Теория графов.
Вторая презентация

Ермолович Анна

Введение

Рассматривается параллельное выполнение алгоритмов **Борувки и MS-Parent BFS** на **Pregel+**

Исследуется **масштабируемость** алгоритмов при увеличении числа MPI-процессов

Анализируется **влияние числа процессов** на **время выполнения**, **ускорение** и **эффективность**

Характеристики вычислительной машины

- **Процессор**: AMD Ryzen 7 5800H (8 ядер, 16 потоков)
  - L1-кэш: 64 KB (на ядро)
  - L2-кэш: 512 KB (на ядро)
  - L3-кэш: 16 MB (общий)
- **RAM**: 32 GB
- **Операционная система**: Ubuntu 24.04.2

Формулировка экспериментов

Эксперименты

Цель: Исследовать масштабируемость Pregel+ для алгоритмов **Борувки** и **MS-Parent BFS** – как изменяется время выполнения и эффективность при увеличении числа процессов – на разных графах из датасетов для разных алгоритмов.

Ход эксперимента:

1. Для каждого выбранного графа из датасета 20 раз запускаем Pregel+ с использованием mpirun на 1, 2, 4, 8 и 16 процессах.
2. Измеряем время выполнения, фиксируем результаты.
3. Строим графики для каждого графа:
   a. Зависимость времени выполнения от числа процессов
   b. Зависимость ускорения и эффективности от числа процессов
4. Сравниваем результаты и делаем выводы о масштабируемости Pregel+.

Проблемы:

- Масштабируемость может различаться для разных графов (разная структура и размер).
- Из-за параллельной природы вычислений возможны колебания времени между запусками.

Метрики оценки масштабируемости

Масштабируемость – это способность системы к увеличению производительности при добавлении новых ресурсов или способность эффективно перераспределять имеющиеся ресурсы при увеличении нагрузки.

Ускорение:

Показывает, во сколько раз выполнение стало быстрее при p процессах, чем при 1:

$$S(p) = \frac{T(1)}{T(p)}$$

Эффективность:

Показывает, насколько эффективно используются вычислительные ресурсы:

$$E(p) = \frac{S(p)}{p}$$

Критерии анализа:

1. Строим графики зависимости $T(p), S(p), E(p)$ от p.
2. Ищем, при каком p эффективность заметно падает.
3. Оцениваем влияние аппаратной многопоточности (SMT) на масштабируемость Pregel-алгоритмов.

