Синтаксический анализ данных, представленных в виде контекстно-свободной грамматики

Автор: Ковалев Дмитрий Александрович, 444 группа
Научный руководитель: к. ф.-м. н. Григорьев С. В.
Рецензент: программист НИУ ИТМО Авдюхин Д. А.

Санкт-Петербургский государственный университет
Кафедра системного программирования

9 июня 2017 г.

Контекстно-свободное сжатие

- КС-грамматика как компактное представление данных

abcabdabcababcabdabcab

$$
\begin{aligned}
0 &::= 1\,1 \\
1 &::= 2\,d\,2 \\
2 &::= 3\,c\,3 \\
3 &::= a\,b
\end{aligned}
$$

- Известные алгоритмы: LZW, Sequitur, Sequential, Re-Pair, ...
- Грамматика порождает ровно одну строку (Straight-line grammar)
  - отсутствуют рекурсивные вызовы

Поиск шаблонов в сжатых данных

- Поиск шаблонов в КС-представлении без декомпрессии
- Шаблон может быть
  - строкой (compressed pattern matching)
  - 'straight-line' грамматикой (fully compressed pattern matching)
  - регулярным выражением

Поиск шаблонов в сжатых данных

- Поиск шаблонов в KC-представлении без декомпрессии
- Шаблон может быть
  - строкой (compressed pattern matching)
  - 'straight-line' грамматикой (fully compressed pattern matching)
  - регулярным выражением
- Шаблоны, описываемые KC-грамматикой:
  - поиск pPHK в сжатом геноме

Синтаксический анализ КС-представления

Определение

G_p, G_d — произвольные КС-грамматики; $L(G_d) = \{\omega_1, \omega_2, \ldots, \omega_k, \ldots\}$. Необходимо определить, существуют ли строки ω', для которых верно: $\omega' \in L(G_p)$ и ω' — подстрока одной из строк $\omega_i \in L(G_d)$.

- В общем случае задача неразрешима
  - сводится к задаче о проверке пустоты пересечения двух КС-языков

Постановка задачи

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

- Определить ограничения, при которых синтаксический анализ КС-представления является разрешимой задачей
- Разработать алгоритм синтаксического анализа КС-представления данных с учетом поставленных ограничений
- Реализовать предложенный алгоритм
- Провести экспериментальное исследование

Разрешимость задачи

Теорема (Nederhof, Satta, 2004 г.)

Пусть G_1 — произвольная KC-грамматика, G_2 — грамматика, которая не содержит непосредственной или скрытой рекурсий. Тогда проблема проверки $L(G_1) \cap L(G_2) = \emptyset$ относится к классу PSPACE-complete.

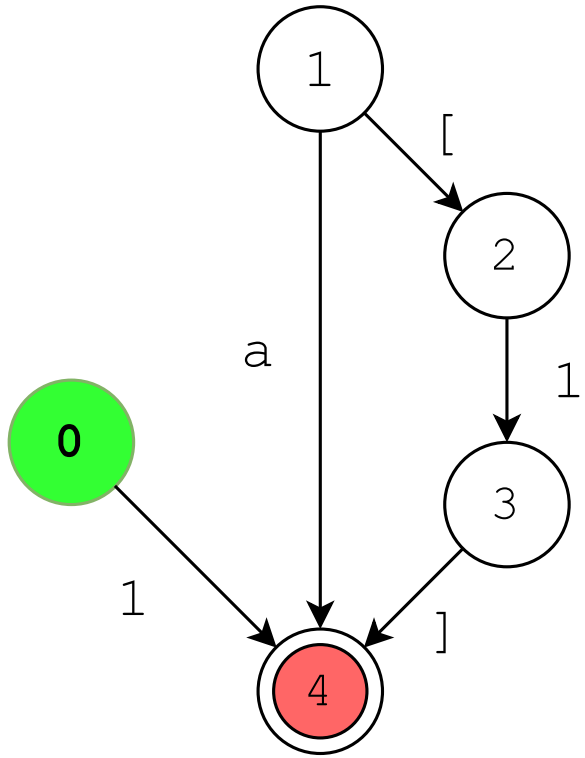
Следствие

Пусть G_p — произвольная KC-грамматика, G_d задает конечный язык $L(G_d) = \{\omega_1, \ldots, \omega_n\}$. В таком случае задача синтаксического анализа G_d разрешима и принадлежит PSPACE-complete классу.

