Санкт-Петербургский государственный университет

Кафедра системного программирования

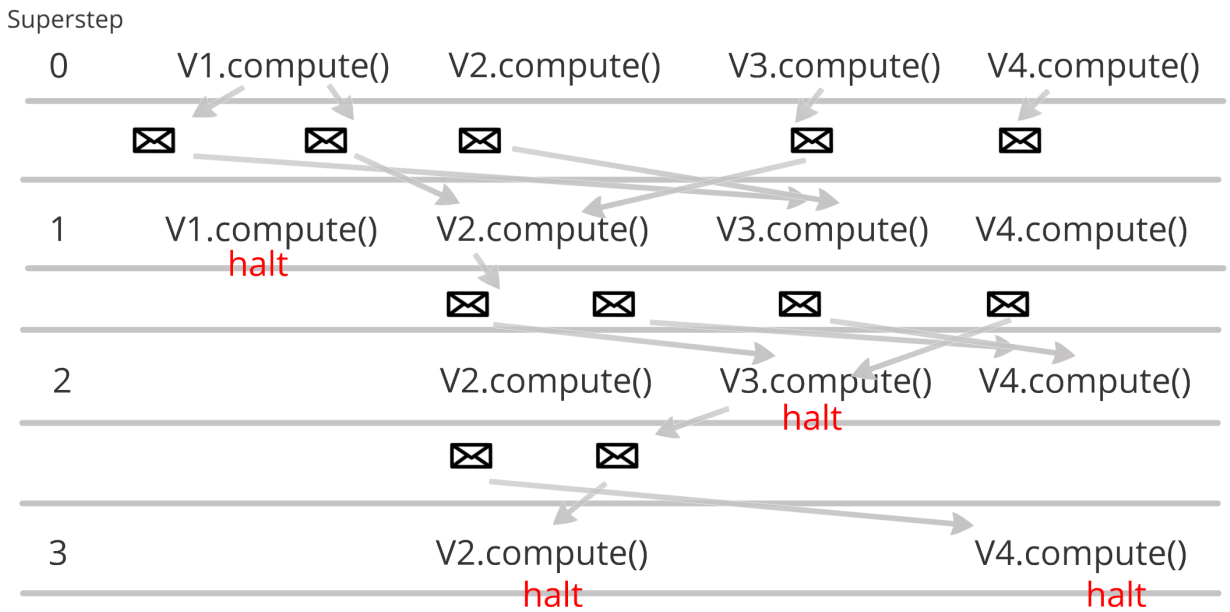
Алгоритм Борувки с SPLA и Apache Giraph

Ахмедов Д., Парфёнов Д.

Санкт-Петербург
2025

Apache Giraph

- Та же идея, что и Pregel
- "Думаем" как вершина, а не как граф
  - Получаем сообщение
  - Обрабатываем, что-то делаем направленна данная работа
  - Отправляем сообщение



Алгоритм Борувки в контексте Apache Giraph

- Пусть у вершин есть поле в виде ID компоненты (в начале это ID самой вершины).
- Каждая вершина отсылает ID компоненты смежным вершинам.
- Получаем сообщения, находим минимальное ребро, отсылаем его лидеру компоненты.
- Лидер находит минимальное ребро среди присланных, добавляет в ответ.
- Лидер "влившийся" компоненты передает остальным вершинам в компоненте новый ID.
- Проверку на минимальное остовное дерево (сколько сейчас компонентов связаности) можно осуществлять с помощью функции агрегирования Apache Giraph.

Алгоритм Борувки с SPLA [1/2]

Алгоритм работает итеративно. В цикле будем поддерживать значения:

- M – матрица с ребрами между вершинами разных К.С.
- edge[i] – индекс к вершине в построенном MST
- super[i] – вершины-представители К.С.
- min_edge[i] – для поддержки минимального ребра из К.С. в К.С. (только для представителей)

Алгоритм Борувки с SPLA [2/2]

Как устроен основной цикл:

- Полукольцо
  - область значений — $\mathbb{R} \cup \{+\infty\}$
  - операция "умножения" — конструктор пары вида (вес, номер вершины)
  - операция "сложения" — выбор минимума среди весов
  - $0 = +\infty$
- Для каждого представителя К.С. будем искать минимальное ребро к любой другой К.С.
- Обновим представителей из К.С.
- Далее обновим M, оставив лишь ребра между новыми К.С.

Экспериментальный набор данных

Граф	Тип	Вершин	Ребер	К.С.	Средн. степ.	Макс. степ.
Борувка[1]						
rmat16.sym.egr	RMAT	65k	483k	3900	14.8	569
internet	topology	125k	387k	1	3.1	151
USA-road-d.NY	road map	264k	730k	1	2.8	8
citationCiteseer	citations	268k	2.3M	1	8.6	1.3k
amazon0601.egr	product co-purchases	403k	2.4M	7	2.1	2752
2d-2e20.sym	grid	1.0M	2.1M	1	4.0	4

[1] `https://dl.acm.org/doi/pdf/10.1145/3581784.3607093` – A High-Performance MST Implementation for GPUs (2023)

Эксперимент

- Сравнение алгоритмов с использованием SPLA vs Giraph: время выполнения
- Сравнение алгоритмов с использованием SPLA vs Giraph: потребление памяти

Характеристики тестового оборудования

- AMD Ryzen 7 6800H (встроенная видеокарта AMD Radeon 680M (2200 МГц))
  - 8 ядер
  - все ядра равнозначны
  - без гипертредининга
- RAM 16 GB @ 6400 MHz
  - без свопа
- Ubuntu 24.10 x86_64