



Поддержка расширенных контекстно-свободных грамматик в алгоритме синтаксического анализа Generalised LL

Автор: Горохов Артем Владимирович, 471 гр.

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

Рецензент: программист СУИ НИУ ИТМО Авдюхин Д.А.

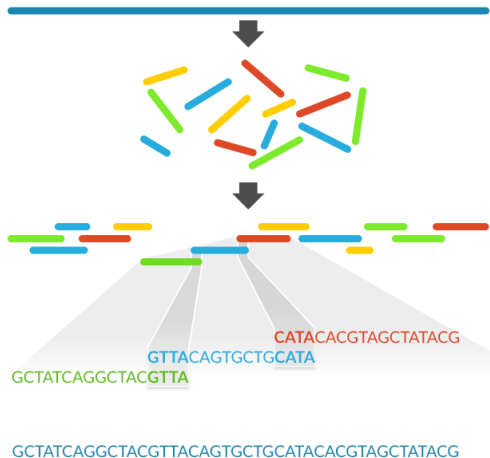
Санкт-Петербургский Государственный Университет

9 июня 2017

- Множество задач, связанных с обработкой и пониманием биологических данных
- Одна из задач — поиск организмов в метагеномных сборках
- Геном — длинная последовательность нуклеотидов
- На деле строка над алфавитом $\{A, C, G, T\}$

Получение данных

- Из биологического материала читаются короткие строчки
- Эти кусочки склеиваются в более длинные строки
- Множество строчек — сборка
- Данных очень много, поэтому строится конечный автомат, пути в котором содержат полученные строки



Метагеномная сборка

- Изучаем набор генов всех микроорганизмов в образце
- Нужно уметь определять содержащиеся в сборке организмы

Как ищем

- Такие последовательности как тРНК, рРНК и др. позволяют провести классификацию организма
- У этих последовательностей есть вторичная структура, которая может быть описана КС-грамматикой

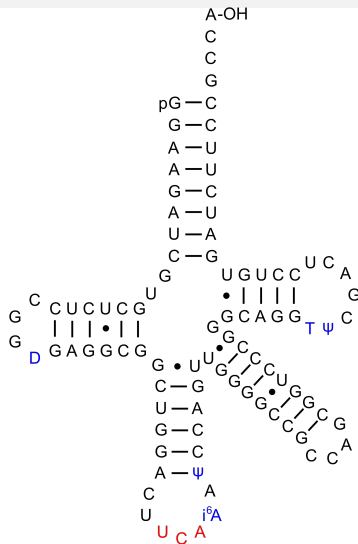


Рис.: Структура тРНК

- В магистерской прошлого года реализован алгоритм синтаксического анализа регулярных множеств
 - ▶ Основан на алгоритме Generalised LL
 - ▶ Умеет решать задачу поиска цепочек в конечном автомате, удовлетворяющих КС-грамматике

Расширенные контекстно-свободные грамматики

В правых частях продукций регулярные выражения

$$\begin{aligned} S &= a M^* \\ M &= a? (B K)^+ \\ &\quad | u B \\ B &= c \mid \varepsilon \end{aligned}$$

- Теоретические работы о синтаксическом анализе ECFG
 - ▶ L. Breveglieri, S. Crespi Reghizzi, A. Morzenti (2014). ELR parsing
 - ▶ K. Hemerik.(2009) ECFG and RRPg Parsing
- Описываются лишь LL(k), LR(k) подходы
- Работать с LL проще чем с LR
- Нет инструментов допускающих произвольные ECFG
- **Generalised LL**
 - ▶ Допускает произвольные CFG (включая неоднозначные)
 - ▶ Не может использовать ECFG без преобразований

Цель и задачи

Цель работы: разработать и реализовать модификацию алгоритма GLL, работающую с расширенными контекстно-свободными грамматиками, и проверить, как полученный алгоритм повлияет на производительность поиска структур, заданных с помощью контекстно-свободной грамматики, в метагеномных сборках. Для её достижения были поставлены следующие задачи:

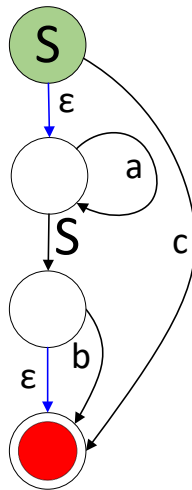
- Выбрать или разработать подходящее представление ECFG
- Спроектировать структуру данных для представления леса разбора по ECFG
- Разработать алгоритм на основе Generalised LL, строящий лес разбора по ECFG
- Разработать механизм анализа регулярных множеств в алгоритме
- Реализовать алгоритм в рамках проекта YaccConstructor
- Провести эксперименты и сравнение