Рекурсивные автоматы и КС-грамматики

- Контекстно-свободную грамматику можно представить в виде рекурсивного автомата
- На ребрах могут быть как терминальные символы, так и состояния-нетерминалы

$$
\begin{aligned}
S' &::= S \\
S &::= [\ S\ ] \\
S &::= \mathtt{a}
\end{aligned}
$$



Алгоритм

- **Вход**:
  - G_p — произвольная KC-грамматика
  - G_d — KC-грамматика без рекурсии, представленная в виде рекурсивного автомата R
- **Результат**: $\{(n_1, n_2)\}$, где n_1, n_2 — номера состояний автомата R, при этом существует ω такая, что $\omega \in L(G_p)$ и $\omega \in L(R')$, где R' получен из R заменой стартового и конечного состояний на n_1, n_2

Реализация

- Основан на алгоритме обобщенного синтаксического анализа GLL
  - позволяет использовать произвольную KC-грамматику
  - более сложная структура стека (GSS)
- Модификация GLL для синтаксического анализа конечных автоматов [Рагозина А., 2016 г.; Горохов А., 2017 г.]
  - производит анализ регулярного множества строк, представленного автоматом
- Основные управляющие функции изменены для поддержки рекурсивных автоматов
  - обработка нетерминальных переходов и финальных состояний
  - два взаимосвязанных GSS
- Алгоритм реализован в рамках исследовательского проекта YaccConstructor на F#

Эксперименты

- Последовательность случайных символов
- Содержит определенное количество шаблонов, удовлетворяющих грамматике

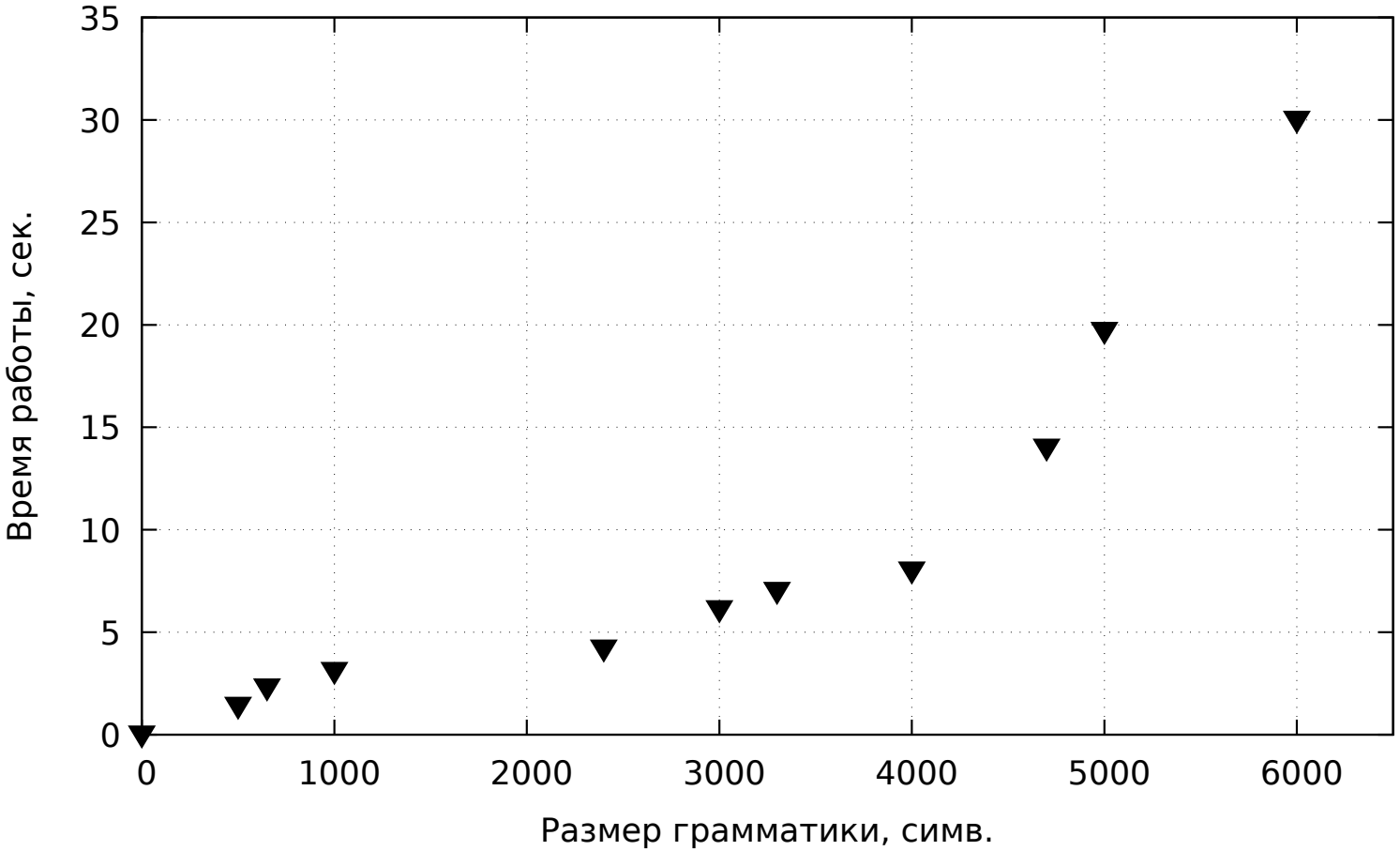
$$S ::= [\, S \,] \mid a$$

- Сжатие в KC-грамматику (алгоритм Sequitur)
- Для замеров оценивается размер входной грамматики

