Улучшение производительности алгоритма поиска путей с контекстно-свободными ограничениями для графовой базы данных Neo4j

Автор: Погожельская Влада Владимировна, 18.Б11-мм
Научный руководитель: к. ф.-м. н., доцент кафедры информатики Григорьев С. В.

Санкт-Петербургский государственный университет
Кафедра системного программирования

11 июня 2021г.

Введение

- Графовая модель представления данных
  - Основные сущности — вершины графа
  - Взаимосвязи между сущностями хранятся в самой графовой модели
- Графовые базы данных
  - Одной из наиболее распространенных является Neo4j
  - Для запросов поддерживаются только частично регулярные грамматики
- Контекстно-свободные ограничения
  - Строго расширяют выразительность запросов по сравнению с регулярными ограничениями
  - Имеют широкое применение в биоинформатике, анализе RDF-файлов

Задача поиска путей с контекстно-свободными ограничениями

Задача поиска путей и задача достижимости с контекстно-свободными ограничениями

Дано:

- *Контекстно-свободная грамматика* $\mathbb{G} = \langle N, \Sigma, P, S \rangle$
- *Ориентированный граф* $\mathbb{D} = \langle V, E, T \rangle$
- *Множество стартовых вершин* $V_S \subseteq V$ *и финальных вершин* $V_F \subseteq V$

***Задача поиска путей**:*

- *Найти все такие пути* $\pi = (e_0, e_1, \cdots, e_{n-1}, e_n)$, $e_k = (v_{k-1}, t_k, v_k)$ *в графе* $\mathbb{D}$, *что* $l(\pi) = t_1 t_2 \cdots t_n \in L(\mathbb{G})$ *и* $v_0 \in V_S$, $v_n \in V_F$

***Задача достижимости**:*

- *Найти множество пар* $\{(v_i, v_j) \mid \exists\, l(\pi) \in L(\mathbb{G})$ *и* $v_0 \in V_S$, $v_n \in V_F\}$

Применение Generalized LL для запросов с контекстно-свободными ограничениями

- Классический алгоритм синтаксического анализа Generlized LL был обобщен для выполнения контекстно-свободных запросов на графах[1]
- На реальных данных алгоритм в большинстве случаев дал существенный прирост в производительности
- При проведении экспериментального исследования выявлено неожиданное ухудшение в поведении полученного решения
- На практике восстановление самих путей в графе не всегда требуется

[1] Дипломная работа Власовой А.С.: `https://se.math.spbu.ru/thesis/texts/Vlasova_Anna_Sergeevna_Bachelor_Thesis_2020_text.pdf`

Цели и задачи

Целью данной работы является улучшение производительности алгоритма поиска путей с контекстно-свободными ограничениями для графовой базы данных Neo4j

Задачи:

- Провести анализ и рефакторинг кода с целью выявления и устранения проблем производительности текущей реализации GLL алгоритма
- Добавить возможность отключения построения путей и возврата информации лишь о достижимости в графе
- Провести экспериментальное исследование на реальных данных и сравнить полученное решение с уже существующим

Обзор существующего решения

Обобщенный LL-алгоритм (GLL)

- Поддерживает весь класс контекстно-свободных языков
- Для восстановления путей поддерживается сжатое представление леса разбора (SPPF)

Реализация

- Решение базируется на реализации GLL в библиотеке Iguana[2], написанной на Java
- В качестве хранилища графов использована графовая база данных Neo4j
- Полученное решение интегрировано с Neo4j при помощи Native Java API

[2] Репозиторий библиотеки Iguana: `https://github.com/iguana-parser/iguana`

Экспериментальное исследование существующего решения

Графы:

- Enzyme — граф о белковых последовательностях (48 тыс. вершин и 86 тыс. ребер);
- Geospecies — граф о таксономической иерархии видов животных (450 тыс. вершин и 2.2 млн ребер).

