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

Автор: Анна Сергеевна Власова, группа 16.Б10
Научный руководитель: доцент, к.ф.-м.н. С.В. Григорьев
Рецензент: программист ООО "ИнтеллиДжей Лабс" Р.Ш. Азимов

Санкт-Петербургский государственный университет
Кафедра системного программирования

Введение

- Графовые базы данных
  - Одной из самых распространенных является Neo4j
  - Для запросов поддерживаются только регулярные грамматики
- Поиск путей в графах с контекстно-свободными ограничениями
  - КС грамматика $\mathbb{G} = (S, \Sigma, N, P)$
  - Ориентированный граф $G = (V, E, T)$, $T \subseteq \Sigma$, $E \subseteq V \times T \times V$
  - Задача: найти такие пути, что конкатенация меток их ребер лежит в $L(\mathbb{G})$

Область применения

- Анализ RDF данных
- Статический анализ кода
  - Межпроцедурный анализ указателей
  - Анализ алиасов
  - Анализ типов
- Биоинформатика
  - Анализ геномных последовательностей
- Сегментация графов в задаче происхождения данных

Существующие решения задачи КС-запросов

- Матричные алгоритмы
  - Высокая производительность
  - Большой расход памяти
  - Решают только задачу достижимости
- Парсер-комбинаторы для КС-запросов
  - Интегрирован с Neo4j (Meerkat)
  - Низкая производительность на больших графах
- Решения, основанные на алгоритмах синтаксического анализа
  - Generalized LL
  - Generalized LR

Библиотека Iguana

- Реализация GLL с оптимизациями
- Высокая производительность
- Реализована на Java
- Удобно интегрировать с Neo4j

Постановка задачи

Целью работы является реализация поддержки запросов с контекстно-свободными ограничениями для графовой базы данных Neo4j

Задачи:

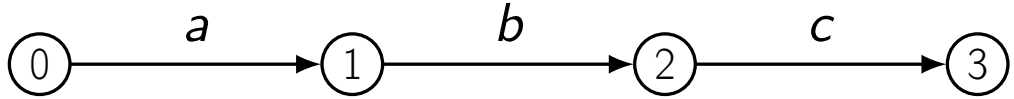
- Модифицировать библиотеку Iguana для поддержки входных данных, представленных в виде графа
- Интегрировать расширенную версию Iguana с графовой базой данных Neo4j
- Провести экспериментальное исследование

Обобщенный LL-алгоритм (GLL)

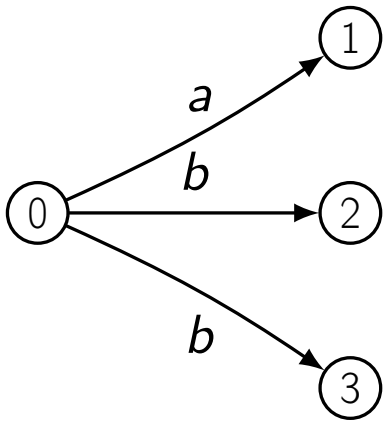
- Поддерживает любые контекстно-свободные грамматики
- Состояние синтаксического анализа описывается дескриптором:
  - Позиция во входной строке
  - Слот — позиция в грамматике, например, $S ::= A \ . \ b$
  - Стек
  - Представление результата разбора
- Для обработки всех состояний дескрипторы хранятся в очереди
- Строит сжатое представление леса разбора
- Обобщается с линейного входа на графы

Модификация Iguana: входные данные

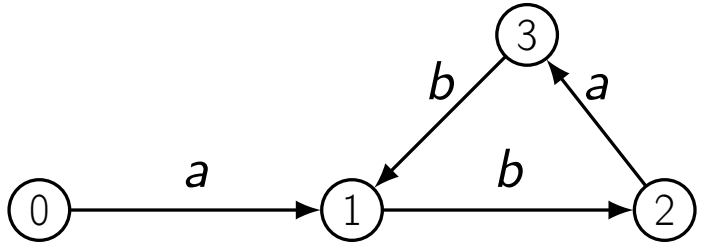
- Строка — линейный граф



- Но у графа есть отличия



(a) Ветвления



(b) Циклы

Модификация Iguana

- Ветвления
  - Порождение нескольких состояний из текущего
- Циклы
  - Переиспользование вычисленных результатов

