Санкт-Петербургский государственный университет

Кафедра системного программирования

Ковалев Дмитрий Александрович

Синтаксический анализ данных, представленных в виде контекстно-свободной грамматики

Выпускная квалификационная работа

Научный руководитель:
к. ф.-м. н., доц. Григорьев С. В.

Рецензент:
программист НИУ ИТМО Авдюхин Д. А.

Санкт-Петербург
2017

SAINT PETERSBURG STATE UNIVERSITY

Chair of Software Engineering

Dmitry Kovalev

Parsing of the data represented as context free grammar

Graduation Project

Scientific supervisor:
associate professor Semyon Grigorev

Reviewer:
Programmer at ITMO University Avdiukhin Dmitrii

Saint-Petersburg
2017

Оглавление

Введение

Контекстно-свободные грамматики, наряду с регулярными выражениями, повсеместно используются для решения задач, связанных с разработкой формальных языков и синтаксических анализаторов. Одним из основных достоинств контекстно-свободных грамматик является возможность задания широкого класса языков при сохранении относительной компактности представления. Благодаря данному свойству, грамматики также представляют интерес в такой области информатики, как кодирование и сжатие данных. В частности, существует ряд алгоритмов, позволяющих производить сжатие текстовой информации, используя в качестве конечного [8] или промежуточного [3] представления контекстно-свободную грамматику (grammar-based compression).

Использование компактного представления текста может быть актуально для некоторых задач биоинформатики. Примером такой задачи является анализ метагеномных сборок — помеченных графов большого размера: порядка 10^6 ребер и 10^6 вершин. Метками на ребрах графа являются строки над алфавитом нуклеотидных символов. В магистерской диссертации Анастасии Рагозиной [9] был предложен алгоритм, позволяющий искать в метагеномных сборках шаблоны РНК, задаваемые при помощи контекстно-свободной грамматики. Производительность данного алгоритма предположительно может быть увеличена за счет контекстно-свободного сжатия меток на ребрах графа и трансформации его в грамматику, представленную в расширенной форме Бэкуса-Наура.

Задача поиска шаблонов при использовании подобного представления метагеномной сборки формулируется следующим образом: необходимо найти все строки, принадлежащие пересечению языков, задаваемых грамматикой шаблона и грамматикой сборки. Или, иначе, — найти все строки, порождаемые одной грамматикой и выводимые в другой. Назовем эту задачу *синтаксическим анализом данных, представленных в виде КС-грамматики*. В общем случае такая задача неразрешима, так как она сводится к задаче о проверке пересечения двух

языков, порождаемых произвольными КС-грамматиками, на пустоту. Для проведения экспериментов с метагеномными сборками необходимо точнее исследовать возможность проведения синтаксического анализа КС-представления и разработать прототип алгоритма, позволяющего решить данную задачу.

1. Постановка задачи

Целью данной работы является разработка алгоритма синтаксического анализа данных, представленных в виде контекстно-свободной грамматики. Для ее достижения были поставлены следующие задачи.

- Изучить существующие подходы к анализу данных, представленных в виде КС-грамматик.
- Определить ограничения, при которых синтаксический анализ такого представления является разрешимой задачей.
- Разработать алгоритм синтаксического анализа КС-представления данных с учетом поставленных ограничений и доказать его завершаемость.
- Реализовать предложенный алгоритм.
- Провести тестирование и апробацию.