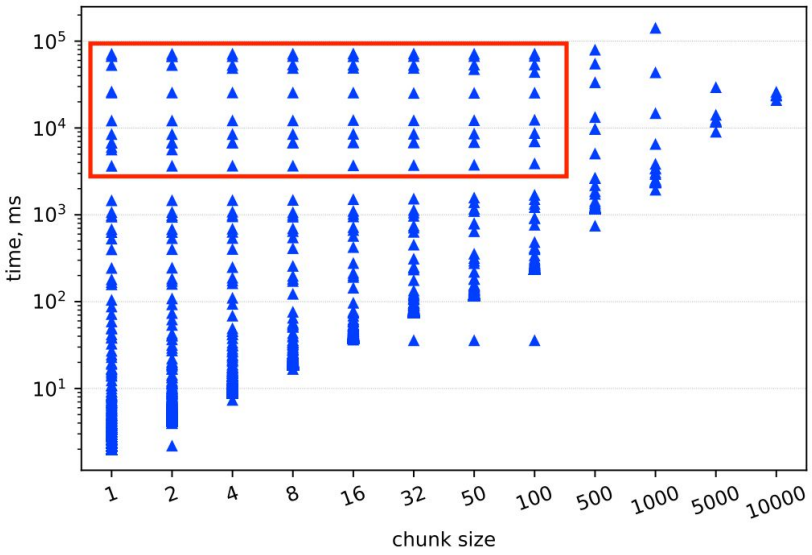
Грамматики:

$$\begin{aligned} S \to & \overline{subClassOf}\ \ S\ \ subClasOf \mid \overline{type}\ \ S\ \ type \\ & \mid \overline{subClassOf}\ \ subClassOf \mid \overline{type}\ \ type \end{aligned} \tag{1}$$

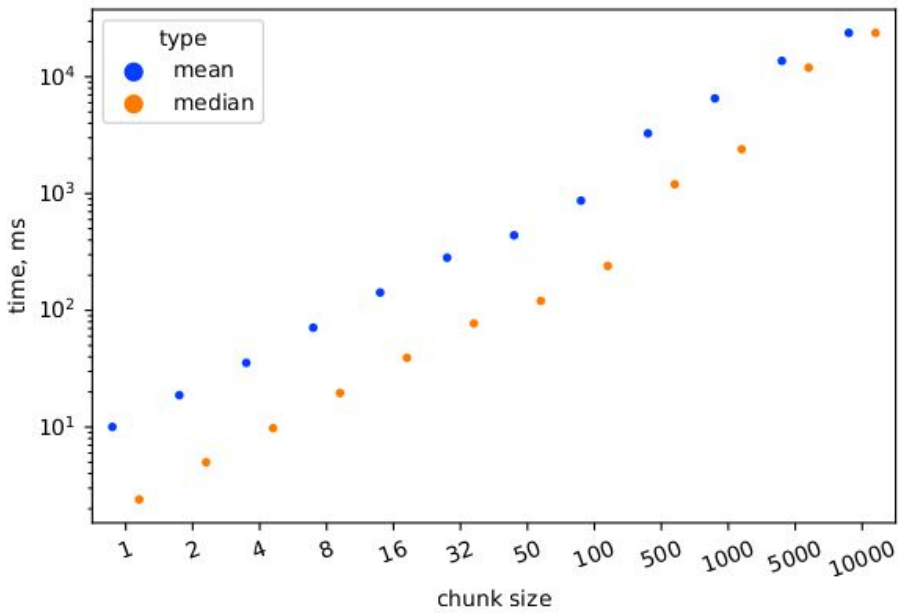
$$S \to \overline{subClassOf}\ \ S\ \ subClasOf \mid subClassOf \tag{2}$$

$$\begin{aligned} S \to & broaderTransitive\ \ S\ \ \overline{broaderTransitive} \\ & \mid broaderTransitive\ \ \overline{broaderTransitive} \end{aligned} \tag{3}$$