Грамматика G_0

$$S = a^* S b? \mid c$$

\Rightarrow

РА для
грамматики G_0

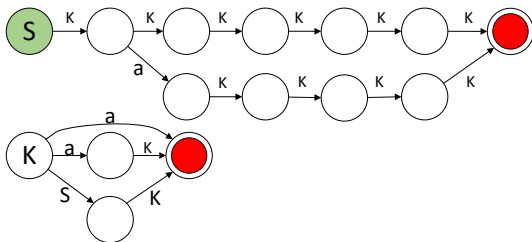


Минимизация рекурсивных автоматов

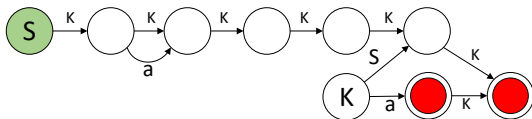
Грамматика G_1

$$S = K K K K K K \mid K a K K K K$$
$$K = S K \mid a K \mid a$$

Автомат для G_1



Минимизированный автомат для G_1

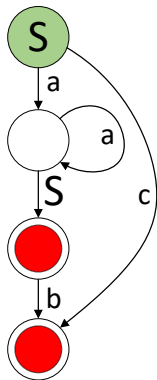


Деревья вывода для рекурсивных автоматов

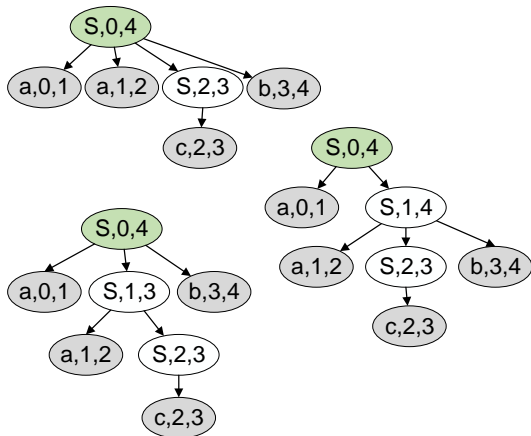
Вход:

aacb

Автомат:



Деревья вывода:



- Основан на Generalised LL
 - ▶ Определено сжатое представление леса разбора(SPPF) для рекурсивных автоматов
 - ▶ Поддержаны рекурсивные автоматы вместо грамматик
- Поддержан механизм анализа регулярных множеств
- Реализован в рамках проекта кафедры системного программирования YaccConstructor

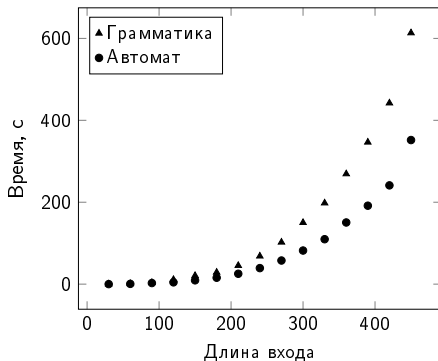
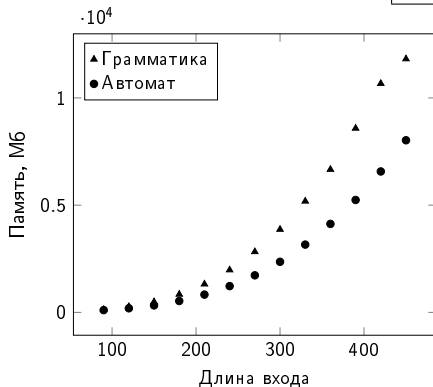
Эксперименты

Грамматика G_1

$$S = K(K K K K K \\ | a K K K K) \\ K = S K | a K | a$$

Результаты для входа a^{450}

	Время	Память, Мб
Грамматика	10мин.13с.	11818
RA	5мин.51с.	8026
Ratio	43%	33 %



Поиск в метагеномных сборках

- Были проведены эксперименты на метагеномной сборке
- 10 компонент с 400-100 состояний и переходов и 1118 компонент с менее чем 100 состояний и переходов

	Память	Время
Существующее решение	27 Гб	02.26 мин
Предложенный в данной работе	10 Гб	01.25 мин
Ratio	63 %	45 %

- В качестве подходящего представления ECFG предложены рекурсивные автоматы
- Определён SPPF для ECFG
- Разработан алгоритм на основе GLL, строящий SPPF по ECFG
- Разработан механизм анализа регулярных множеств в алгоритме
- Алгоритм реализован в рамках проекта YaccConstructor
- Эксперименты показали двухкратный прирост производительности по сравнению с существующим решением
- Выступление на международной конференции “Tools and Methods of Program Analysis”(Москва, 2017г.)