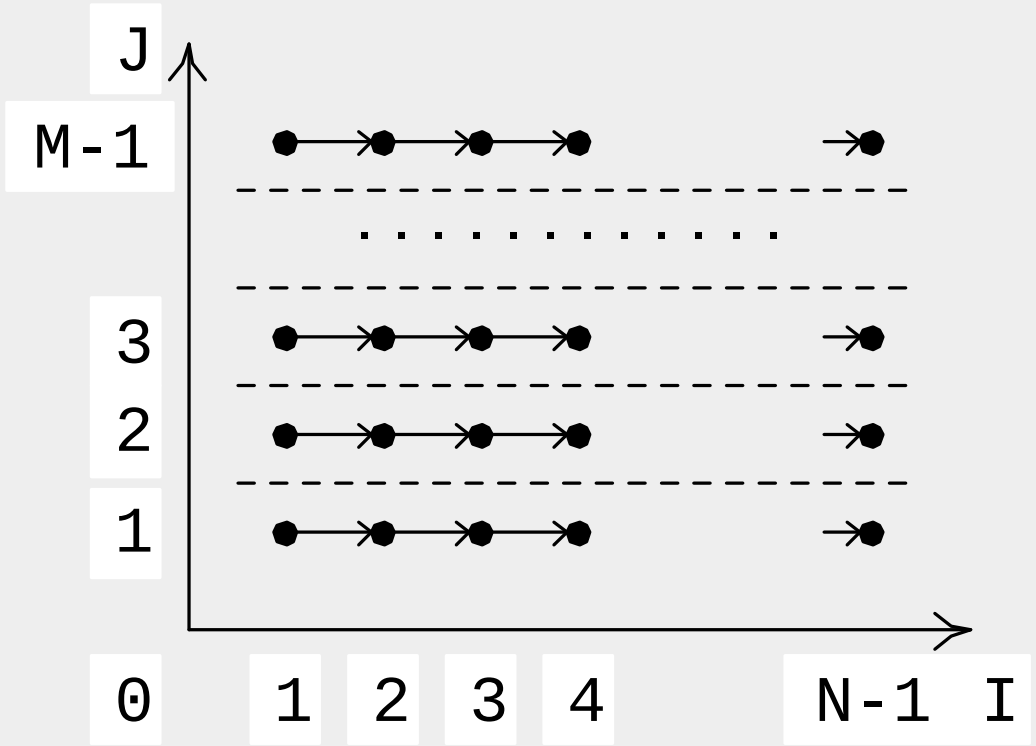
Тема 3. Параллелизм и его использование.

Циклическое распределение:

```
for (i=MyProc; i<N; i+=P)
   a[i] = a[i] + b[i];
```

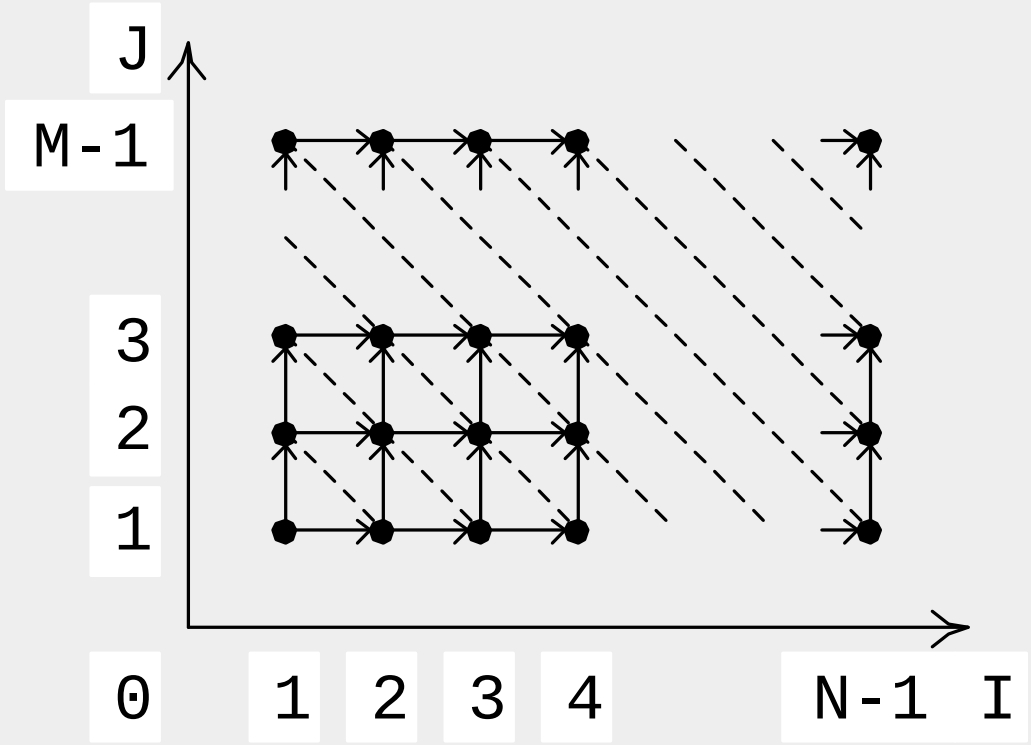
Тема 3. Параллелизм и его использование.

```
for (i=1; i<N; i++)
  for (j=1; j<M; j++)
    a[i][j] = a[i-1][j] + a[i][j];
```



Тема 3. Параллелизм и его использование.

```
for (i=1; i<N; i++)
 for (j=1; j<M; j++)
     a[i][j] = a[i-1][j] + a[i][j-1];
```



Тема 3. Параллелизм и его использование.

Цели распараллеливания:

- *равномерная загрузка процессоров*
- *минимизация количества и объема необходимых пересылок данных*

Пересылка данных требуется, если есть информационная зависимость между операциями, которые при выбранной схеме распределения попадают на разные процессоры.

Тема 3. Параллелизм и его использование.

Закон Амдала:

Пусть **f** – доля последовательных операций,

0 ≤ f ≤ 1,

1-f – доля параллельных операций,

S – ускорение, **p** – число процессоров

$$\frac{T_1}{T_p} = S \le \frac{1}{f + \frac{1-f}{p}}$$