Санкт-Петербургский государственный университет

Кафедра системного программирования

Single, Multiple source BFS-parents с SPLA и SuiteSparse:GraphBLAS

Ахмедов Давид

Санкт-Петербург
2025

Характеристики тестового оборудования

- AMD Ryzen 5 5500U 2.1MHz
  - 6 ядер
  - все ядра равнозначны
  - без гипертредининга
- RAM 16 GB @ 3200 MHz
  - без свопа
- Ubuntu 22.04.5 x86_64

Детали реализации

Оба алгоритмы реализованы на Python

- Python 3.11.12
- Версии зависимостей[1]
- clang 14.0.0
- pyspla в качестве обертки SPLA
  - Версия SPLA – 74658a9, закоммичено 9 апреля 2025
  - для реализации необходимо было реализовать некоторые операторы и добавить их в ядро и библиотеку
- suitesparse-graphblas в качестве обертки SuiteSparse:GraphBLAS
  - необходимые операторы реализованы в Python-коде

[1] https://github.com/Parzival-05/graphs-hw/blob/main/requirements.txt – сохранённые зависимости в GitHub репозитории

Эксперимент

RQ:

- (ms) Какой из алгоритмов лучше масштабируется при увеличении числа стартовых вершин? Выбор 3, 30, 100 стартовых вершин (вершины выбираются случайно с фиксированным сидом, из K.K.C.).
- (ss) Сравнение алгоритмов с использованием SPLA vs SuiteSparse:GraphBLAS: время выполнения
- Сравнение алгоритмов с использованием SPLA vs SuiteSparse:GraphBLAS: потребление памяти (мониторинг пикового потребления за все итерации)

Замеры: Сбор статистик: N=10 повторов, M=10 экспериментов, IQR для борьбы с выбросами на каждом запуске