Критерии оценки масштабируемости

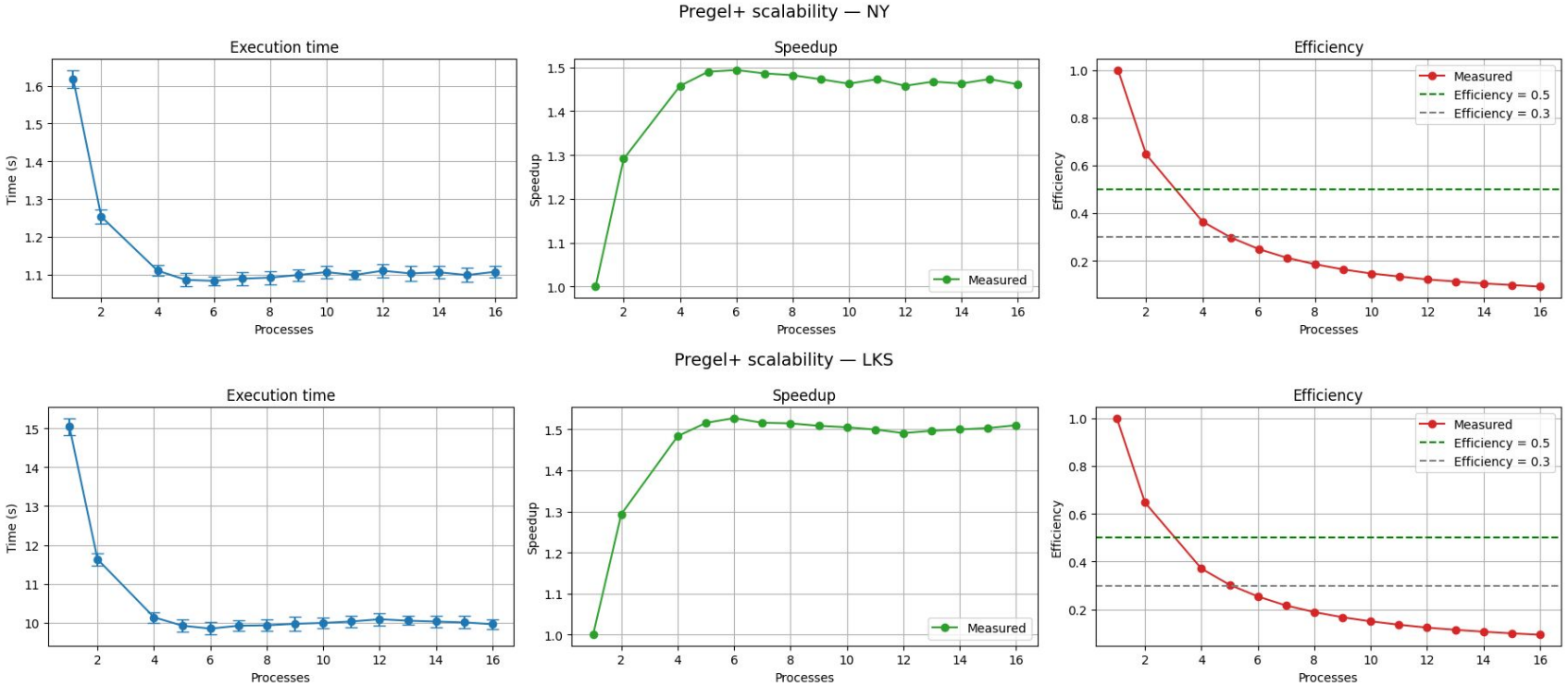
1. **Анализ зависимости** $T(p), S(p), E(p)$ **от** p**.** Строятся графики:
   a. время выполнения vs число процессов
   b. ускорение vs число процессов
   c. эффективность vs число процессов
2. **Оценка рентабельности добавления ресурсов.** Используется эмпирическая шкала эффективности:
   a. **50–70%** – **хорошая** масштабируемость
   b. **30–50%** – **умеренная** масштабируемость
   c. **< 30%** – **низкая** рентабельность добавления процессов
3. **Влияние аппаратной многопоточности (SMT) на масштабируемость Pregel-алгоритмов.**

