УНИВЕРСИТЕТ ИТМО

Реализация TinkerPop инфраструктуры для WebGraph

Бакалаврская работа

Стародубцев Андрей Игоревич, M34391

Научный руководитель: Аксенов В.Е., доцент ФИТиП, к.т.н.

Консультант: Stefano Zacchiroli, HDR // Full Professor, Polytechnic Institute of Paris

Санкт-Петербург, 2022

Анализ данных в графовом представлении

Нативная реализация

- Ручная реализация на каждый запрос

Графовые базы данных

- Поддерживают DSL, например Gremlin

Актуальность

Большие графы

- Десятки млрд вершин, сотни млрд ребер

Графовые базы данных

- Затратны в масштабировании

Сжатые графы

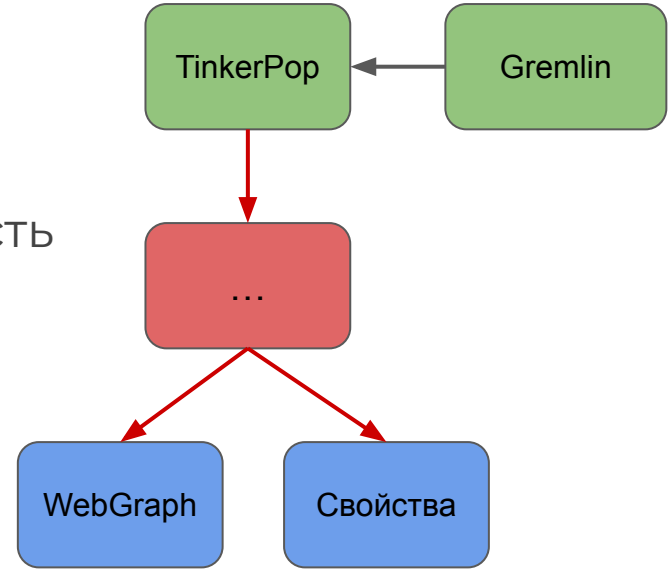
- Не поддерживают DSL

Цель

Добавление поддержки Gremlin для WebGraph

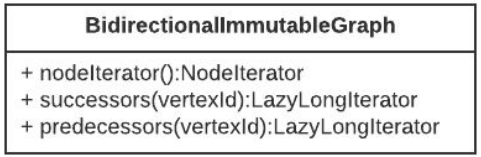
Постановка задачи

- Реализовать TinkerPop для WebGraph
- Добавить универсальный способ указания свойств
- Проанализировать производительность

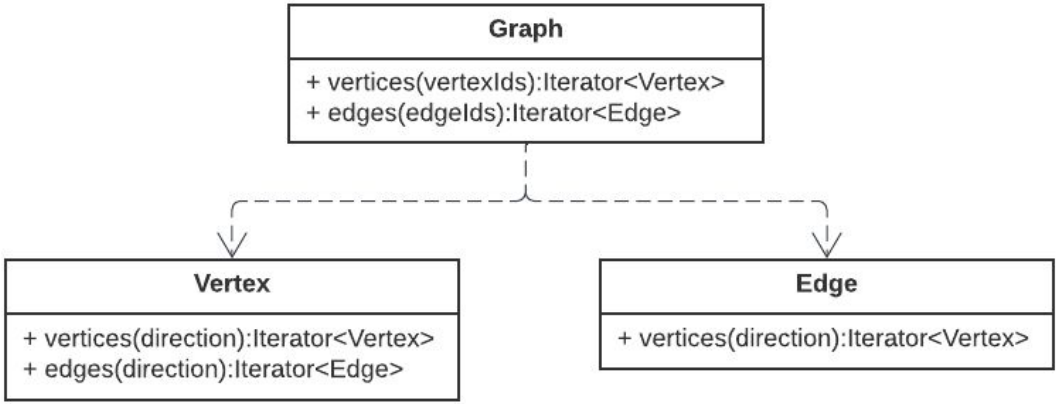


Реализация Apache TinkerPop для WebGraph

WebGraph



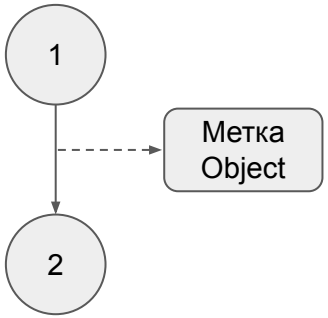
TinkerPop



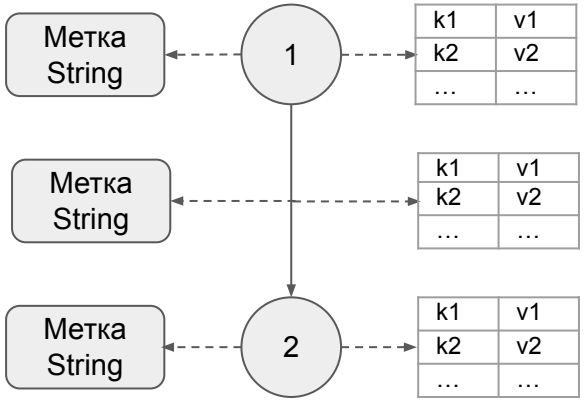
**Неполная диаграмма. Изображены только классы и методы, связанные со структурой графа*

