Отображение графовых задач на архитектуру графических ускорителей – теория и практика

Колганов А.С.

План доклада

- Обзор параллельных архитектур
- Проблемы обработки графов
- Пример оптимизации задачи SSSP и MST
- Заключение

Параллельные архитектуры

- Графические ускорители (GPU);
- Многоядерные процессоры x86 (CPU);
- Сопроцессоры Intel Xeon Phi (MIC);
- ПЛИС (FPGA);
- другие.

Параллельные архитектуры

- Графические ускорители (GPU);
  - Большая вычислительная мощность ~ 10 TFLOPS
  - Большая пропускная способность памяти ~ 0,3–1 TB/s
  - Мало быстрой памяти ~ 32 GB
  - Узкий канал связи с CPU ~ 12–50 GB/s
- Многоядерные процессоры x86 (CPU);
- Сопроцессоры Intel Xeon Phi (MIC);

Параллельные архитектуры

- Графические ускорители (GPU);
- **Многоядерные процессоры x86 (CPU);**
  - Большое количество памяти ~ 6 TB в одном узле
  - Возможность объединения 4 CPU на общей памяти
  - Низкая вычислительная мощность ~ 0,5 TFLOPS / CPU
- Сопроцессоры Intel Xeon Phi (MIC);

Параллельные архитектуры

- Графические ускорители (GPU);
- Многоядерные процессоры x86 (CPU);
- **Сопроцессоры Intel Xeon Phi (MIC);**
  - Быстрая внутренняя память (GDDR5) и медленная внешняя (DDR4)
  - Средняя вычислительная мощность ~ 3 TFLOPS
  - Необходимость векторизации кода

Тенденции развития

- Один большой узел с общей памятью;
- Многоуровневая память: мало быстрой (GDDR5 или HBM2) и много медленной (DDR4);
- Большая вычислительная мощность:
  - много ядер (Xeon Phi);
  - много потоков (GPU Nvidia/ AMD);
  - векторизация (1024 битные вектора Xeon Phi);
  - быстрая связь с соседними устройствами (например, NVlink).

Проблемы обработки графов

- Косвенная индексация;
- Произвольный доступ в память;
- Различные виды графов;

Проблемы обработки графов

- Косвенная индексация;
- Произвольный доступ в память;
- Различные виды графов;

- Что приводит к:
  - Промахам в кэш;
  - Невозможности векторизации;
  - Необходимости изменять даже самые простые алгоритмы (BFS, SSSP, MST) для достижения высокой производительности.

R-MAT и SSCA2 графы

- RMAT – A Recursive Model for Graph Mining;
  - Одна большая связная компонента и небольшое количество маленьких.
- SSCA2 – HPCS Scalable Synthetic Compact Applications graph analysis;
  - Набор небольших сильно связных компонент.

Задачи BFS, SSSP и MST

- BFS – Breadth-first search (тест для Graph500);
- SSSP – Single Shortest Path Problem;
- MST – Minimum Spanning Tree;
- Community Detection;
- …..

- Характеристики:
  - Простые фундаментальные алгоритмы;
  - Совсем нет вычислений;
  - Скорость обработки напрямую зависит от скорости случайного доступа в память.

Оптимизации обработки графов на GPU – подходы

- Переупорядочивание данных при неизменном алгоритме и/или использование другой структуры хранения данных;
- Переписывание алгоритма при неизменной структуре данных;
- Использование как переупорядочивания, так и другого алгоритма (отличного от исходного);
- Использование сжатия данных;
- И др.

Варианты представления графа (разреженной матрицы)

- Coordinate Format (COO);
- Compressed Sparse Row Format (CSR);
- Compressed Sparse Column Format (CSC);
- Ellpack-Itpack Format (ELL);
- Hybrid Format (HYB) (ELL + COO);
- Block Compressed Sparse Row Format (BSR);

Варианты представления графов – выводы

- Нет единого и удобного представления (хранения) графов (разреженных матрицы);
- В каждой библиотеке (или программе) могут использоваться свой набор форматов (cuSparse, MKL);
- Для каждой задачи может быть удобно то или иное представление.

Практическое решение гравфовых задач на GPU

- SSSP – переупорядочивание данных при неизменном алгоритме;
- MST – сильное изменение алгоритма и небольшое преобразование данных для достижения высокой производительности.
- BFS – и переупорядочивание данных и изменение алгоритма.