Экспериментальное исследование существующего решения



(a) время выполнения запросов



(b) медиана и среднее время выполнения запросов

Рис.: Грамматика G_2 на Enzyme

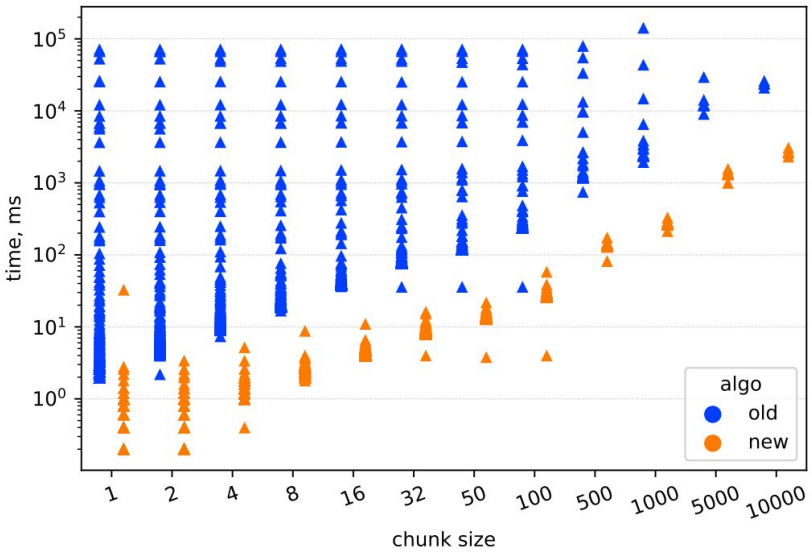
«Узкие» места существующего решения

- Наибольшее количество процессорного времени тратится на сопоставление текущего входа с терминалом грамматики и получение меток на ребрах
- Результат обращения к базе данных в явном виде сохраняется в список
- Практически всё процессорное время тратится на вычисления внутри базы данных

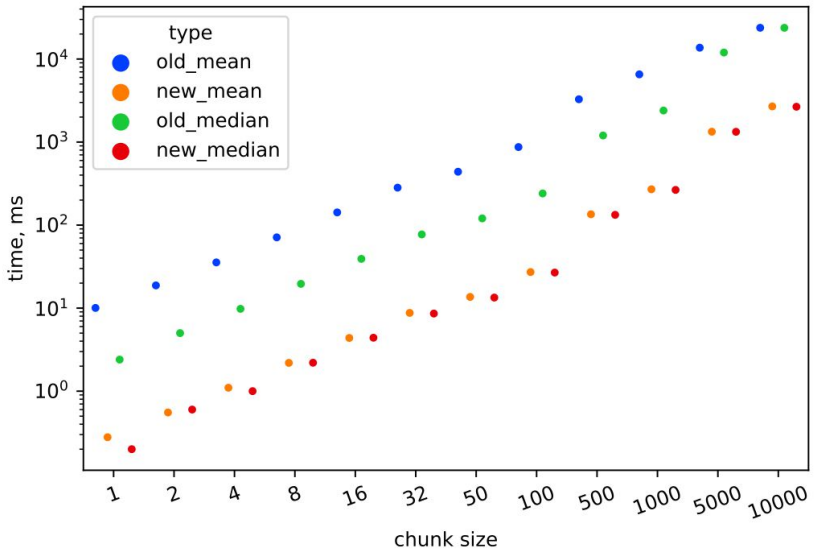
Предложенное решение

- Native Java API предоставляет способ получить итератор над множеством исходящих из вершины ребер
- После сопоставления входа с терминалом, возможно, далеко не все метки понадобятся для дальнейшей работы алгоритма
- Stream API позволяет обеспечить потоковую обработку данных, извлекаемых посредством обращения к Neo4j