Универсальный способ управлениями свойствами

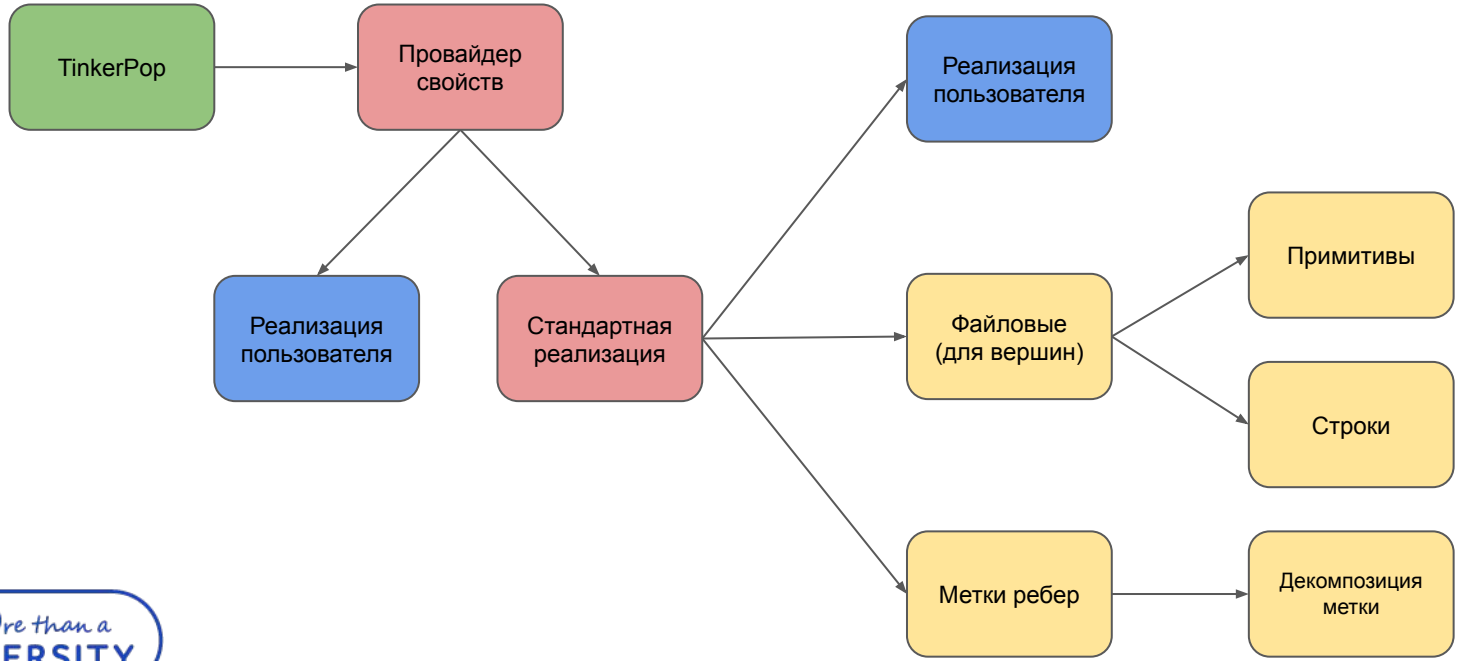
WebGraph



TinkerPop



Универсальный способ управлениями свойствами



Анализ производительности

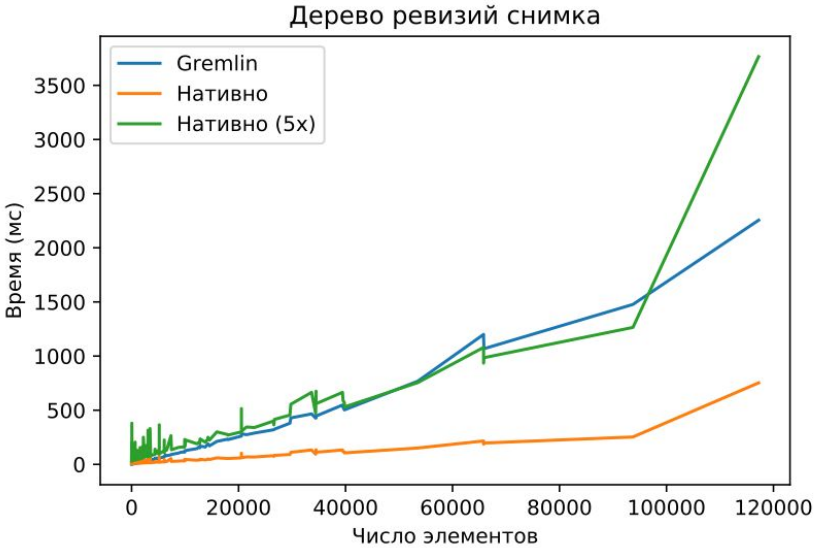
Домен
- Репозитории с открытым исходным кодом (git)