Алгоритм Форда–Беллмана для SSSP

SSSP – поиск кратчайшего пути

```
if(k < maxV)
{
    double w = weights[k];
    unsigned en = endV[k];
    unsigned st = startV[k];

    if(dist[st] > dist[en] + w)
        dist[st] = dist[en] + w;

    else if(dist[en] > dist[st] + w)
        dist[en] = dist[st] + w;
}
```

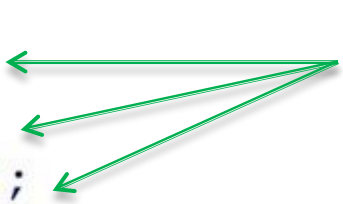
Алгоритм Форда–Беллмана для SSSP

SSSP – поиск кратчайшего пути

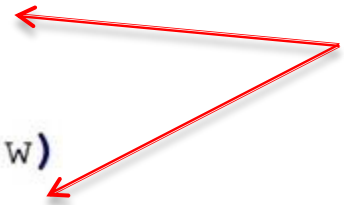
```
if(k < maxV)
{
    double w = weights[k];
    unsigned en = endV[k];
    unsigned st = startV[k];

    if(dist[st] > dist[en] + w)
        dist[st] = dist[en] + w;

    else if(dist[en] > dist[st] + w)
        dist[en] = dist[st] + w;
}
```