Архитектура Iguana после модификаций

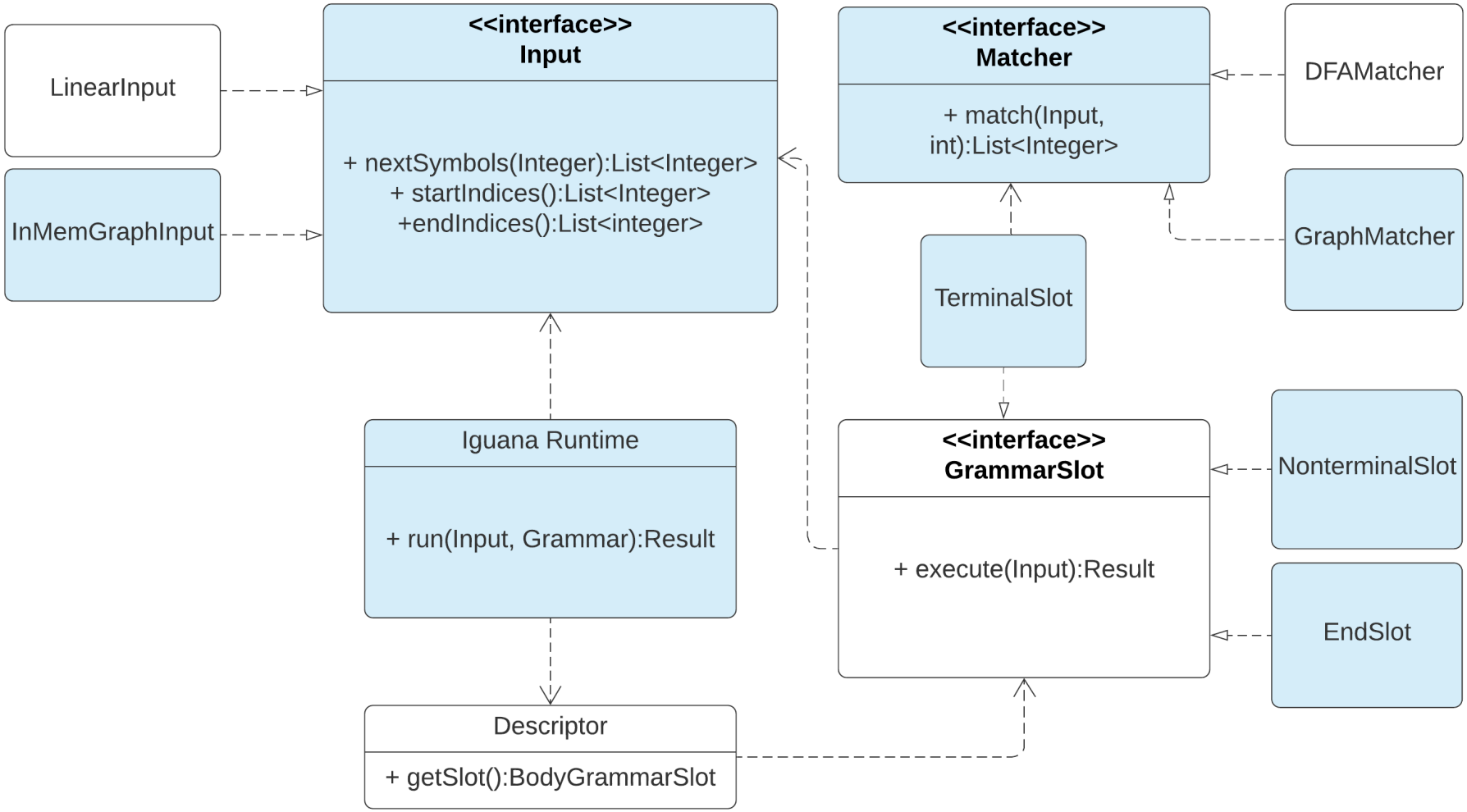


Рис.: Диаграмма основных классов

Интеграция с Neo4j