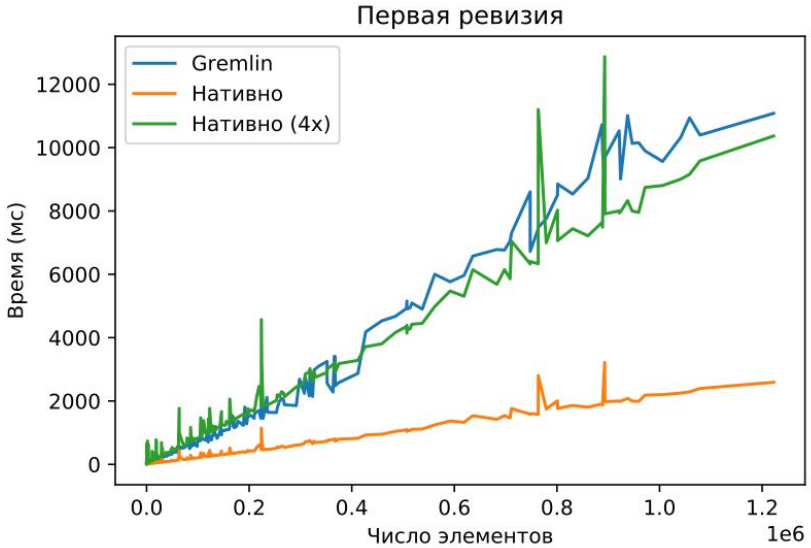
Набор данных
- python3k
- 46 млн вершин
- 1.2 млрд ребер

Запросы
- Первая ревизия для файла/директории
- Список файлов ревизии (ls -R)
- Дерево ревизий снимка (git log)

Структура архива

Анализ производительности





Анализ производительности

Набор данных	Число вершин	Число ребер	Средняя степень вершины	Замедление
imdb-2021	2 996 317	10 739 291	3.584	4.71
python3k	45 691 499	1 218 489 224	26.668	4.92
hollywood-2011	2 180 759	228 985 632	105.003	2.05

Сравнение библиотеки и нативного подхода

+ Сокращение размера и сложности запросов
+ Удобная работа с ребрами и свойствами
+ Исполнение строковых запросов
- Замедление исполнения в 4-5 раз

```
g.V().not(in())
 .repeat(out().dedup())
 .until(not(out()))
```



```
public Set<Long> leaves(SwhBidirectionalGraph g) {
    NodeIterator nodes = g.nodeIterator();
    Set<Long> roots = new HashSet<>();
    while (nodes.hasNext()) {
        long cur = nodes.nextLong();
        if (g.predecessors(cur).nextLong() == -1) {
            roots.add(cur);
        }
    }
    Set<Long> leaves = new HashSet<>();
    boolean[][] used = BooleanBigArrays.newBigArray(g.numNodes());
    for (Long root : roots) {
        dfs(root, used, leaves, g);
    }
    return leaves;
}

private void dfs(long root, boolean[][] used, Set<Long> leaves,
BidirectionalImmutableGraph g) {
    BigArrays.set(used, root, true);
    LazyLongIterator successors = g.successors(root);
    long child;
    boolean hasChild = false;
    while ((child = successors.nextLong()) != -1) {
        hasChild = true;
        if (!BigArrays.get(used, child)) {
            dfs(child, used, leaves, g);
        }
    }
    if (!hasChild) {
        leaves.add(root);
    }
}
```

Полученные результаты

- Разработана библиотека, связывающая Gremlin и WebGraph
  - Запуск Gremlin запросов на любых графах, сжатых WebGraph
  - Минимальная настройка благодаря простому добавлению свойств
  - Исполнение коротких и читаемых графовых запросов
  - Внедрение в проект Software Heritage

- Проведен анализ производительности
  - Замедление по сравнению с нативными запросами в 4–5 раз

andreystar2403@gmail.com

github.com/andrey-star/webgraph-tinkerpop
github.com/andrey-star/swh-graph-tinkerpop

ITMOre than a UNIVERSITY