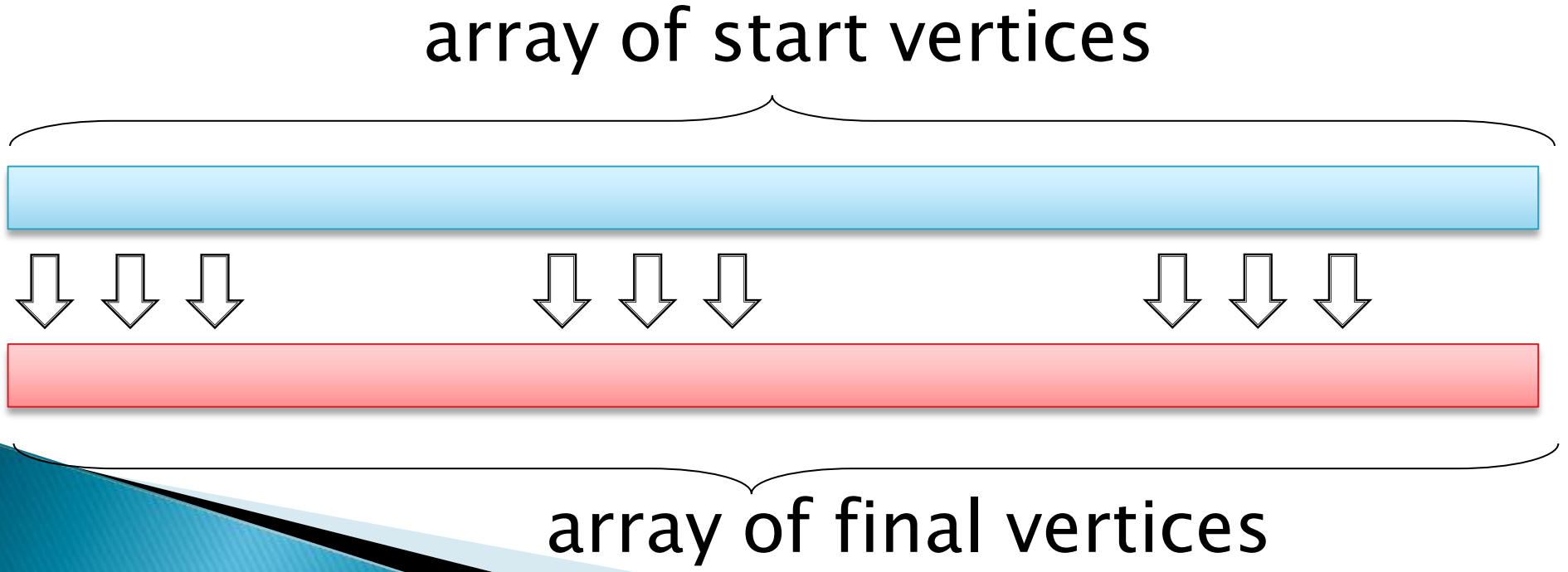


Последовательный доступ в память

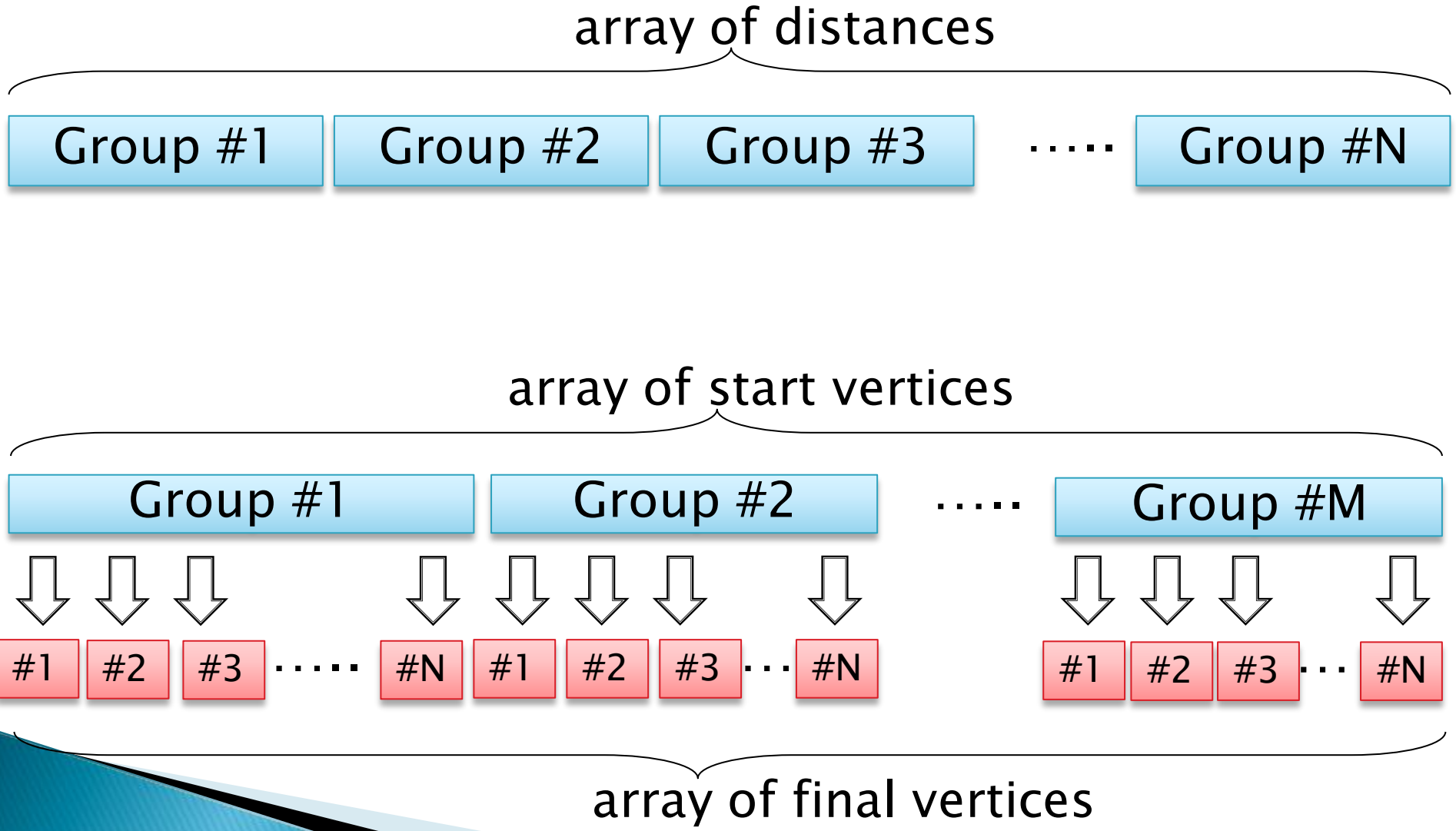


Случайный доступ в память