- Используется Java API
- Создается встроенная база данных

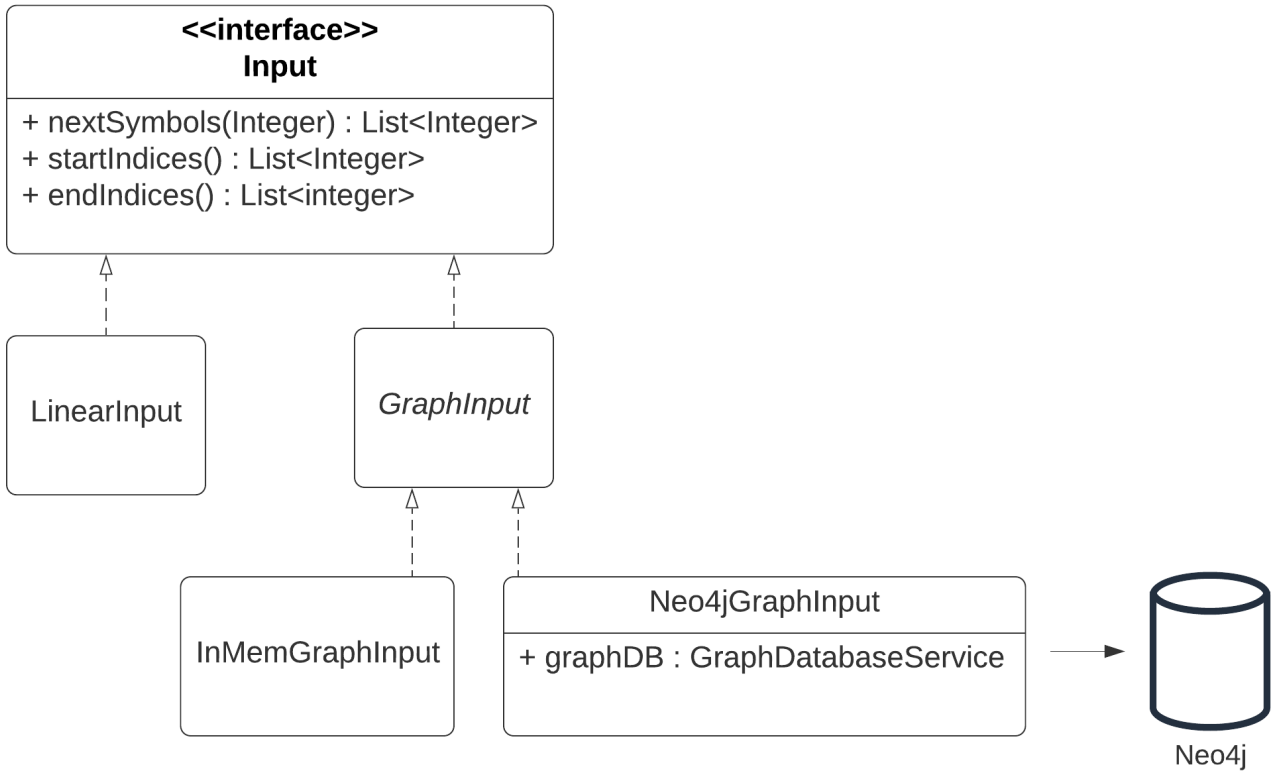
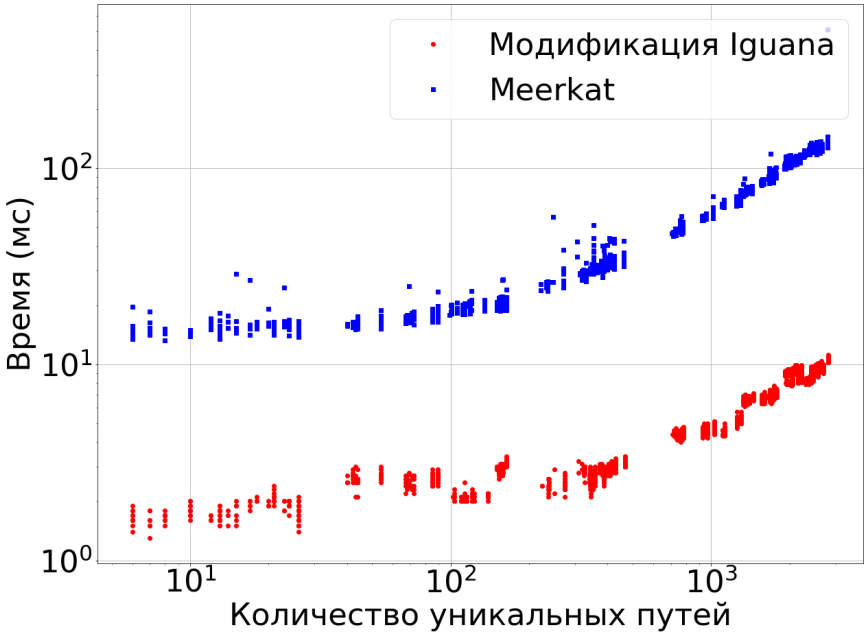