Борувка

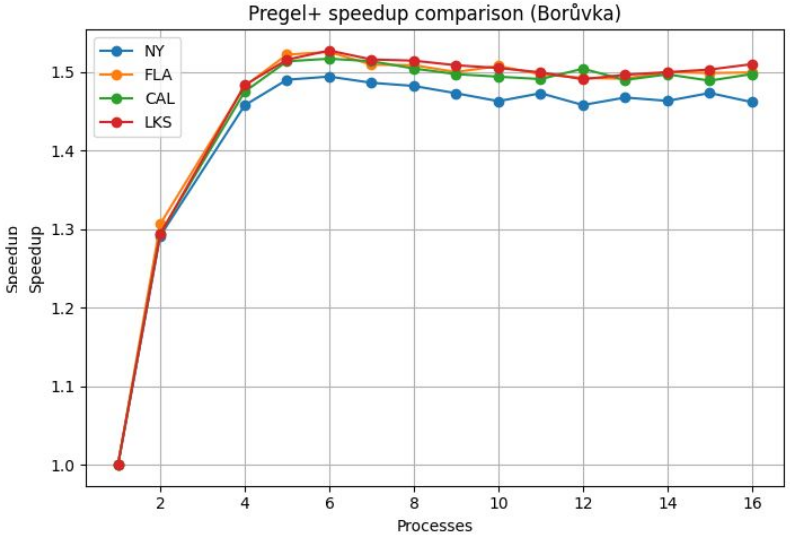
Датасет DIMACS 9th

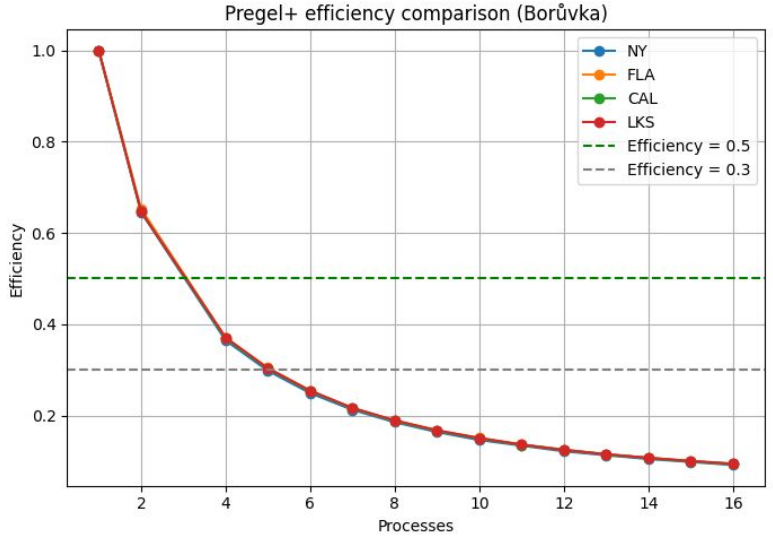
Description	Nodes	Edges
Great Lakes	2,758,119	3,397,404
California and Nevada	1,890,815	2,315,222
Northeast USA	1,524,453	1,934,010
Northwest USA	1,207,945	1,410,387
Florida	1,070,376	1,343,951
Colorado	435,666	521,200
San Francisco Bay Area	321,270	397,415
New York City	264,346	365,050

Эксперимент 1. Результаты для Борувки



Эксперимент 1. Результаты для Борувки





Почему масштабируемость Борувки ограничена

Факты реализации

- Одна итерация Борувки – 12 супершагов
- Между супершагами – глобальная синхронизация Pregel

Что происходит в фазах

- Поиск минимального ребра: линейный проход по рёбрам компоненты
- Pointer jumping: 1 сообщение – ожидание ответа
- Формирование супервершин: 1 сообщение – обязательный барьер

Ключевой эффект

- По мере выполнения алгоритма объём полезной работы на фазу уменьшается
- Стоимость синхронизации остаётся постоянной

Вывод: Ограничение связано с алгоритмической структурой Борувки

Почему низкая степень вершин мешает Борувке

Свойства графов DIMACS

- Средняя степень вершины: 2–3
- Максимальная степень: редко выше 6–8

Где степень участвует напрямую

- Поиск минимального ребра:
  - время пропорционально числу инцидентных рёбер
  - 2–3 сравнения на компоненту

Следствие

- Фазы вычислительно пустые
- Глобальные барьеры доминируют по времени

Сравнение

- В social / web-графах степени – сотни и тысячи
- Там та же фаза содержит больше полезной работы и масштабируется лучше

Вывод

- Низкая степень – малый объём вычислений на фазу
- Это усиливает эффект глобальной синхронизации

Multiple-source Parent BFS

Датасет SNAP

Description	Nodes	Edges
Twitch gamers network	168,114	6,797,557
Gemsec Facebook dataset	134,833	1,380,293
Gemsec Deezer dataset	143,884	846,915
Twitch social networks	34,118	429,113
Github developer network	37,700	289,003
Facebook page-page network	22,470	171,002
Facebook social circles	4,039	88,234
Deezer Europe social network	28,281	92,752
LastFM Asia social network	7,624	27,806

Стартовые вершины в MS-Parent BFS

Множество стартовых вершин $S=\{s0,s1,\ldots,sk-1\}$ задаётся во входных данных.