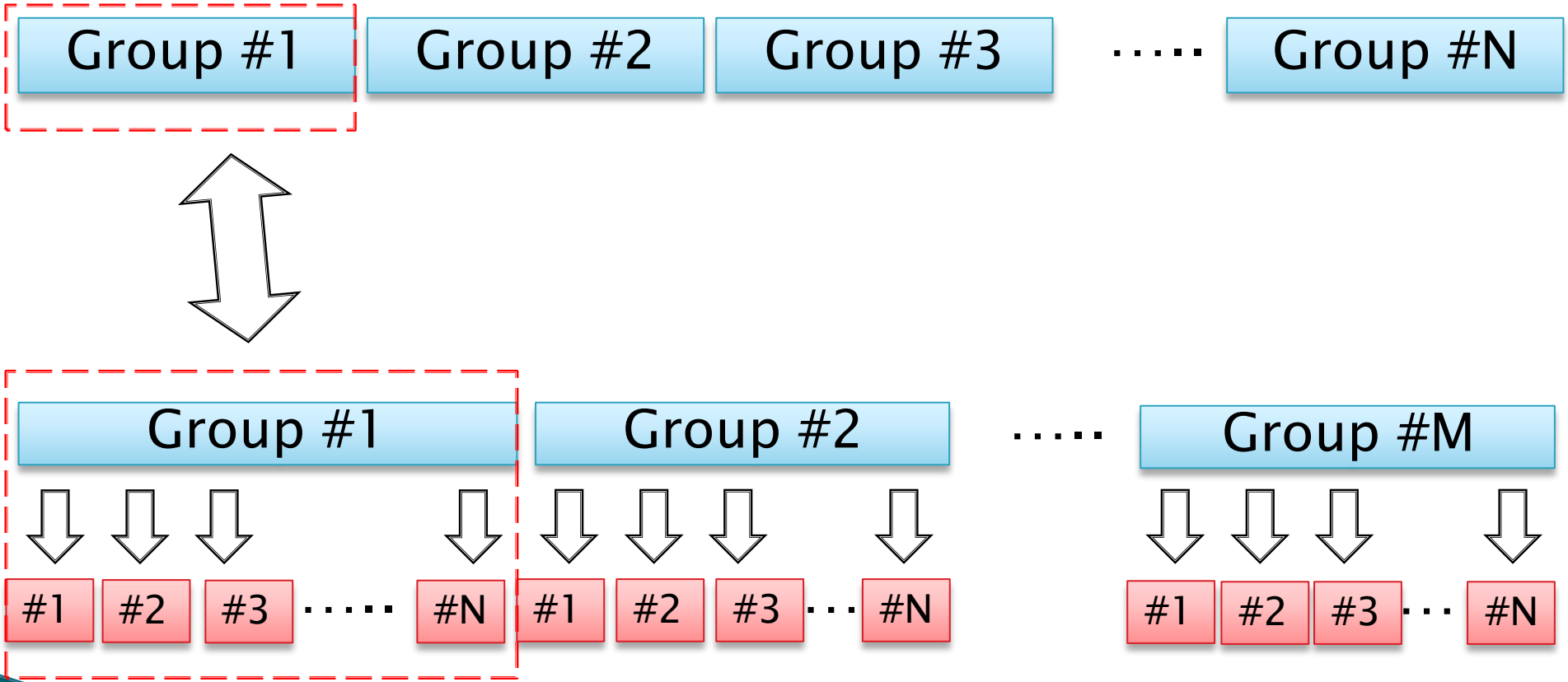
Переупорядочивание данных (исходный вариант)



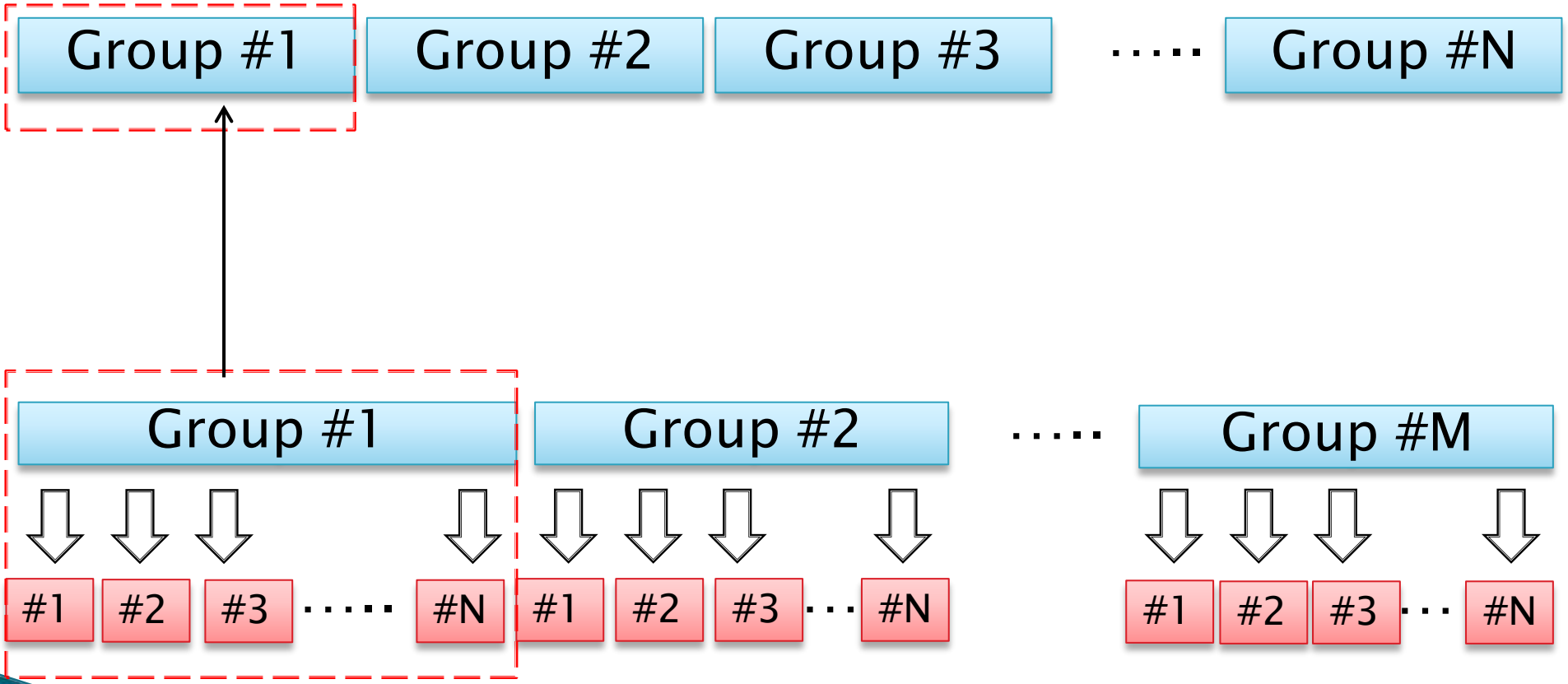
Преобразование (группировка)



