Казанский федеральный
УНИВЕРСИТЕТ
ВЫСШАЯ ШКОЛА
информационных технологий
и информационных систем

# Реализация алгоритмов с использованием MPI

Эдуард Храмченков

# Пример 1

- Вычисление числа π с помощью метода Монте-Карло
- Генерация случайных точек и проверка сколько из них попадают в окружность радиуса 1

$$\pi = 4\frac{count}{n}$$

- Точность зависит от числа испытаний и выбранного ГПСЧ

# Пример 1

- Самый очевидный из параллельных вариантов – каждый процесс проводит свою часть испытаний
- Для сбора количества попаданий в окружность используется редукция с суммированием
- За счет параллелизации цикла и небольшого числа коммуникаций ускорение близко к линейному

# Пример 2

- Решето Эратосфена – алгоритм поиска простых чисел $\leq n$
- Перебор массива с маркировкой элементов
- Очевидная оптимизация – поиск только в нечетных числах
- Оптимизация с сокращением предела внешнего цикла до $\sqrt{n}$ не дает ощутимого эффекта за счет возникновения дополнительного цикла

# Пример 2

- Параллелизация по данным
- Разбиение массива чисел от 2 до *n* на несколько частей(блоков), каждая часть обрабатывается своим процессом
- Одинаковый размер блоков для каждого из процессов дает лучшую производительность
- Последовательно берем каждое простое число *prime* и отмечаем *2*prime*, *3*prime*… как составные