Экспериментальное исследование предложенной реализации



(a) время выполнения запросов



(b) медиана и среднее время выполнения запросов

Рис.: Грамматика G_2 на Enzyme

Модификация алгоритма GLL

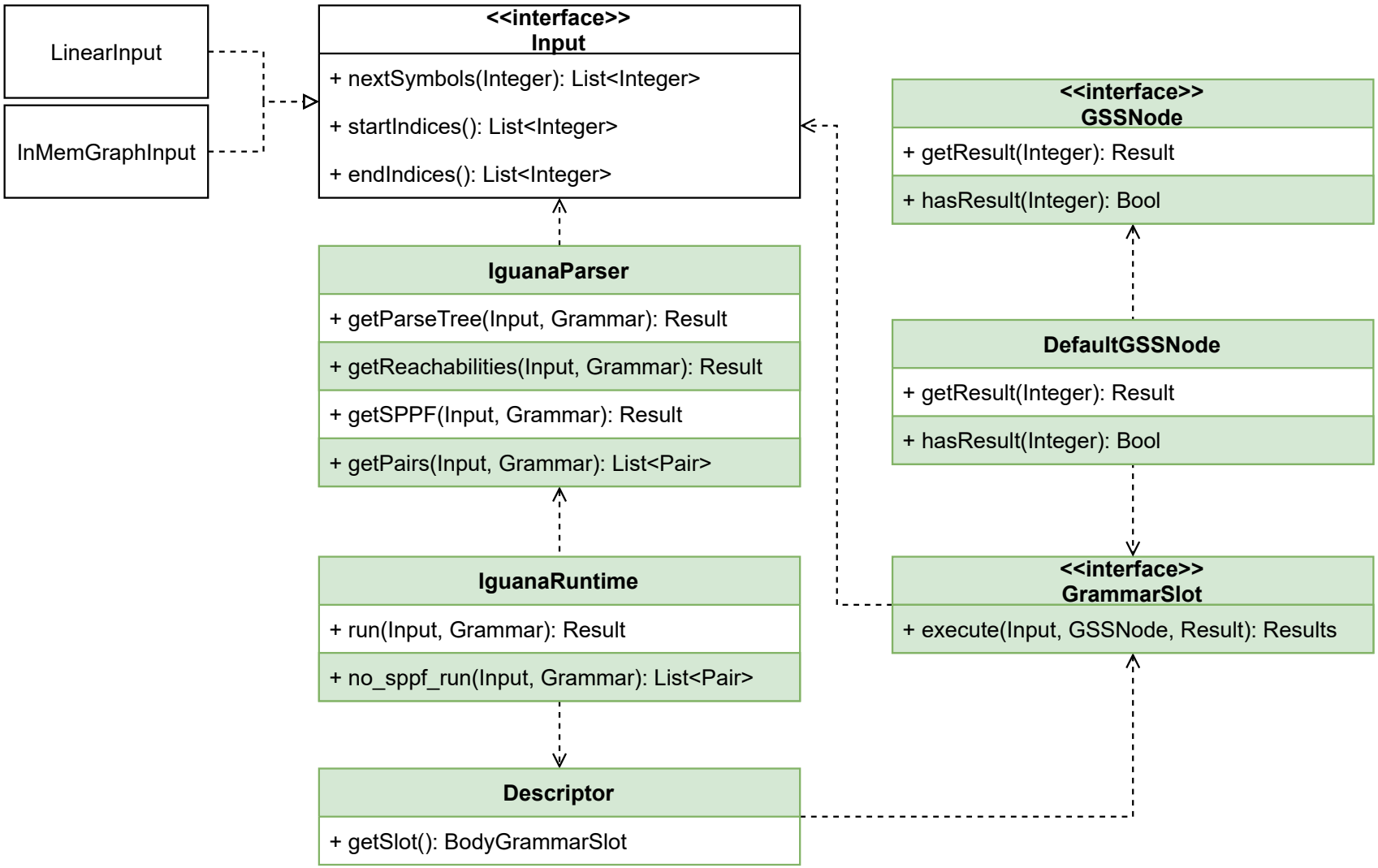
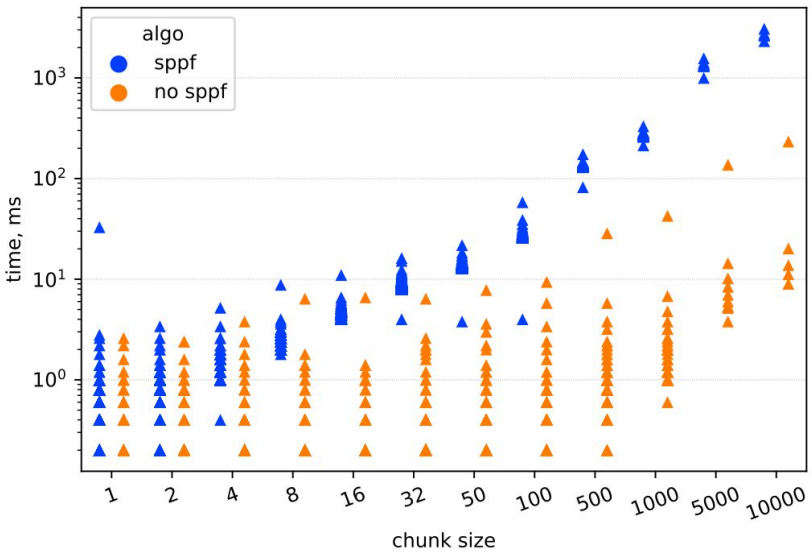
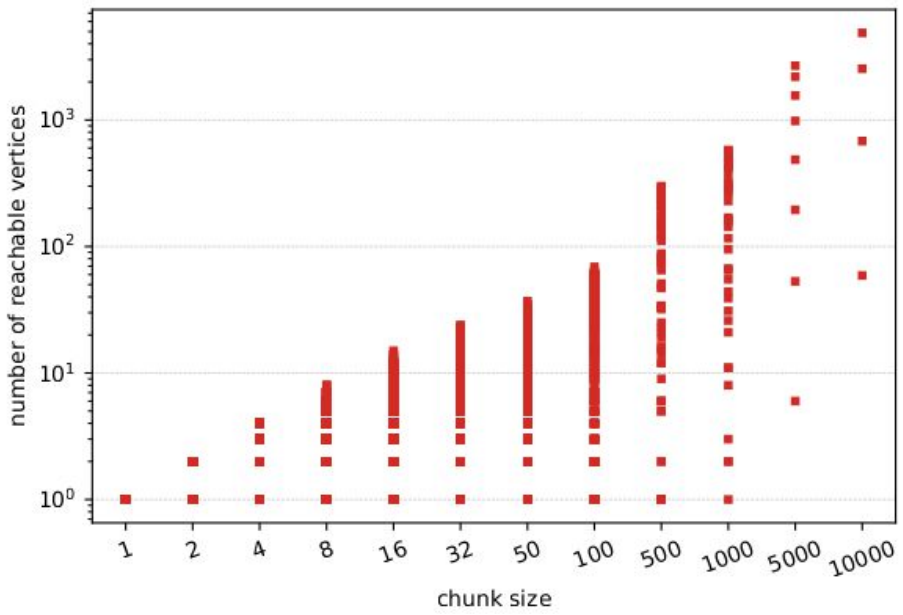