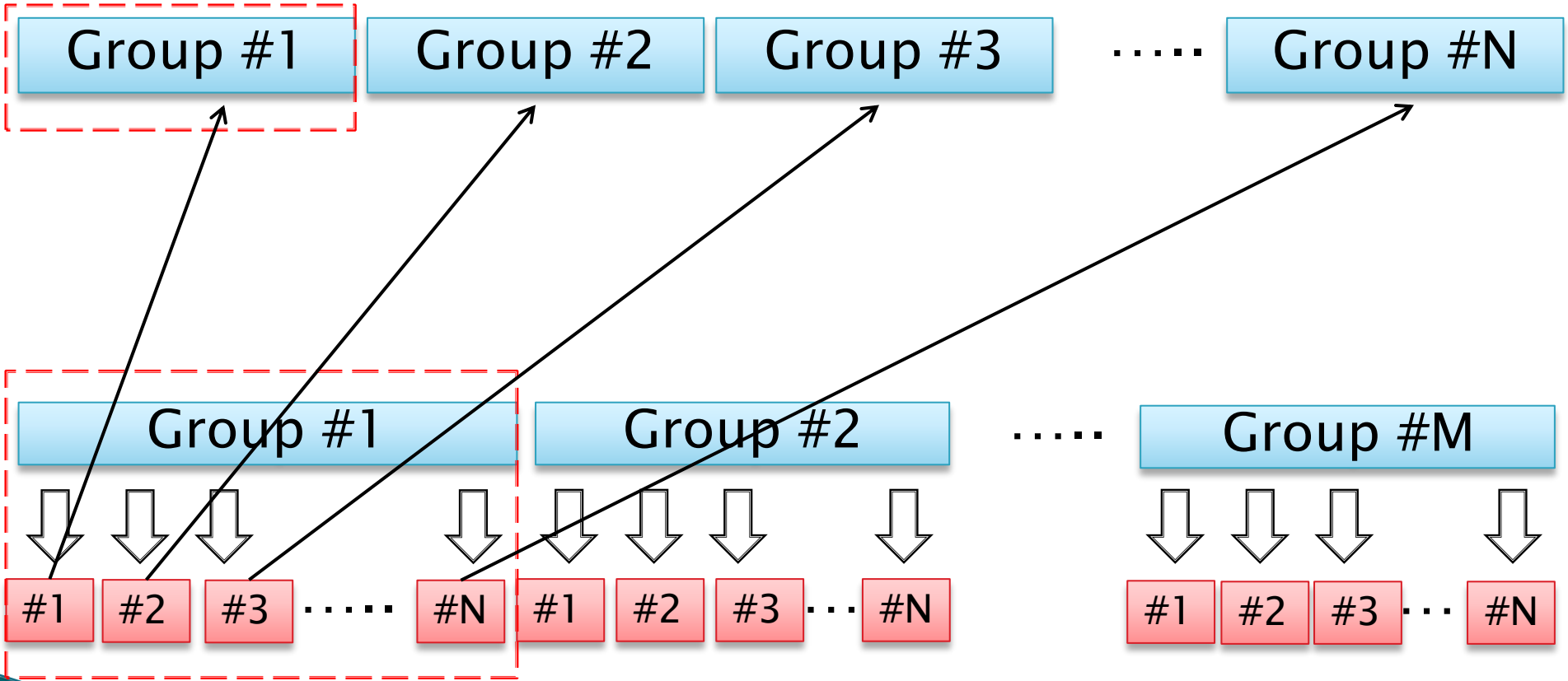
Преобразование (группировка)