2. Обзор

2.1. Подходы к анализу данных, представленных в виде КС-грамматики

Пусть G — произвольная КС-грамматика, M — конечный автомат. Тогда задача проверки

- включения языков ($L(M) \subseteq L(G)$) — неразрешима
- пустоты пересечения ($L(M) \cap L(G) = \emptyset$) — разрешима (т.к. в пересечении не более чем КС-язык) за полиномиальное время [6]
- регулярности языка $L(G)$ — неразрешима [5]

Если использовать представление регулярного языка $L(M)$ в виде КС-грамматики G_r, то задача проверки пустоты пересечения ($L(G_r) \cap L(G) = \emptyset$) становится немного интереснее: если G_r

- нерекурсивная — задача из PSPACE [7] (точнее результата нет (я не нашел, по крайней мере))
- лево- или праволинейная — ничего не известно (см. последний абзац заключения из [7])
- принадлежит еще более широкому классу — тем более ничего не известно

Еще немного про вложенную рекурсию и регулярность языка. Грамматика без вложенной рекурсии (NSE) порождает регулярный язык [4] (обратное тоже верно, для регулярного языка можно построить NSE грамматику, т.к. праволинейная, например, — частный случай NSE). Существует алгоритм, который позволяет проверять грамматику на наличие вложенной рекурсии за полином [2]. Однако, грамматика с вложенной рекурсией тоже может порождать регулярный язык [1], поэтому задача о проверке регулярности языка, порождаемого КС-грамматикой, остается неразрешимой.

2.2. Обобщенный синтаксический анализ

2.3. Алгоритм GLL и его модификации

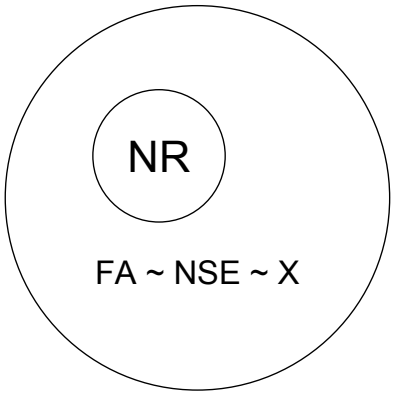
2.4. Проект YaccConstructor

3. Алгоритм синтаксического анализа контекстно-свободного представления

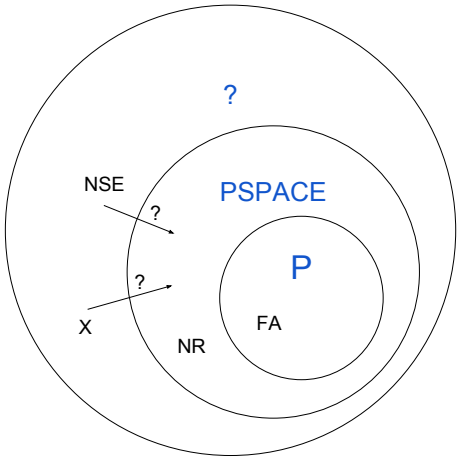
Мы пытаемся решать следующую задачу: пусть $G_1 = (N, T, S, P)$ — произвольная КС-грамматика, G_2 — NSE КС-грамматика. Алгоритм принимает на вход два рекурсивных автомата, M_1 и M_2, построенных по грамматикам G_1 и G_2 соотвественно, при этом в автомате M_2 левые/правые рекурсивные вызовы заменены на циклы, как в обычном конечном автомате.

Результатом работы алгоритма являются тройки вида (X, n_1, n_2), где $X \in N$, n_1, n_2 — номера состояний автомата M_2. Для каждой из таких троек выполняется следующее утверждение: $\exists\, \omega \in T^*$ такая, что $X \to^* \omega$ в G_1 и $\omega \in L(M')$, где M' — рекурсивный автомат, полученный из M_2 путем замены начального и конечного состояния на n_1 и n_2 соответственно.

Получая такие результаты, мы, по сути, отвечаем на вопрос о проверке пустоты пересечения КС-языка и регулярного, представленных в необычных абстракциях. Для КС-языка мы используем рекурсивный автомат, а для регулярного — нечто среднее между конечным автоматом и NSE грамматикой (это нечто все еще использует стек, но только для обработки нерекурсивных вызовов). Такое представление эквивалентно по выразительности NSE и, следовательно, FA (см. рис. 1a). Но неизвестно, к какому классу сложности относится задача (и разрешима ли вообще) о проверке пустоты пересечения регулярного языка, представленного в данной форме, с КС-языком (см. рис. 1b).