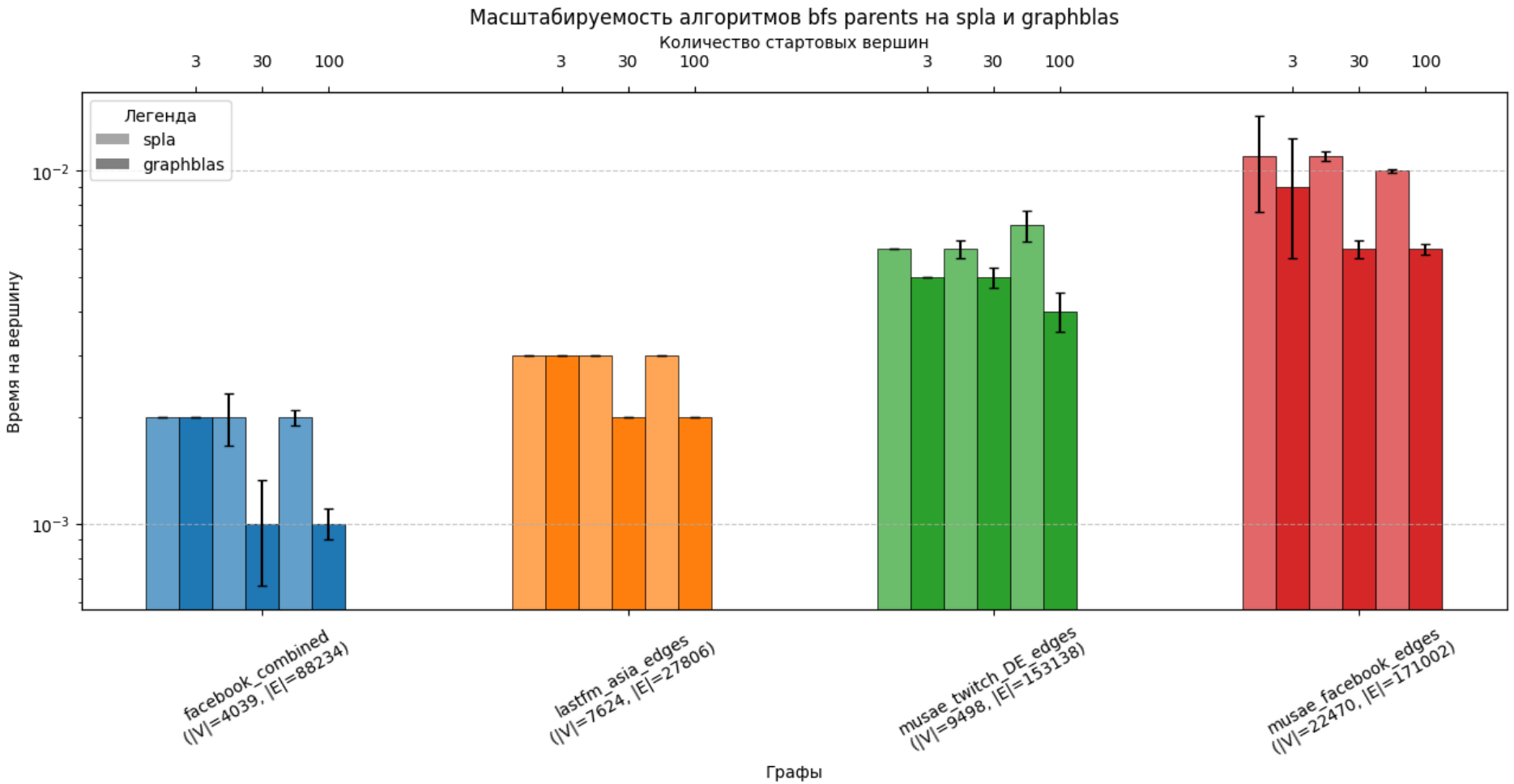
Экспериментальный набор данных

Таблица: Статистика графов SNAP[2]