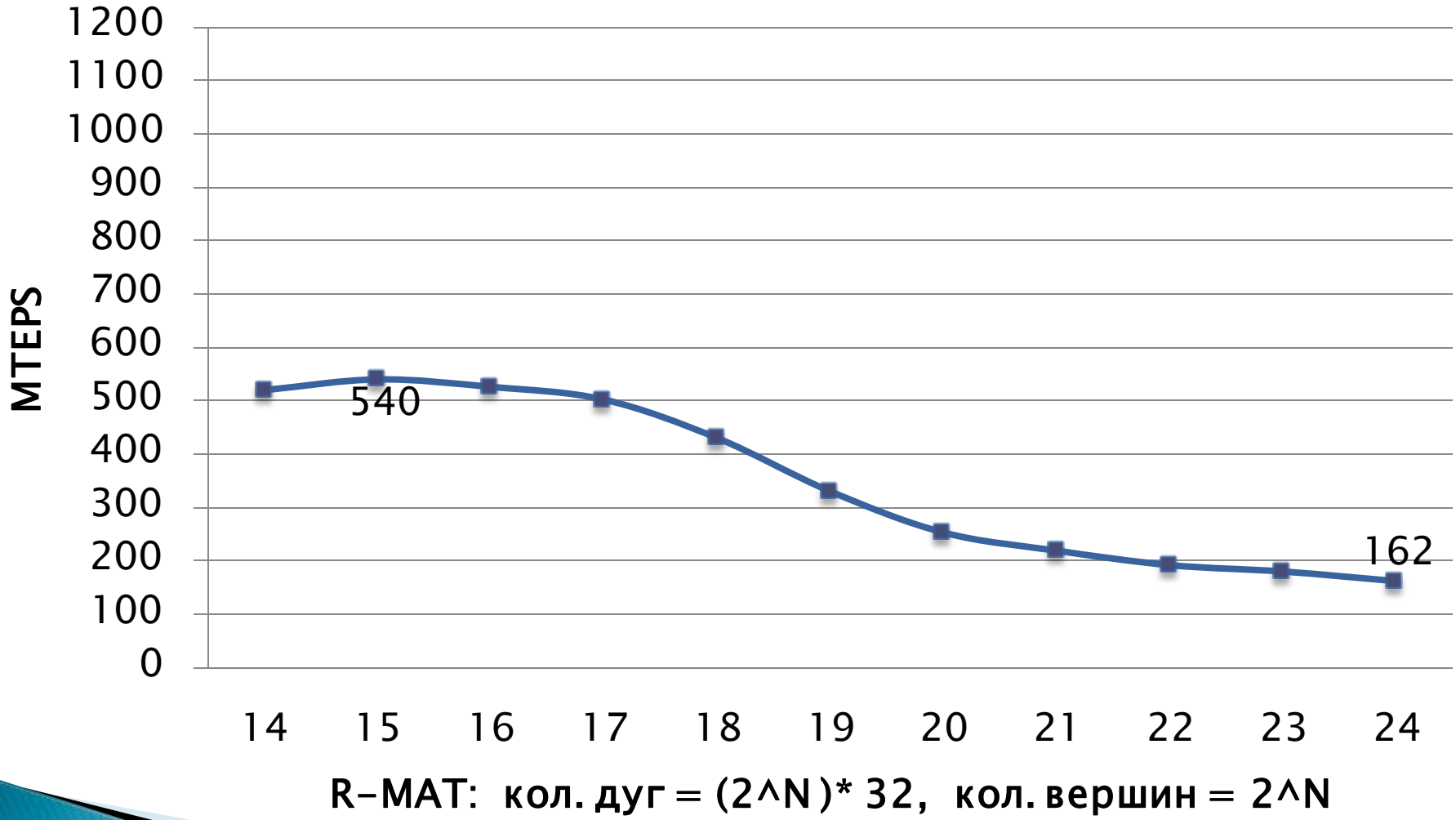
Преобразование (группировка)