В первую строку входного файла добавляется строка вида: `p k s0 s1 s2 ... s_{k-1}`
`p` – идентификатор строки параметров алгоритма
`k` – число стартовых вершин
`si` – идентификатор i-й стартовой вершины графа

Использование стартовых вершин в Pregel+

- Каждая стартовая вершина инициирует собственную BFS-волну.
- Все BFS-волны распространяются параллельно в рамках одного запуска Pregel+.
- Параллелизм достигается за счет распределения вершин графа между процессами, а не за счет прямого соответствия источник–процесс.

Параметры эксперимента

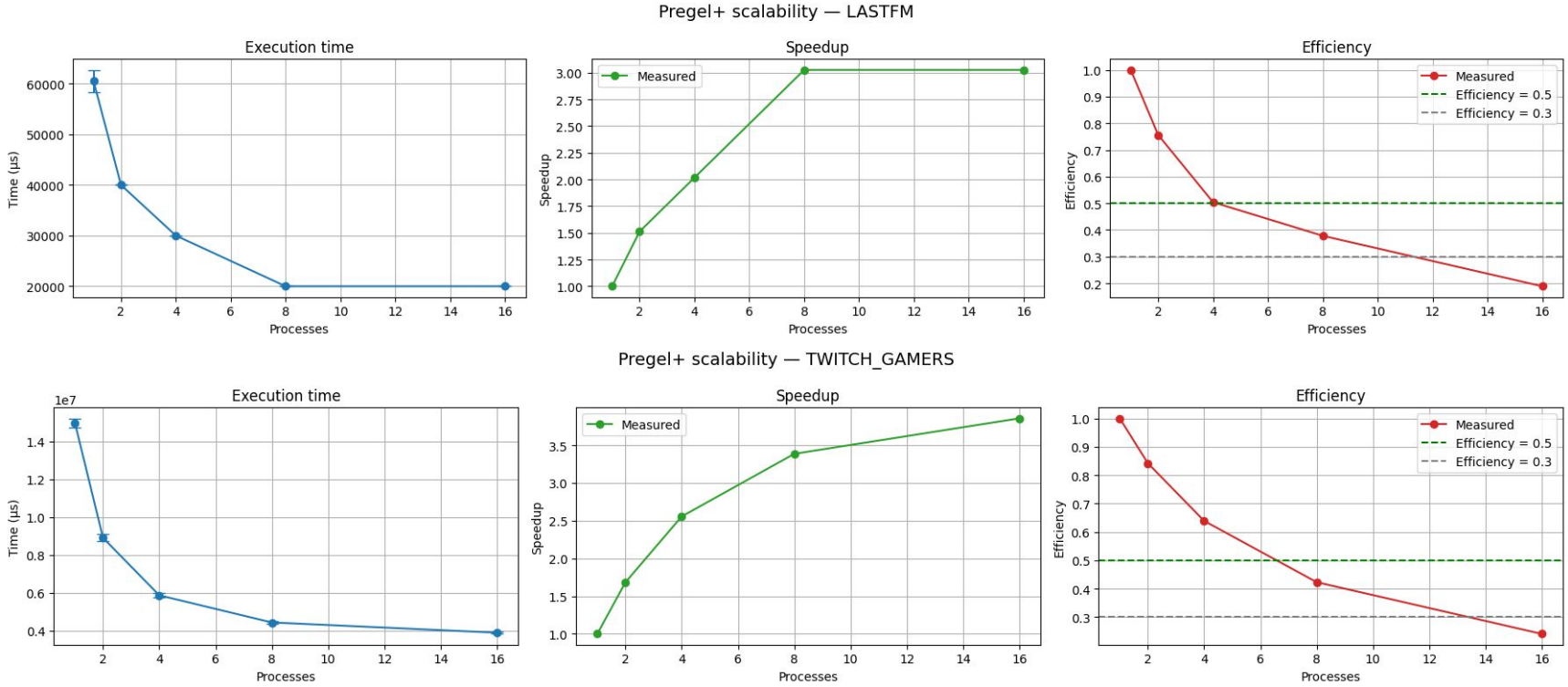
- В экспериментах использовалось фиксированное число стартовых вершин **k = 8**.
- Значение k выбрано как достаточное для формирования широкого BFS-фронта и загрузки физических ядер.
- Увеличение k сверх этого значения приводит к росту накладных расходов без заметного выигрыша в производительности.

