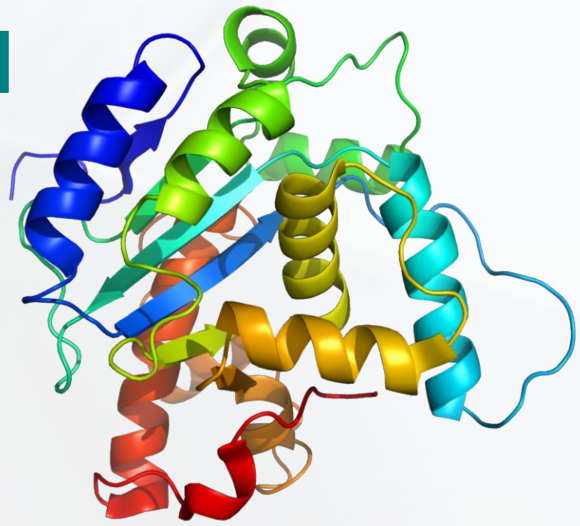
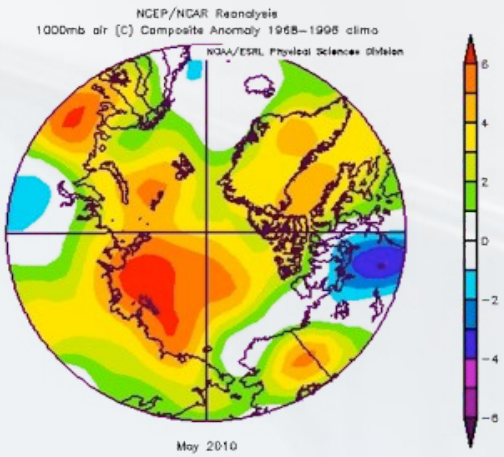
Параллельное программирование