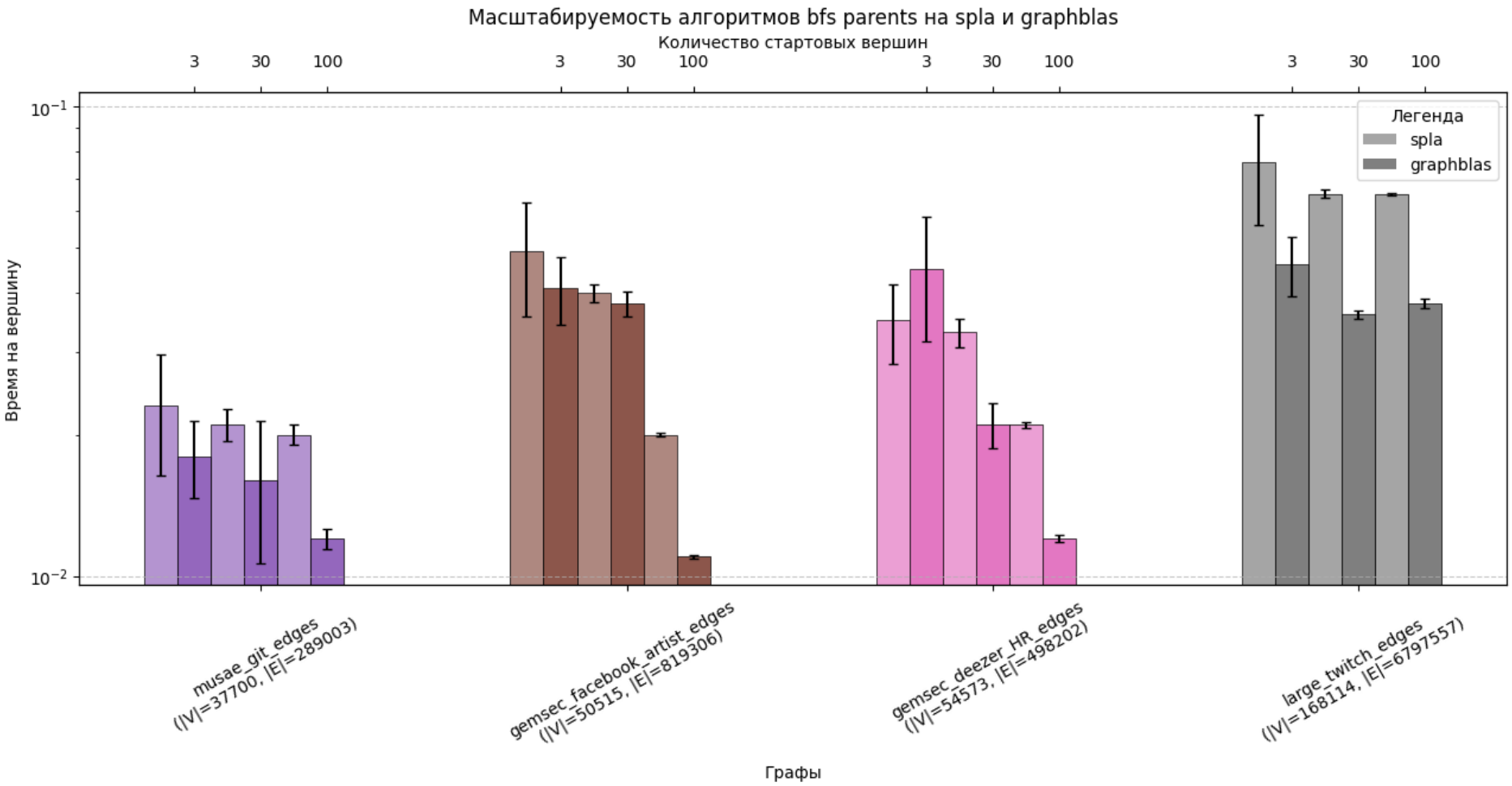
Граф (название в SNAP К.К.С)	Вершин	Ребер	К.С.	Ср. степ. в К.К.С	М. степ. в К.К.С	Крупн. К.С.
Twitch gamers network (large_twitch_edges)	168.1K	6.8M	1	80.86	35.2K	168.1K (100%)
Gemsec Facebook dataset (gemsec_facebook_artist_edges)	134.8K	1.4M	8	32.43	1.4K	50.5K (37.4 %)
Gemsec Deezer dataset (gemsec_deezer_HR_edges)	143.9K	846.9K	3	18.25	420	54.5K (37.9%)
Twitch social networks (musae_twitch_DE_edges)	34.1K	429.1K	6	32.24	4.2K	9.4K (27.5%)
Github developer network (musae_git_edges)	37.7K	289.0K	1	15.33	9.5K	37.7K (100%)
Facebook page-page network (musae_facebook_edges)	22.5K	171.0K	1	15.22	709	22.5K (100%)
Facebook social circles (facebook_combined)	4.0K	88.2K	1	43.69	1.0K	4.0K (100%)
LastFM Asia social network (lastfm_asia_edges)	7.6K	27.8K	1	7.29	216	7.6K (100%)