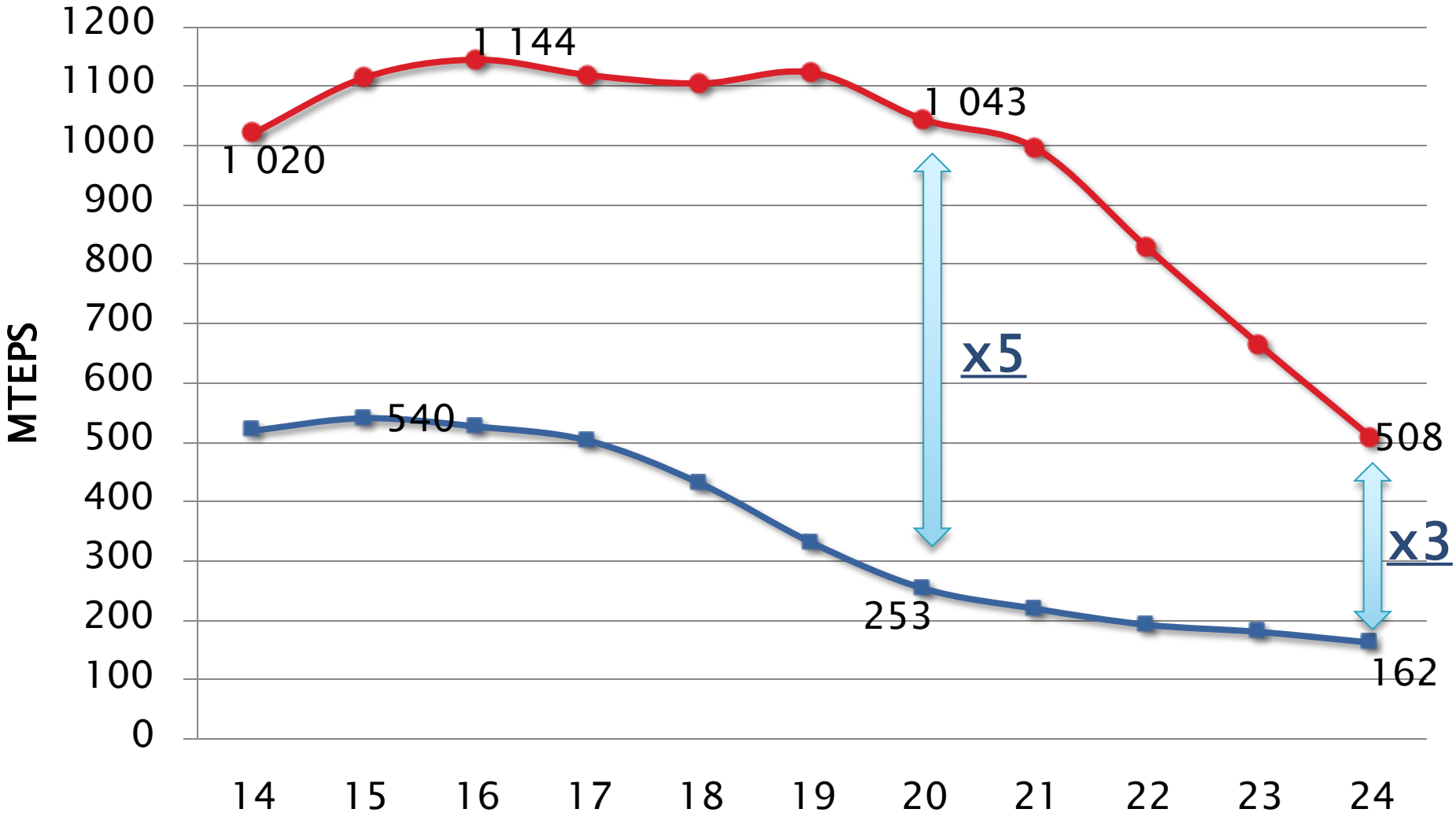
Преобразование (группировка)



Результаты SSSP: без оптимизаций



Результаты SSSP:

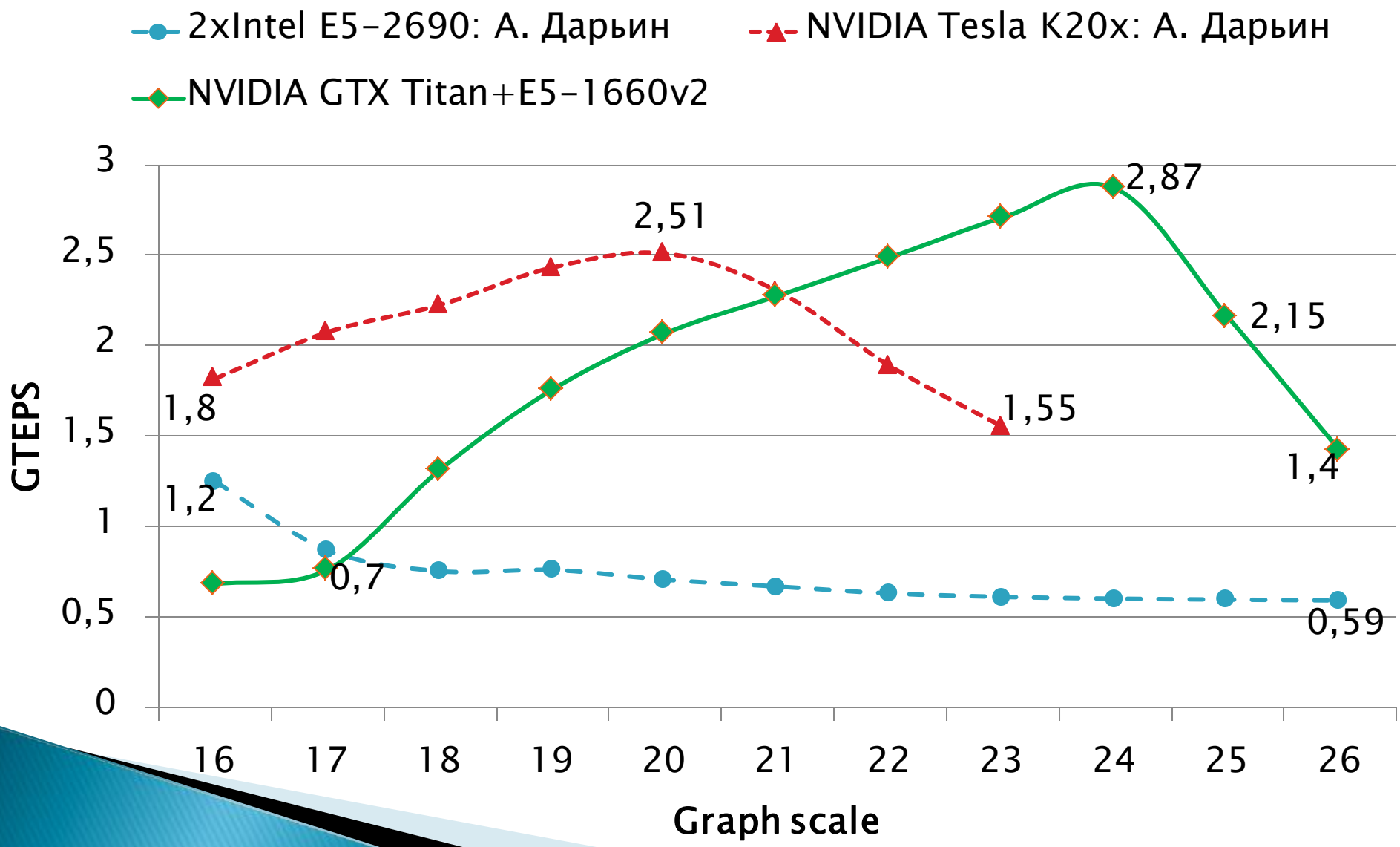


R-MAT: кол. дуг = (2^N)* 32, кол. вершин = 2^N

Преобразования данных для MST

- Локальная сортировка списка вершин;
- Перенумерация всех вершин графа, чтобы связные вершины имели более близкие номера;
- Отображение весов графа в целые числа; (возможно для графов с количеством уникальных ребер до 2^32);
- Сжатие информации о вершинах (применимо только для SSCA2 графа). Позволяет сжать исходный граф в **2.9 раз**.

Результаты конкурса (MST) GraphHPC 2015



Спасибо за внимание !

Вопросы ?

Ссылки:
SSSP – http://habrahabr.ru/post/214515/
MST – http://habrahabr.ru/post/253031/
GraphHPC 2014 – http://dislab.org/GraphHPC-2014/
GraphHPC 2015 – http://dislab.org/GraphHPC-2015/

контакты: alexander.k.s@mail.ru / Колганов А.С.