Стартовые вершины в MS-Parent BFS

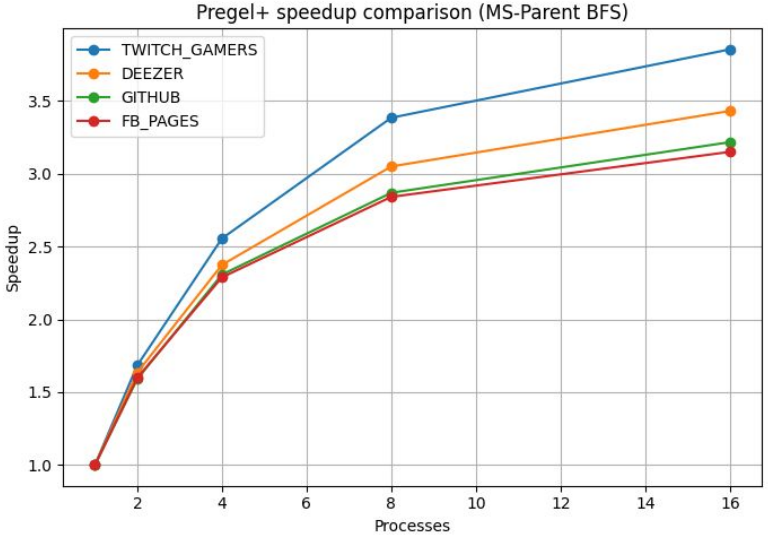
В качестве стартовых берутся вершины из графа с максимальной степенью
Примеры реальных задач

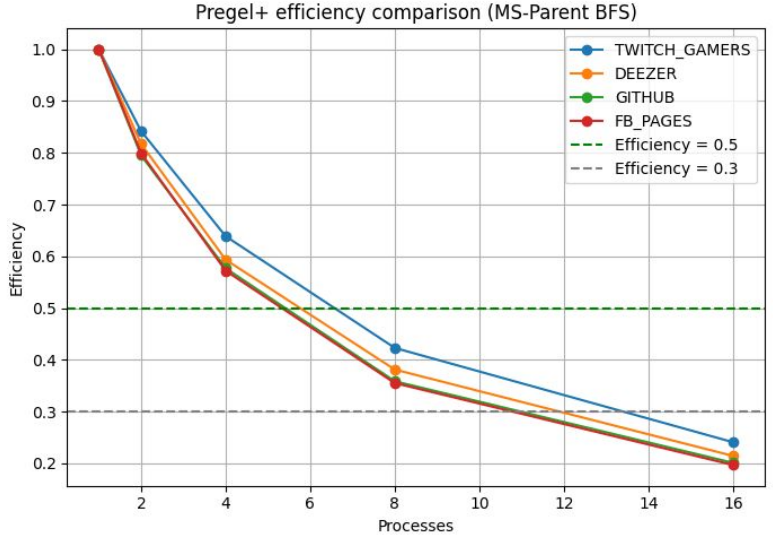
1. **Социальные сети**
   a. источники – популярные аккаунты
   b. анализ распространения информации
   c. Практический смысл:
      i. прогноз распространения новостей
      ii. анализ конкуренции источников информации
2. **Инфраструктура**
   a. источники – ключевые узлы сети
   b. анализ отказоустойчивости
   c. Практический смысл:
      i. при отказе источника s_i понятно, какие узлы затронуты первыми,
      ii. оценка зон риска
      iii. планирование резервирования