[2] http://snap.stanford.edu/data – Stanford Large Network Dataset Collection

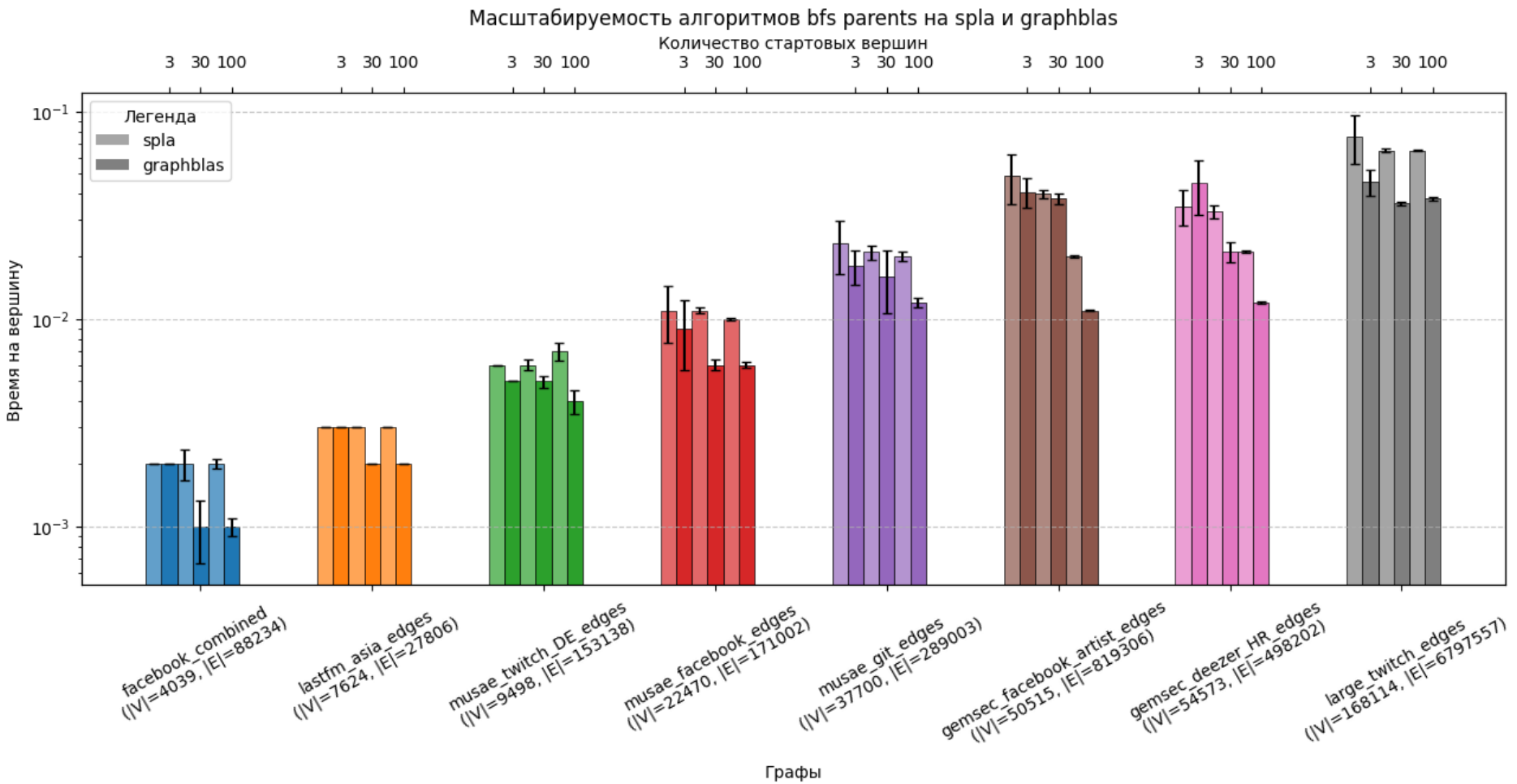
RQ1: Масштабирование



RQ1: Масштабирование



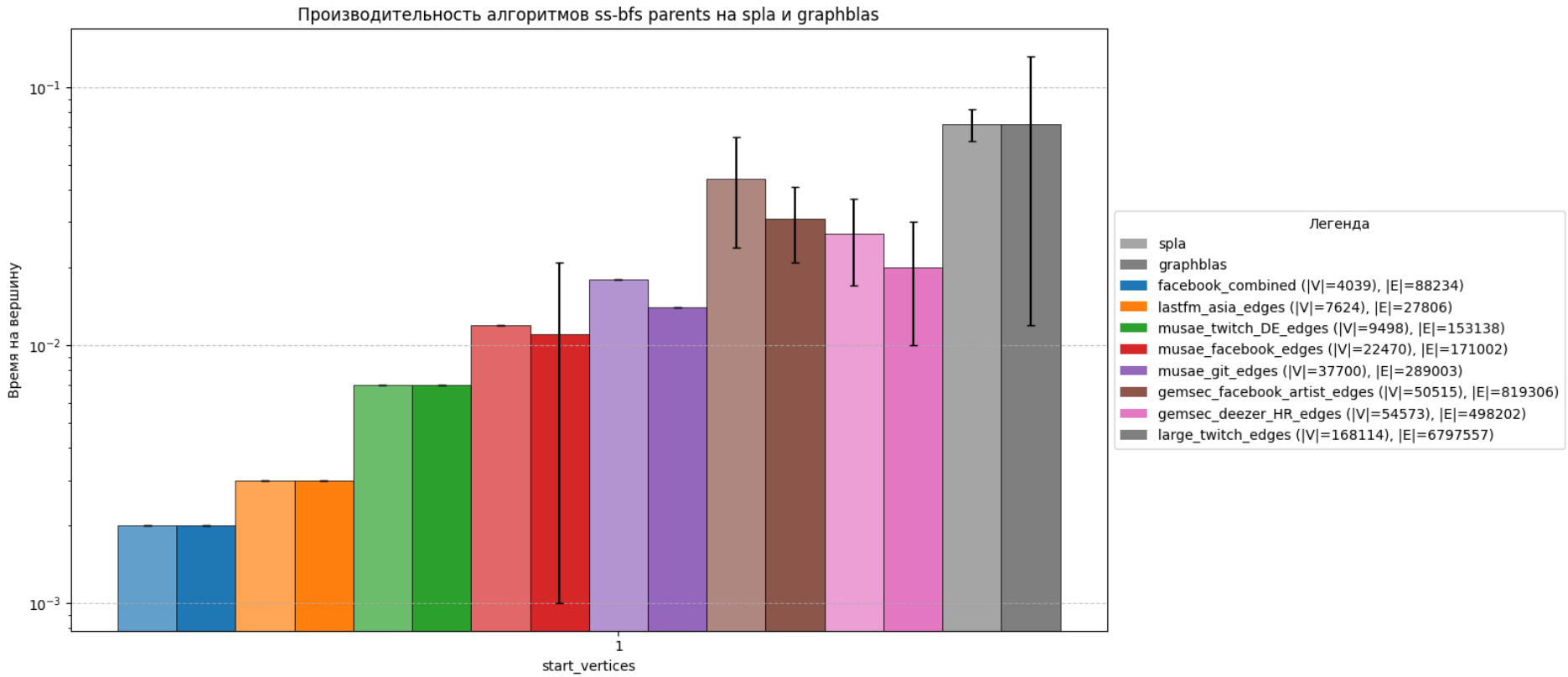
RQ1: Масштабирование



RQ1: выводы

- На тестовой выборке GB стабильно лучше масштабируется при увеличении количества стартовых вершин
- В некоторых случаях увеличение количества стартовых вершин давало прирост производительности в 2-4 раза
  - gemsec_facebook_artist_edges
  - gemsec_deezer_HR_edges