(a) Выразительность языка, задаваемого абстракцией



(b) Класс сложности для задачи о проверке пустоты пересечения с КС-языком

Рис. 1: Красивые круги. Здесь NR — нерекурсивная грамматика, FA — конечный автомат, NSE — грамматика без вложенной рекурсии, X — наше представление

4. Доказательство корректности алгоритма

Theorem 1 *Завершаемость. Алгоритм завершает работу за конечное число шагов для произвольных входных данных*

Theorem 2 *Корректность. ???*

5. Реализация и тестирование

5.1. Архитектура предложенного решения

text

5.2. Тестирование производительности

text

Заключение

В ходе данной работы получены следующие результаты.

- Изучена предметная область — существующие подходы к анализу данных, представленных в виде КС-грамматик.
- Определены ограничения, при которых синтаксический анализ контекстно-свободного представления является разрешимой задачей.
- Разработан алгоритм синтаксического анализа КС-представления, учитывающий поставленные ограничения, и доказана его завершаемость.
- Предложенный алгоритм реализован на языке программирования F# в рамках исследовательского проекта YaccConstructor.
- Проведена апробация: тестирование производительности и тестирование на синтетических данных.

В дальнейшем планируется провести апробацию на реальных данных (метагеномных сборках) и доказать теоретическую оценку сложности алгоритма.

Исходный код проекта YaccConstructor может быть найден на сайте `https://github.com/YaccConstructor/YaccConstructor`.

Список литературы

[1] Andrei Stefan, Chin Wei-Ngan, Cavadini Salvador Valerio. Self-embedded context-free grammars with regular counterparts // Acta Informatica. — 2004. — Vol. 40, no. 5. — P. 349–365. — Access mode: http://dx.doi.org/10.1007/s00236-003-0133-8.

[2] Anselmo Marcella, Giammarresi Dora, Varricchio Stefano. Finite Automata and Non-self-embedding Grammars // Proceedings of the 7th International Conference on Implementation and Application of Automata. — CIAA'02. — Berlin, Heidelberg : Springer-Verlag, 2003. — P. 47–56. — Access mode: http://dl.acm.org/citation.cfm?id=1756384.1756390.

[3] Arimura Mitsuharu. A grammar-based compression using a variation of Chomsky normal form of context free grammar // 2016 International Symposium on Information Theory and Its Applications (ISITA). — 2016.

[4] Chomsky Noam. A note on phrase-structure grammars // Information and Control. — 1959. — Vol. 2. — P. 393 – 395.

[5] Greibach Sheila. A note on undecidable properties of formal languages // Mathematical systems theory. — 1968. — Vol. 2, no. 1. — P. 1–6. — Access mode: http://dx.doi.org/10.1007/BF01691341.

[6] Hunt III Harry B., Rosenkrantz Daniel J., Szymanski Thomas G. On the Equivalence, Containment, and Covering Problems for the Regular and Context-free Languages // J. Comput. Syst. Sci. — 1976. — Apr. — Vol. 12, no. 2. — P. 222–268. — Access mode: http://dx.doi.org/10.1016/S0022-0000(76)80038-4.

[7] Nederhof Mark-Jan, Satta Giorgio. The Language Intersection Problem for Non-recursive Context-free Grammars // Inf. Comput. — 2004. — Aug. — Vol. 192, no. 2. — P. 172–184. — Access mode: http://dx.doi.org/10.1016/j.ic.2004.03.004.

[8] Nevill-Manning Craig G., Witten Ian H. Identifying Hierarchical Structure in Sequences: A Linear-time Algorithm // J. Artif. Int. Res. — 1997. — Sep. — Vol. 7, no. 1. — P. 67–82. — Access mode: http://dl.acm.org/citation.cfm?id=1622776.1622780.

[9] Ragozina Anastasiya. GLL-based relaxed parsing of dynamically generated code. — 2016.