Эксперимент 2. Результаты для MS-Parent BFS

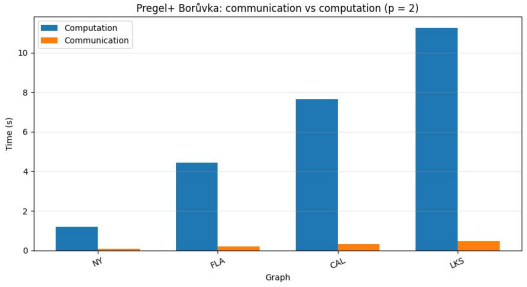


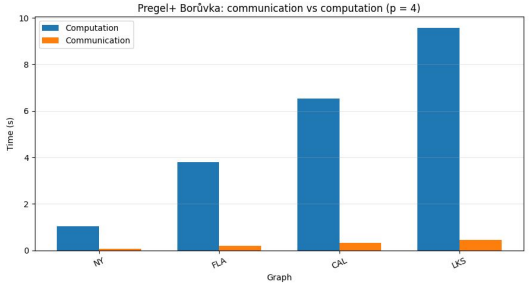
Эксперимент 2. Результаты для MS-Parent BFS

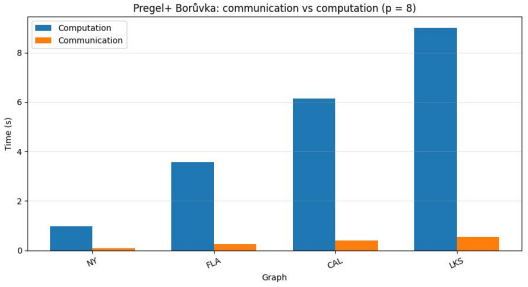


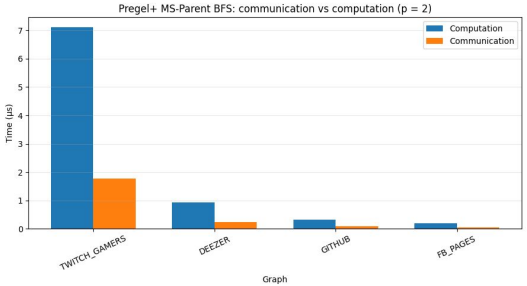


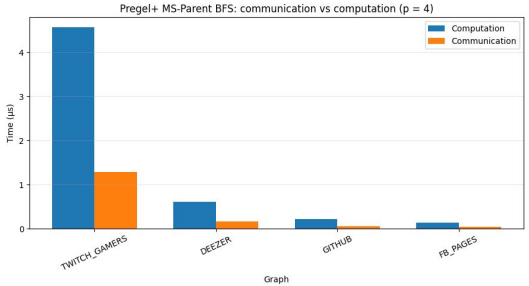
Коммуникации

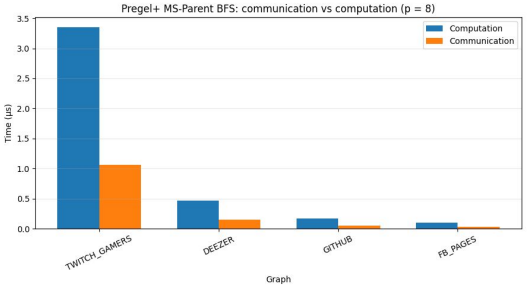




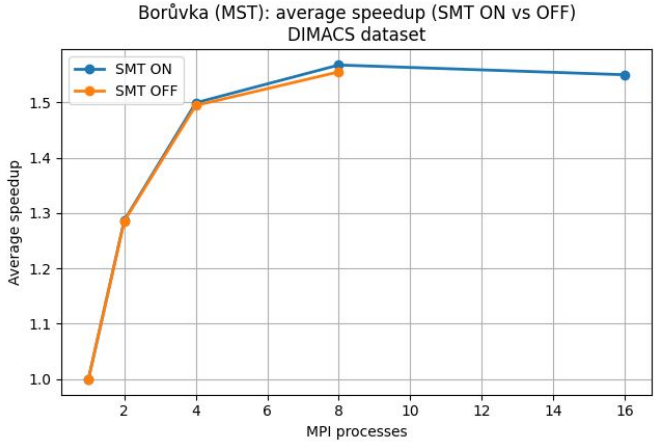


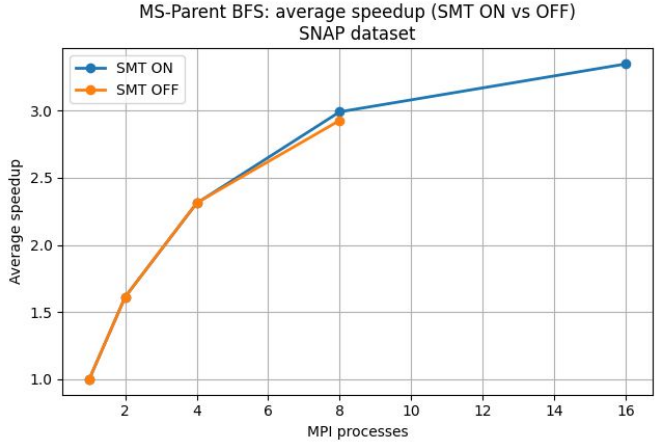






Экспериментальная оценка влияния SMT





Сравнение измерений с включенным и отключенным **SMT (Simultaneous Multithreading)** показывает, что различия в ускорении носят вторичный характер и не оказывают существенного влияния на общую масштабируемость Pregel+

Влияние SMT (Hyper-Threading) на результаты экспериментов

Эксперименты проводились на процессоре AMD Ryzen 7 5800H
(8 физических ядер, 16 логических потоков)

В рамках рассмотренных конфигураций (1, 2, 4, 8 и 16 MPI-процессов) добавление 16 процессов не приводит к росту ускорения по сравнению с 8 процессами и сопровождается заметным снижением эффективности

- Сравнение результатов эксперимента с результатами такого же, но с отключенным SMT показывает, что различия в ускорении не превышают 1–2% и не изменяют общего характера масштабируемости алгоритмов
- Это указывает на то, что наблюдаемое падение эффективности при $p = 16$ не связано напрямую с использованием SMT
- Влияние аппаратной многопоточности на наблюдаемое падение эффективности, согласно измерениям, является вторичным и незначительным

Выводы