RQ2: производительность ss-bfs parents



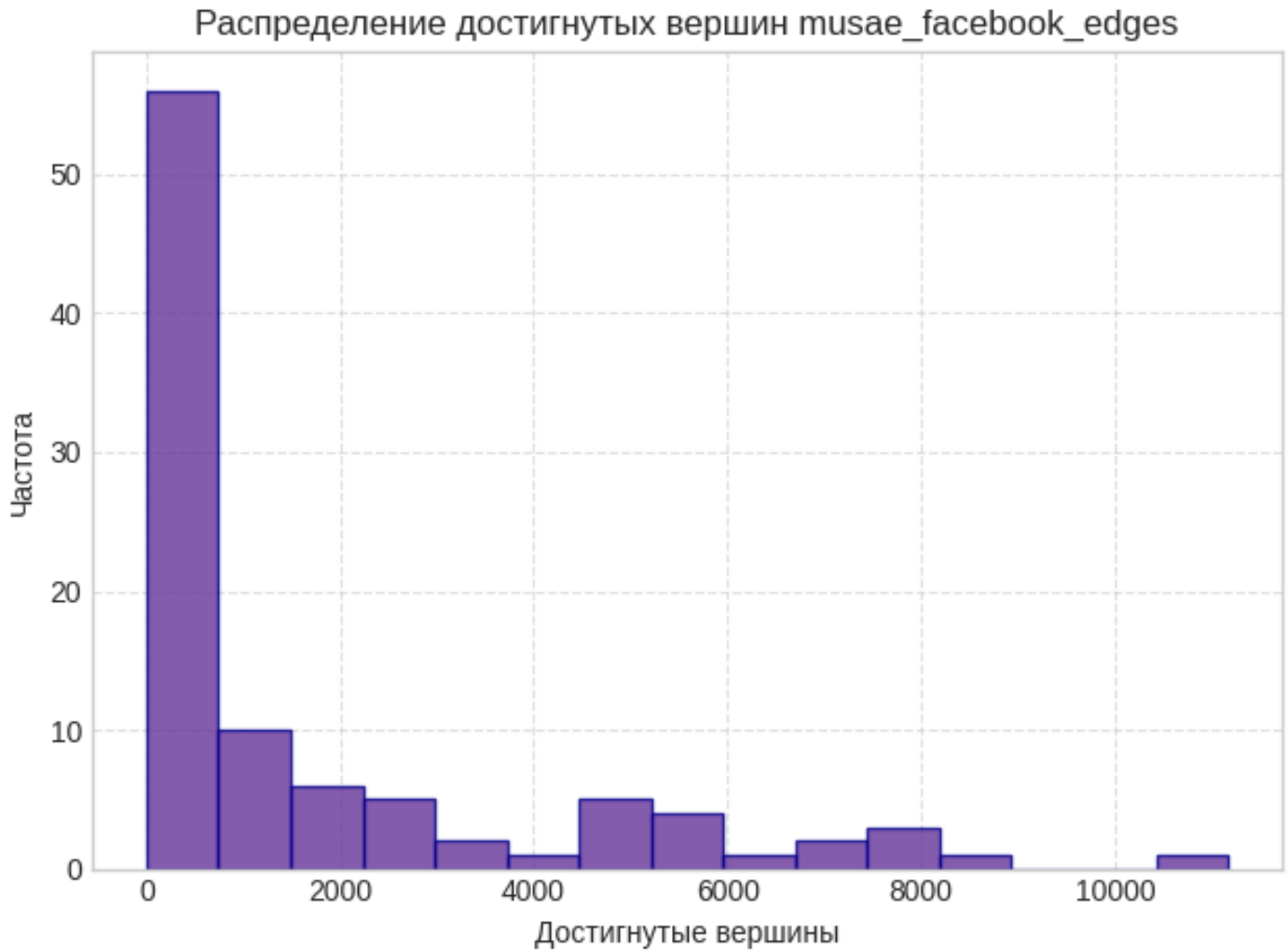
RQ2: обсуждение

Однако GB оказался вычислительно менее стабилен, чем SPLA. В некоторых случаях значительно:

- musae_facebook_edges
- large_twitch_edges

Гипотеза: количество достижимых вершин распределено неравномерно.

RQ2: обсуждение. Проблема с дисперсией.



RQ2: обсуждение. Проблема с дисперсией.

```
200830 function calls (200772 primitive calls) in 8.542 seconds

   Ordered by: internal time

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
      150    3.166    0.021    3.237    0.022 matrix.py:429(from_lists)
      580    1.345    0.002    1.359    0.002 matrix.py:621(mxm)
       50    1.106    0.022    1.106    0.022 bfs_parents.py:40(<listcomp>)
       50    0.712    0.014    0.712    0.014 bfs_parents.py:39(<listcomp>)
       50    0.652    0.013    0.652    0.013 bfs_parents.py:38(<listcomp>)
     2900    0.418    0.000    0.441    0.000 matrix.py:1107(eadd)
```