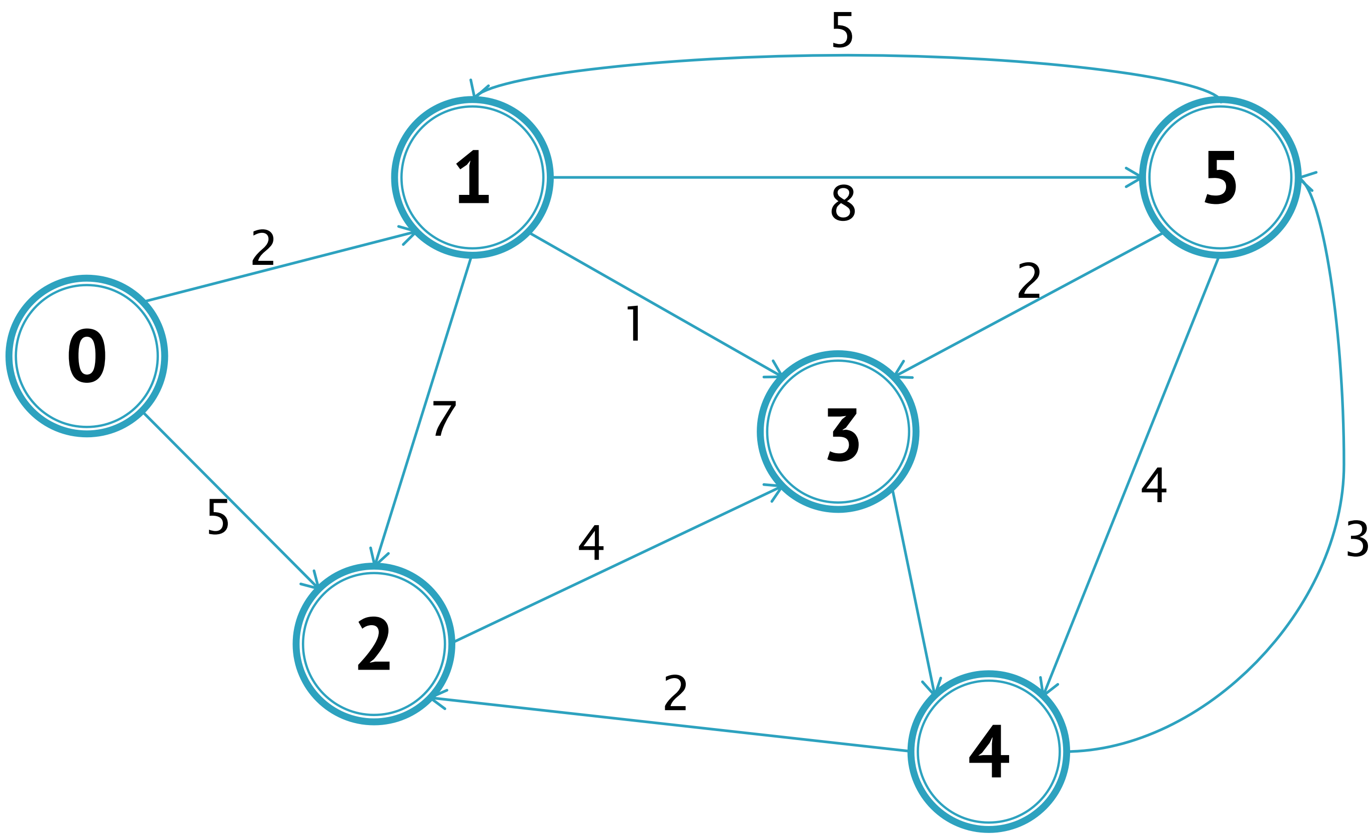
# Пример 2

- Начало и конец диапазона чисел для каждого потока регулируется величинами *low_value* и *high_value*
- Булевый массив помеченных составных чисел у каждого процесса свой
- Соответствие между индексами этих массивов и реальными числами вычисляются при помощи *prime*, *low_value* и *high_value*

# Пример 3

- Задача– построение таблицы кратчайших маршрутов из пункта А в пункт В
- Карту пунктов можно представить как взвешенный ориентированный граф
- Алгоритм Флойда – динамический алгоритм для нахождения кратчайших расстояний между всеми парами вершин взвешенного ориентированного графа

# Пример 3



# Пример 3

## Матрица смежности

| | **0** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|---|
| **0** | 0 | 2 | 5 | ∞ | ∞ | ∞ |
| **1** | ∞ | 0 | 7 | 1 | ∞ | 8 |
| **2** | ∞ | ∞ | 0 | 4 | ∞ | ∞ |
| **3** | ∞ | ∞ | ∞ | 0 | 3 | ∞ |
| **4** | ∞ | ∞ | 2 | ∞ | 0 | 3 |
| **5** | ∞ | 5 | ∞ | 2 | 4 | 0 |

# Пример 3

## Решение задачи

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| **0** | 0 | 2 | 5 | 3 | 6 | 9 |
| **1** | ∞ | 0 | 6 | 1 | 4 | 7 |
| **2** | ∞ | 15 | 0 | 4 | 7 | 10 |
| **3** | ∞ | 11 | 5 | 0 | 3 | 6 |
| **4** | ∞ | 8 | 2 | 5 | 0 | 3 |
| **5** | ∞ | 5 | 6 | 2 | 4 | 0 |

# Пример 3

## Решение задачи

| | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| **0** | 0 | 2 | 5 | 3 | 6 | 9 |
| **1** | ∞ | 0 | 6 | 1 | 4 | 7 |
| **2** | ∞ | 15 | 0 | 4 | 7 | 10 |
| **3** | ∞ | 11 | 5 | 0 | 3 | 6 |
| **4** | ∞ | 8 | 2 | 5 | 0 | 3 |
| **5** | ∞ | 5 | 6 | 2 | 4 | 0 |

# Пример 3

- Параллелизация по данным
- Каждое обновление элемента *a[i][j]* требует доступа к элементам *a[i][k]* и *a[k][j]*
- На *k*-ой итерации каждый элемент в *k*-ом строке/столбце транслируется всем процессам обрабатывающим данные в строке/столбце
- Есть возможность для каждой итерации внешнего цикла *k* обновлять элементы матрицы параллельно

# Пример 4

- Перемножение матриц
- Параллелизация по данным – каждый процесс обрабатывает свои столбцы/строки
- Возможные модификации – блочные алгоритмы Фокса и Кэннона
- Требуют создания карты блоков – более сложные в реализации
- Блочные алгоритмы более эффективны на очень больших матрицах

# Пример 5

- Сортировка больших массивов
- Массив распределяется по процессам
- Каждый из меньших массивов сортируется последовательно
- Затем пошаговое слияние подмассивов
- Большие расходы на изменение размеров массивов и передачу данных
- Возможные оптимизация – паралеллизация алгоритма слияния

Казанский федеральный УНИВЕРСИТЕТ
ВЫСШАЯ ШКОЛА информационных технологий и информационных систем

# Вопросы

ekhramch@kpfu.ru