- Для обоих алгоритмов увеличение числа MPI-процессов на начальном этапе приводит к сокращению времени выполнения, однако характер масштабируемости существенно различается: алгоритм MS-Parent BFS демонстрирует значительно лучшее ускорение, чем алгоритм Борувки.
- **Алгоритм Борувки демонстрирует ограниченную масштабируемость:** ускорение достигает максимума при 5–6 MPI-процессах (≈1.5), после чего наблюдается слабое снижение ускорения при дальнейшем увеличении числа процессов до 16. При этом доля коммуникационных затрат остается невысокой, что указывает на то, что основным ограничивающим фактором являются последовательные фазы алгоритма и необходимость глобальной синхронизации между супершагами
- **Алгоритм MS-Parent BFS масштабируется заметно лучше:** для него наблюдается более высокий прирост ускорения и более эффективное использование вычислительных ресурсов при увеличении числа процессов. Это связано с более равномерным распределением вычислительной нагрузки и отсутствием жёстких последовательных зависимостей между супершагами

Выводы

- Характер изменения ускорения при увеличении числа MPI-процессов определяется ограничениями сильной масштабируемости (strong scaling limit) и BSP-моделью исполнения Pregel+, при которых накладные расходы на глобальную синхронизацию супершагов и управление процессами начинают доминировать над полезными вычислениями.

- Эксперименты с включенным и отключенным SMT показывают, что влияние аппаратной многопоточности на данные эффекты является вторичным и не изменяет общего характера масштабируемости алгоритмов.