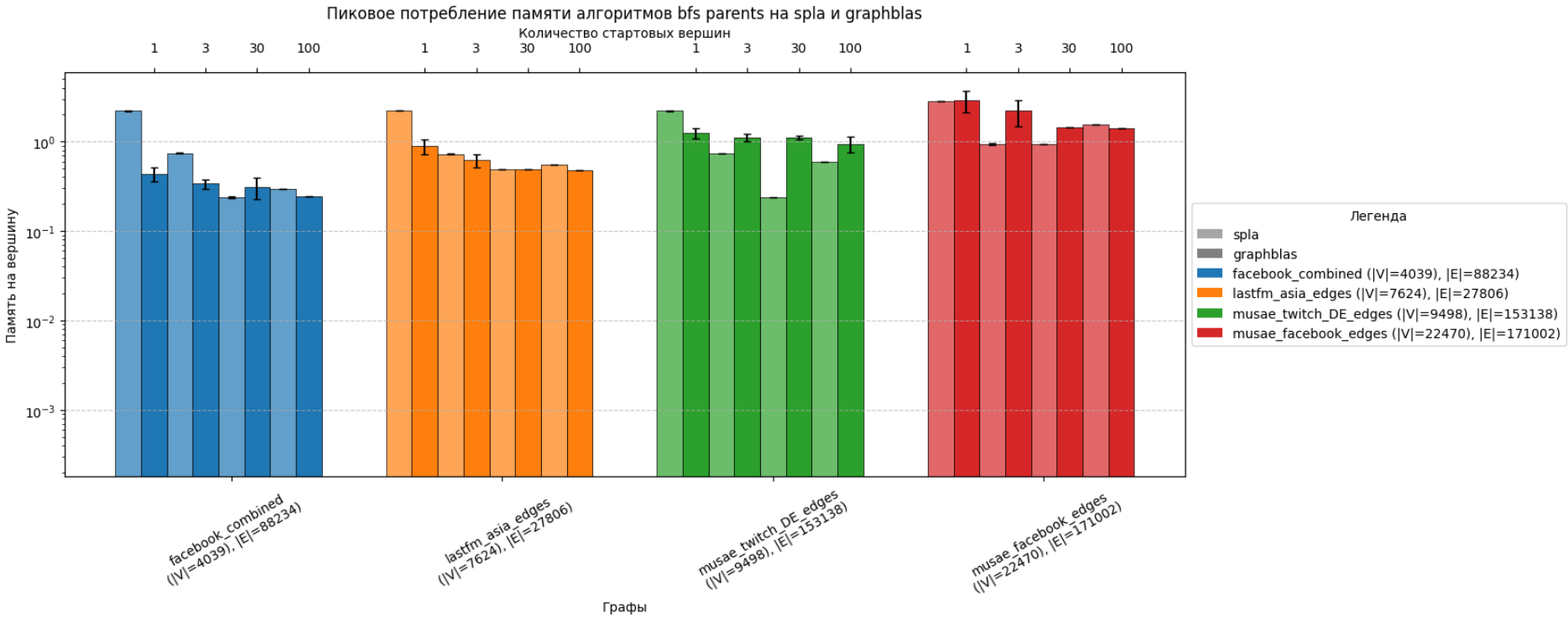
Рис.: Результаты профилирования musae_facebook_edges на SPLA с cProfile

- SPLA тратит много времени на первичную инициализацию, из-за чего более вычислительно стабилен
- Из-за этого он менее производительный
- GB инициализирует структуры на более низком уровне, за счет чего не страдает от этой проблемы