Рис.: Диаграмма классов, представляющих входные данные

Экспериментальное исследование

- Графы:
  - Enzyme (15 тыс. вершин, 47 тыс. ребер) — белковые последовательности
  - Geospecies (225 тыс. вершин, 1.6 млн ребер) — иерархия видов животных
- Грамматики
  - Same generation query — поиск всех видов на том же уровне иерархии (одно поколение)
    Грамматика $S ::= aSb \mid \varepsilon$ задает слова вида $a^n b^n$

Экспериментальное исследование: Enzyme



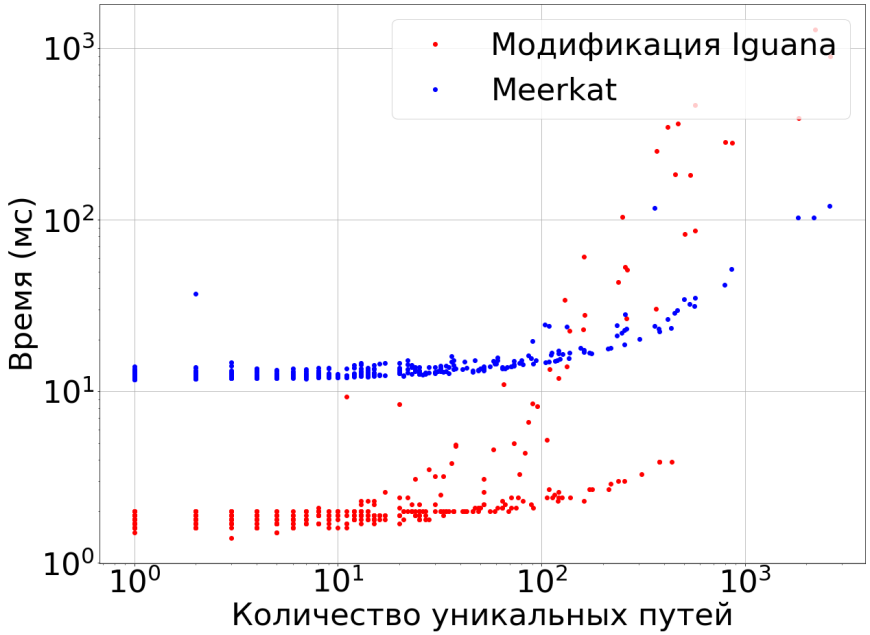
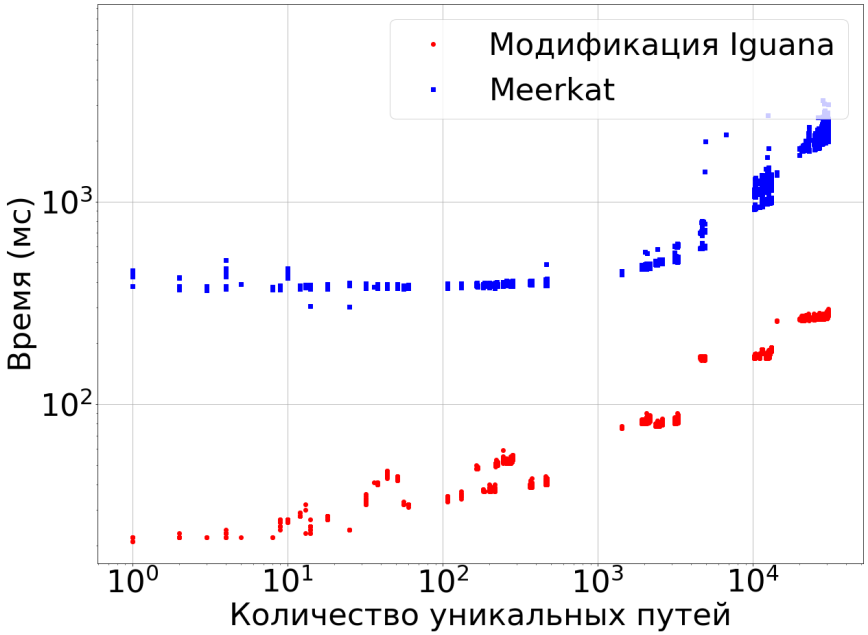