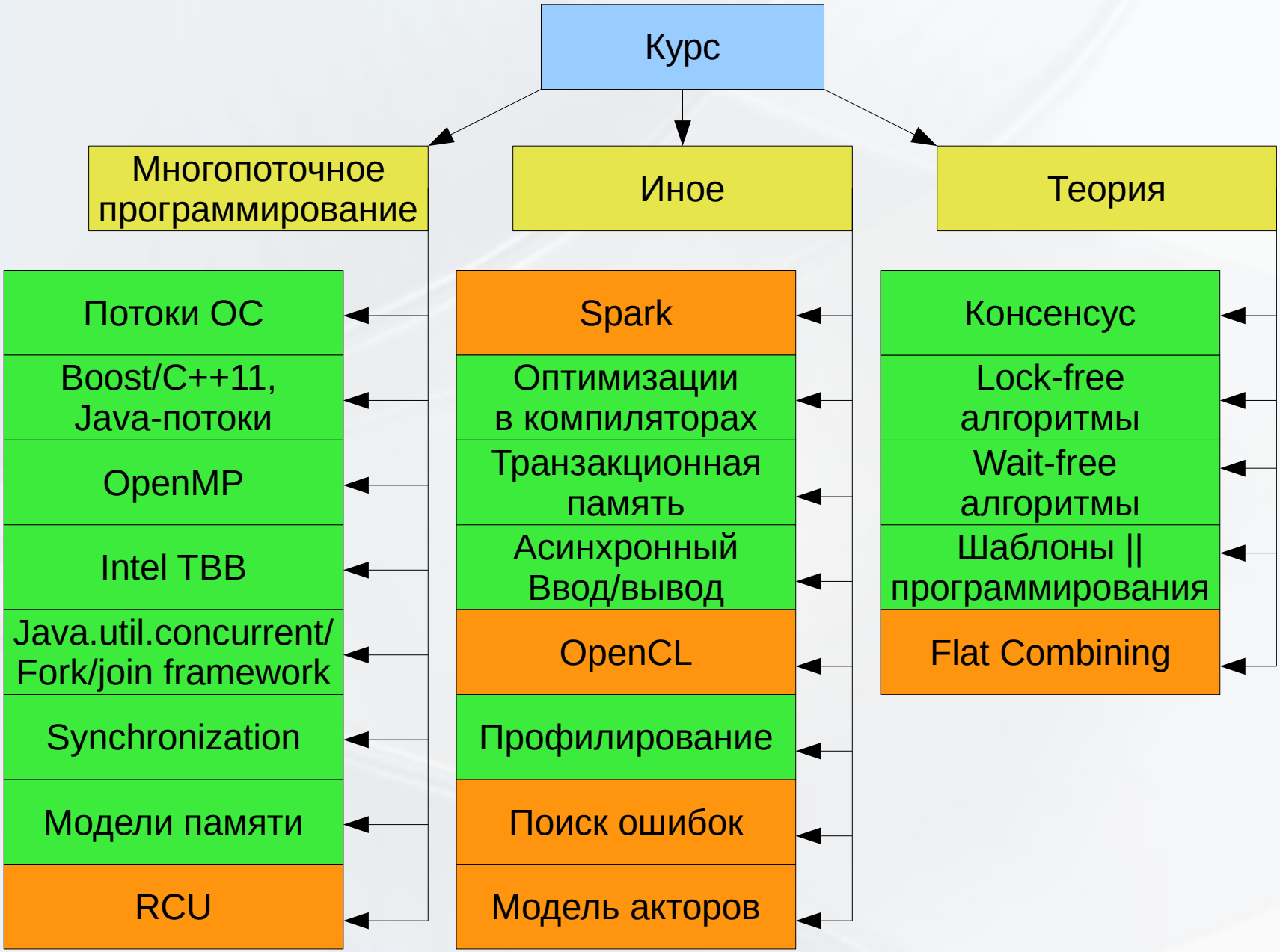
Калишенко Е.Л.
2022

Мотивация



- Генетика и протеомика
- Климатология
- Физика высоких энергий
- Астрономия, банковские транзакции...





http://wiki.osll.ru/doku.php/courses:high_performance_computing

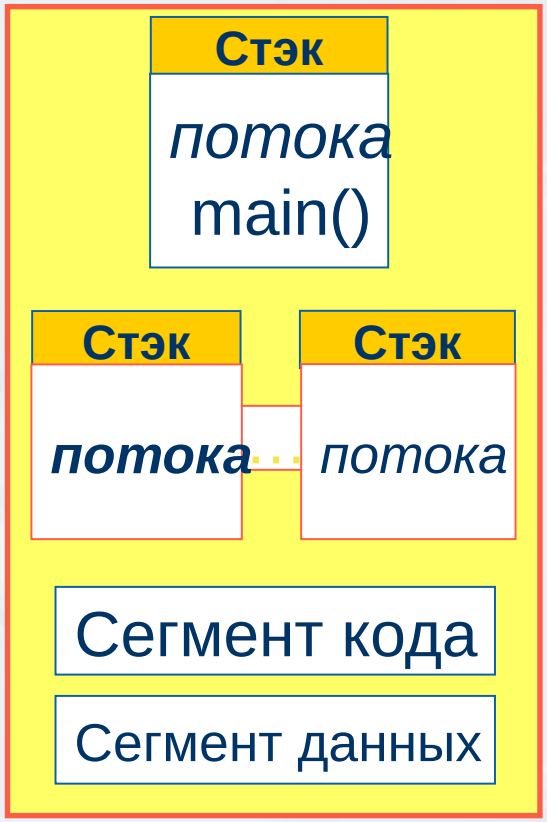
SSE (Streaming SIMD Extensions)

Версия	Возможности
1	Восемь 128-битных регистров для 4 чисел по 32 бит (с плавающей точкой)
2	Теперь 2 64-битных числа в регистре
3	Уже 13 инструкций
4.1	47 инструкций (ускорение видео)
4.2	54 инструкции (операции со строками)

```
float a[4] = { 300.0, 4.0, 4.0, 12.0 };
float b[4] = {   1.5, 2.5, 3.5,  4.5 };

__asm {
    movups xmm0, a    ; // поместить из a в регистр xmm0
    movups xmm1, b    ; // поместить из b в регистр xmm1
    mulps xmm0, xmm1  ; // перемножить пакеты плавающих точек
    movups a, xmm0    ; // выгрузить результаты из регистра xmm0 по адресам a
};
```

Процессы и потоки



- В начале выполнения процесс — 1 поток
- Потоки могут создавать новые в пределах одного процесса
- Все потоки имеют общие сегменты кода и данных
- Каждый поток имеет свой стек выполнения
- *Единица планирования ОС - поток*

Потоки ОС

Операция	Posix
Создание	pthread_create()
Ожидание завершения	pthread_join()
Захват мьютекса	pthread_mutex_lock()
Освобождение мьютекса	pthread_mutex_unlock()

```
int pthread_create(pthread_t *thread,
                   const pthread_attr_t *attr,
                   void *(*start_routine)(void*),
                   void *arg);
```

Закон Амдала

- a — доля последовательного кода
- p — число процессоров

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

Java

- Наследование от Thread
- Реализация Runnable
- Остальное привычно: start(), join()

```
public class IntegrateRunnable implements Runnable {

        public IntegrateTask task;

        @Override
        public void run() {
                task.res = 0;
            for (double x = task.from; x < task.to - 1E-13*task.to; x += task.step) {
                task.res += task.f(x) * task.step;
            }
        }
}
```