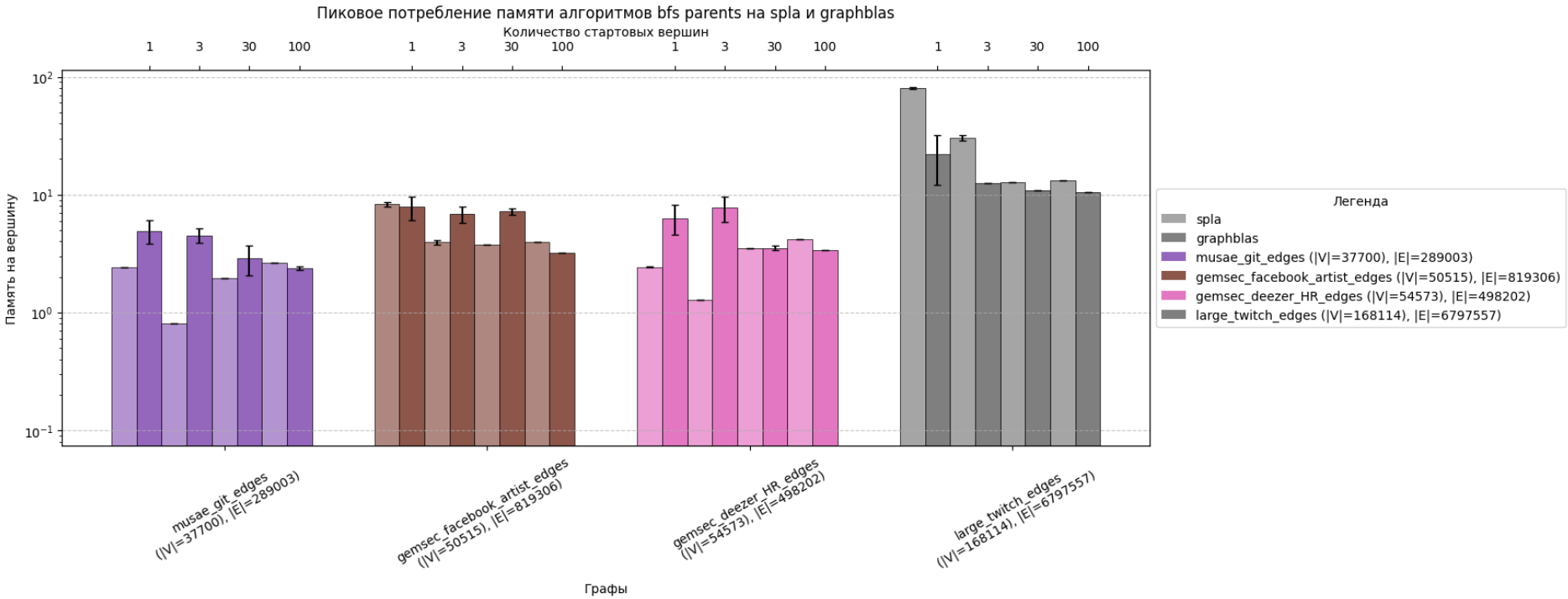
RQ2: выводы

- GB в среднем оказался производительнее. В некоторых случаях разница в производительности была почти в 1.5 раза в пользу GB.
- Вычислительная нестабильность GB связана с тем, что количество достижимых вершин распределено неравномерно и библиотека позволяет инициализировать нужные структуры более тонко, в отличие от SPLA.

RQ3: пиковая память



RQ3: пиковая память



RQ3: выводы

- С небольшим количеством стартовых вершин (1, 3) результаты сопоставимые
- GB использует стабильно меньше памяти при большем количестве стартовых вершин (30, 100 вершин)

Заключение

Итоги работы:

1. Производительность, масштабируемость (время выполнения)
   - GraphBLAS демонстрирует превосходство в скорости на большинстве тестовых графах
   - При увеличении числа стартовых вершин до 100 наблюдается устойчивое преимущество GraphBLAS в 1.5—2 раза
2. Память
   - Для небольшого количества стартовых вершин (1, 3) результаты сопоставимые
   - При большем количестве стартовых вершин (30, 100 вершин) предпочтительнее GraphBLAS
3. Выявлена проблема с неэффективной инициализацией структур через pyspla

Ссылка на код: `https://github.com/Parzival-05/graphs-hw`