Реализация и экспериментальное исследование GLL-парсера, основанного на рекурсивном автомате

Автор: Абзалов Вадим Игоревич, 19.Б11-мм
Научный руководитель: к. ф.-м. н., доцент кафедры информатики Григорьев С. В.

Санкт-Петербургский государственный университет
Кафедра системного программирования

19 мая 2023г.

Введение

- Графовая модель представления данных
  - Основные сущности — вершины графа
  - Взаимосвязи между сущностями хранятся в самой графовой модели
- Ограничения в виде формальных грамматик
  - Наибольшее применение находят регулярные и контекстно-свободные грамматики
  - Имеют широкое применение в биоинформатике, анализе RDF-файлов
- GLL алгоритм не позволяет напрямую работать с регулярными выражениями
- Рекурсивные автоматы поддерживают как регулярные, так и контестно-свободные грамматики

Задача поиска путей 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 поиска достижимостей**:*

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

Цели и задачи

Целью данной работы является реализация и экспериментальное исследование производительности GLL алгоритма, основанного на рекурсивном автомате
Задачи:

- Реализация классического GLL алгоритма
- Модификация алгоритма GLL для поддержки представления грамматики в виде рекурсивного автомата
- Расширение модифицированного алгоритма GLL на входные данные в виде графа
- Проведение экспериментального исследования и сравнение с существующими решениями

Обзор GLL алгоритма

Обобщенный LL-алгоритм (GLL)

- Поддерживает весь класс контекстно-свободных языков
- Для восстановления путей поддерживается сжатое представление леса разбора (SPPF)
- Обобщается на входные данные в виде графов

Архитектура

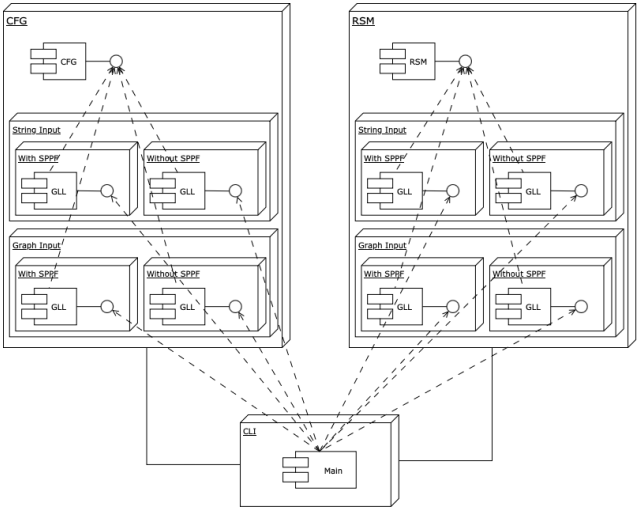


Рис.: Архитектура проекта

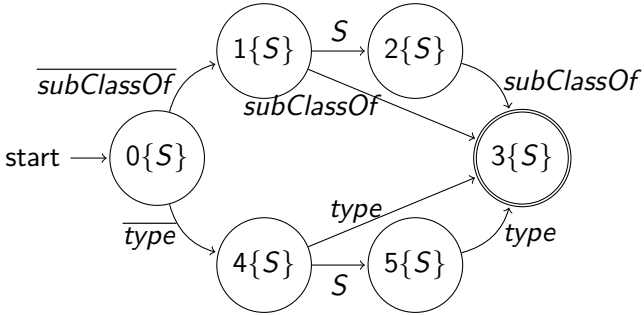
Экспериментальное исследование полученного решения

Название графа	$\|V\|$	$\|E\|$	#subClassOf	#type	#broaderTransitive
enzyme	48815	86543	8163	14989	8156
eclass	239111	360248	90962	72517	0
go_hierarchy	45007	490109	490109	0	0
go	582929	1437437	94514	226481	0
geospecies	450609	2201532	0	89062	20867
taxonomy	5728398	14922125	2112637	2508635	0

Рис.: Графы класса RDF: количество вершин, ребер и ребер с определенными метками

Экспериментальное исследование полученного решения

$$S \to \overline{subClassOf}\ \ S\ \ subClassOf \mid \overline{type}\ \ S\ \ type \mid \overline{subClassOf}\ \ subClassOf \mid \overline{type}\ \ type \qquad (G_1)$$



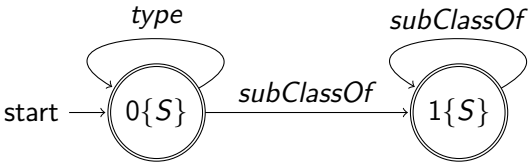
(A_1)

Рис.: Контекстно-свободная грамматика G_1 и соответствующий ей рекурсивный автомат A_1

Экспериментальное исследование полученного решения

$$type^*\ subClassOf^* \tag{R_4}$$

$$\begin{aligned} S &\to A\ B \\ A &\to type\ A \mid \varepsilon \\ B &\to subClassOf\ B \mid \varepsilon \end{aligned} \tag{G_{R_4}}$$



(A_{R_4})

Рис.: Регулярное выражение R_4, соответствующие ему контекстно-свободная грамматика G_{R_4} и рекурсивный автомат A_{R_4}

Результаты

Название графа	Время в секундах				
	G1			R4	
	CFG	RSM	GLL4Graph	CFG	RSM
enzyme	0.107 ± 0.007	0.044 ± 0.008	0.22 ± 0.01	0.31 ± 0.08	0.0577 ± 0.0102
eclass	0.94 ± 0.14	0.43 ± 0.07	1.5 ± 0.03	2.12 ± 0.27	0.46 ± 0.09
go_hierarchy	4.1 ± 0.6	3.0 ± 0.4	3.6 ± 0.2	1.81 ± 0.09	0.4 ± 0.08
go	3.2 ± 0.3	1.86 ± 0.16	5.55 ± 0.08	7.4 ± 2.7	1.6 ± 0.2
geospecies	0.97 ± 0.12	0.34 ± 0.04	2.89 ± 0.6	3.1 ± 0.6	0.521 ± 0.109
taxonomy	31.2 ± 1.5	14.8 ± 0.6	45.4 ± 0.7	OOM	OOM

Рис.: Результаты экспериментов на задаче поиска достижимостей в графе

Выводы

- В подавляющем большинстве случаев использование рекурсивного автомата дало положительный результат на времени работы GLL алгоритма
- Реализованная модификация оказывается в несколько раз эффективнее реализации GLL алгоритма в проекте GLL4Graph на большинстве графов
- Получена не только эффективная реализация GLL алгоритма с использованием рекурсивного автомата, но и классического GLL алгоритма

Заключение

В рамках данной работы были выполнены следующие задачи:

- Реализован классический GLL алгоритм
- Алгоритм GLL был модифицирован для поддержки представления грамматики в виде рекурсивного автомата
- Реализовано расширение модификации алгоритма GLL на входные данные в виде графа
- Проведено экспериментальное исследование, по результатам которого можно сделать выводы о том, что полученная модификация алгоритма GLL работает существенно эффективнее классического алгоритма GLL

Реализация представлена в репозитории: `https://github.com/vadyushkins/kotgll`.