$$|G| = \sum_{p \in P} length(p)$$

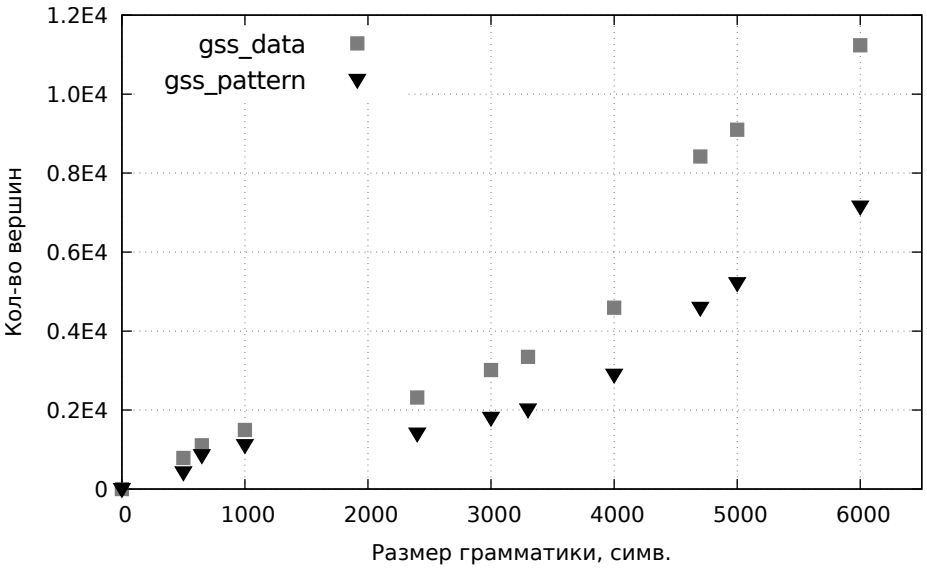
Эксперименты: время работы

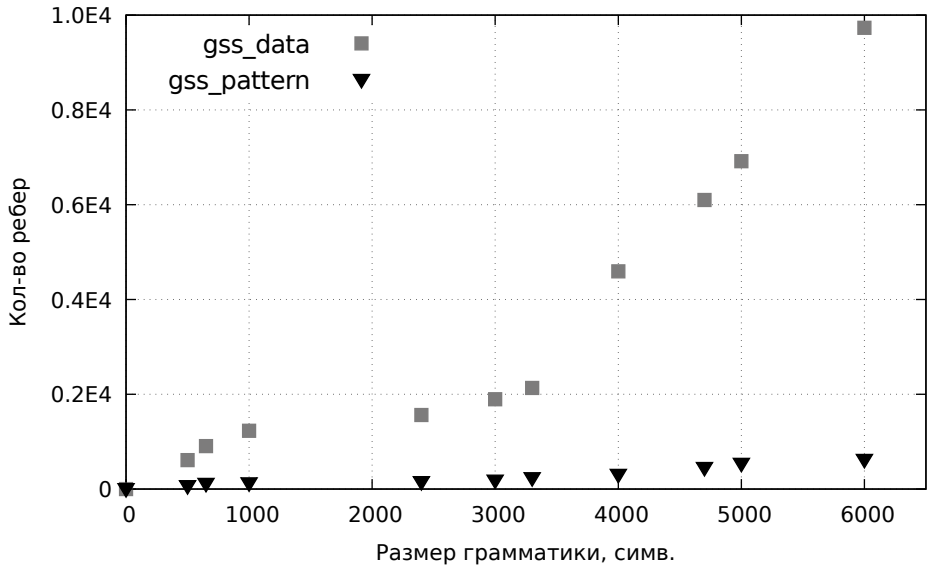
Грамматика шаблона: $S ::= [\, S \,] \mid a$



Эксперименты: память

Грамматика шаблона: $S ::= [\, S \,] \mid a$
Измеряется размер стеков (GSS) анализатора





Полиномиальный рост объема используемой памяти — PSPACE

Заключение

- Определены ограничения, при которых синтаксический анализ контекстно-свободного представления является разрешимой задачей
- Разработан алгоритм синтаксического анализа KC-представления, учитывающий поставленные ограничения
- Предложенный алгоритм реализован на языке программирования F# в рамках исследовательского проекта YaccConstructor
- Проведено экспериментальное исследование

- По материалам работы был выполнен доклад на конференции PLC'17, тезисы опубликованы в сборнике материалов конференции. Принята к публикации статья в журнале, входящем в список ВАК.