Рис.: Сравнение производительности на двух различных запросах

Результаты

- Библиотека Iguana расширена для поддержки входных данных, представленных как в виде графа, так и строк
- Расширенная версия Iguana интегрирована с графовой базой данных Neo4j
- Проведено экспериментальное исследование и сравнение с Meerkat

Экспериментальное исследование: Geospecies



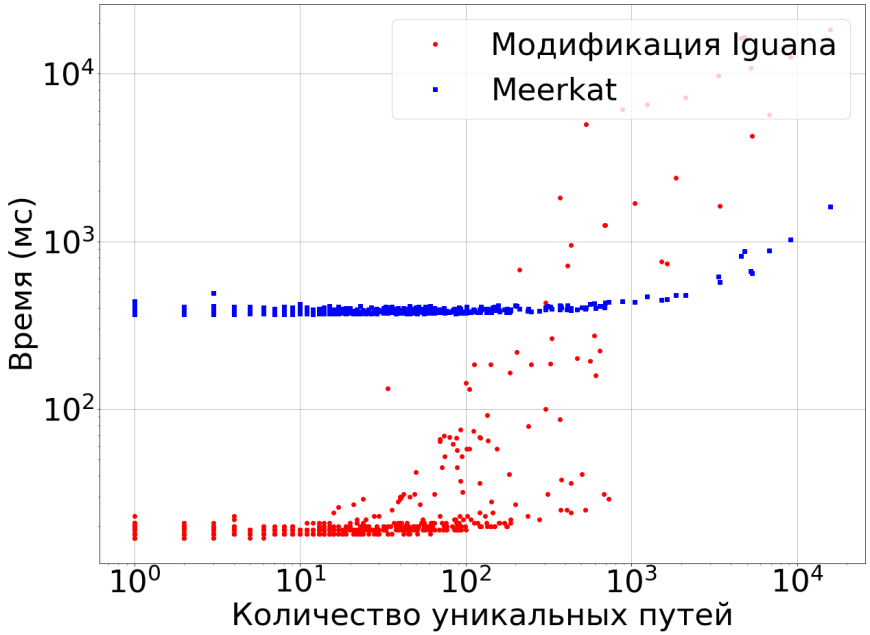


Рис.: Сравнение производительности на двух различных запросах