Рис.: Диаграмма классов Iguana после модификаций для поддержки опции отключения SPPF

Экспериментальное исследование модифицированного алгоритма



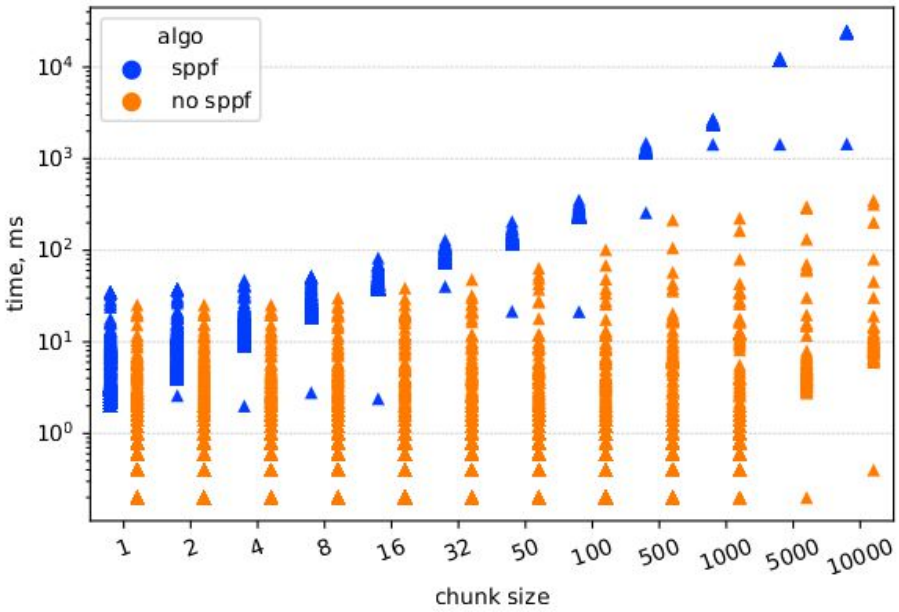
(a) время выполнения запросов



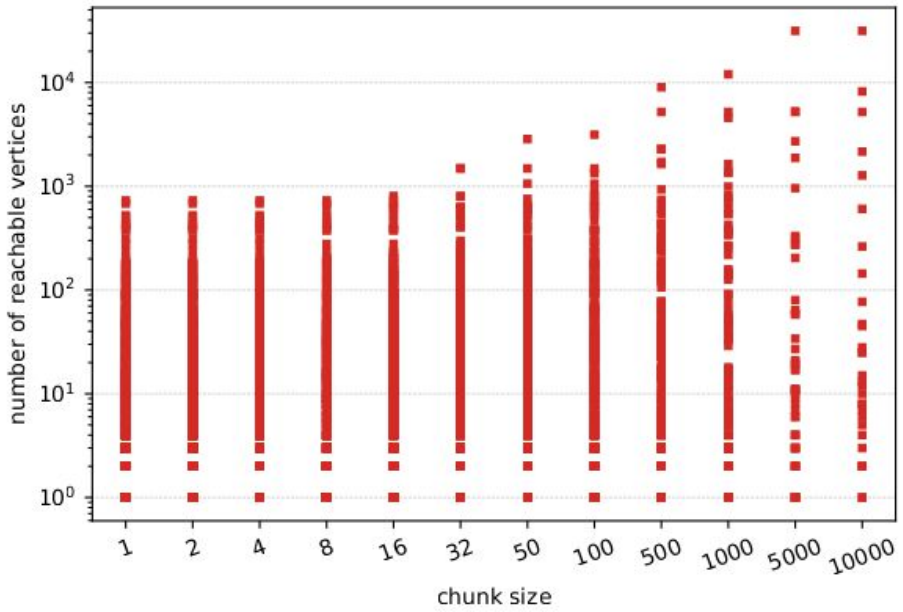
(b) размер ответов на запросы

Рис.: Грамматика G_2 на Enzyme

Экспериментальное исследование модифицированного алгоритма



(a) время выполнения запросов



(b) размер ответов на запросы

Рис.: Грамматика G_3 на Geospecies

Выводы

- Модифицированный алгоритм GLL без построения SPPF может быть эффективно применен для решения задачи поиска достижимости в графе с контекстно-свободными ограничениями на реальных данных
- Полученные результаты делают актуальными дальнейшие исследования, направленные как на улучшение данного алгоритма и реализации, так и на полноценную интеграцию его в графовую базу данных Neo4j

Заключение

В рамках производственной практики были выполнены следующие задачи

- Проведен анализ и рефакторинг кода
- Выявлены и устранены проблем производительности текущей реализации GLL алгоритма
- Добавлена возможность отключения построения SPPF и возврата информации лишь о достижимости в графе
- Проведено экспериментальное исследование на реальных данных и сравнение полученного решения с уже существующим