Реализация возможности сжатия строки в КС-грамматику в YaccConstructor

Автор: Зиновьева А.Г.
Руководитель: к.ф.-м.н. Григорьев С.В.

Введение

- Информация, например текст или музыка, может содержать повторяющиеся фрагменты, которые также могут содержать повторяющиеся фрагменты и так далее
- Можно описать повторяющийся фрагмент только один раз, а не для каждого случая отдельно
- Последовательности ДНК и РНК, содержащие всего 4 разновидности символов, часто имеют много общих подпоследовательностей
- В YaccConstructor нет возможности построить грамматику для строки

Задачи

Для реализации возможности сжатия текстовой строки в грамматику в YaccConstructor были поставлены следующие задачи:

- Реализовать алгоритм сжатия строки
- Реализовать возможность построения общей грамматики для нескольких строк
- Реализовать конечное представление грамматики в формате YARD.IL
- Оформить фронтенд для данного алгоритма к YaccConstructor
- Протестировать решение
- Проверить эффективность данного решения

Алгоритм Sequitur

- Автор - К. Невилл-Манин, 1997 год
- Принимает на вход последовательность дискретных символов: например, текстовую строку
- Посимвольно обрабатывает строку
- Ищет повторяющиеся пары символов и заменяет их на нетерминальные
- Результатом алгоритма является контекстно-свободная грамматика
- Работает за линейное время
- Может использоваться для сжатия данных

Алгоритм Sequitur

Свойства грамматики:

- Уникальность: не может быть двух правил с одинаковой правой частью
  - Невозможна такая ситуация: A -> ab и B -> ab

- Полезность: каждое правило используется более одного раза
  - S -> AA A -> Bc B -> ab : Правило B не является полезным, поэтому правило A должно быть преобразовано в A -> abc

Алгоритм Sequitur

“abcdabc”

1. S->a 2. S->ab 3. S->abc 4. S->abcd 5. S->abcda

Новых правил не образовалось, т.к. нет повторяющихся диграмов

6. S->abcdab : S->AcdA A->ab

Встретился повторяющийся диграм, добавляем новое правило

7. S->AcdAc A->ab : S->BdB B->Ac A->ab : S->BdB B->abc

Создали новое правило, но правило A стало бесполезным, поэтому избавляемся от него

YARD

- Язык спецификаций грамматик, являющийся частью YaccConstructor
- Поддерживает EBNF (Расширенная Форма Бэкуса-Наура)
- Позволяет описать результат работы алгоритма с помощью двух выражений
  - выбор (A|B)
  - конкатенация (A,B)

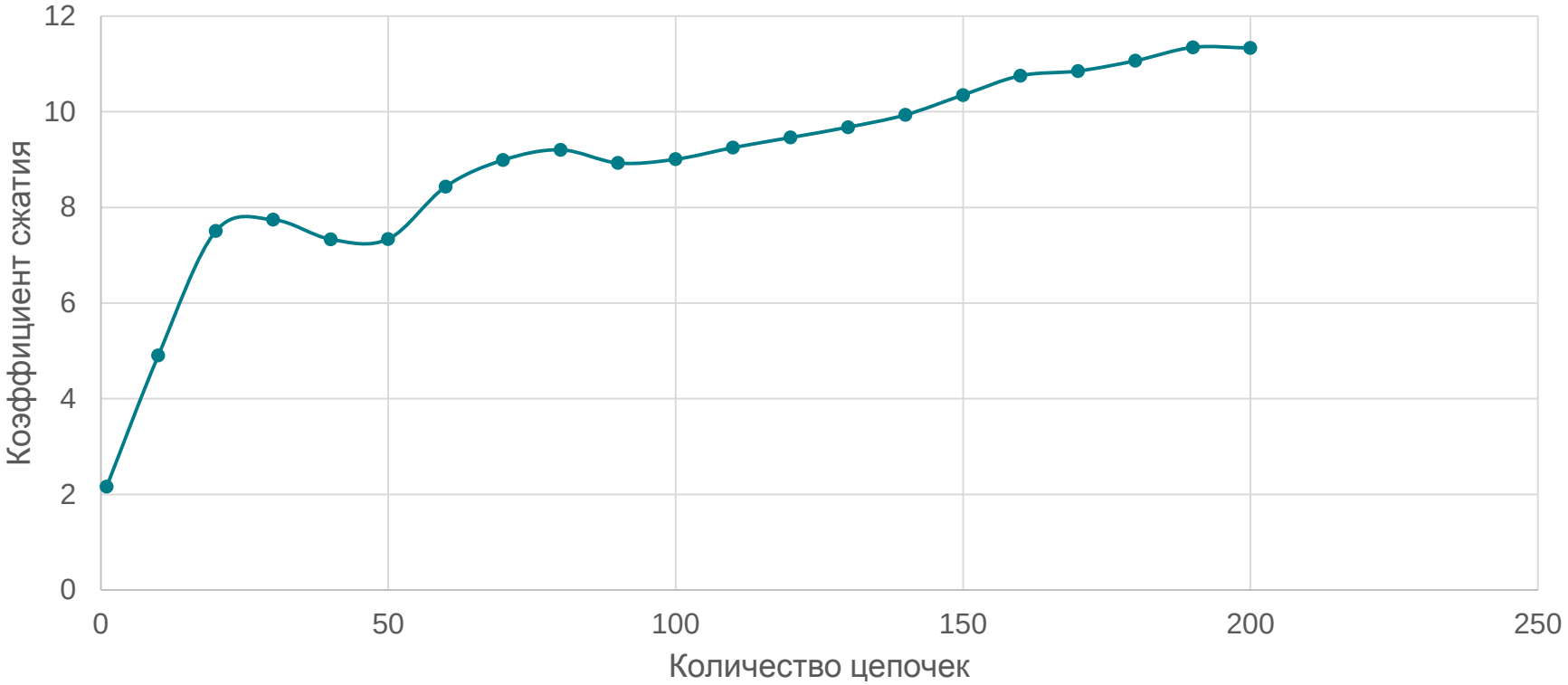
Реализация

- Выбор структуры данных для описания грамматики - двусвязный список
- Основная функция реализации - обработка стека диграм
- Модификация алгоритма для сжатия нескольких строк: обрабатываются только диграмы, не содержащие специальный символ
- Представление в YARD.IL - построение грамматики с помощью конструкторов PAlt, PSeq, PRef и PTok
- Оформление фронтенда к YaccConstructor - возможность сжать строку, строку со специальными символами и массив строк

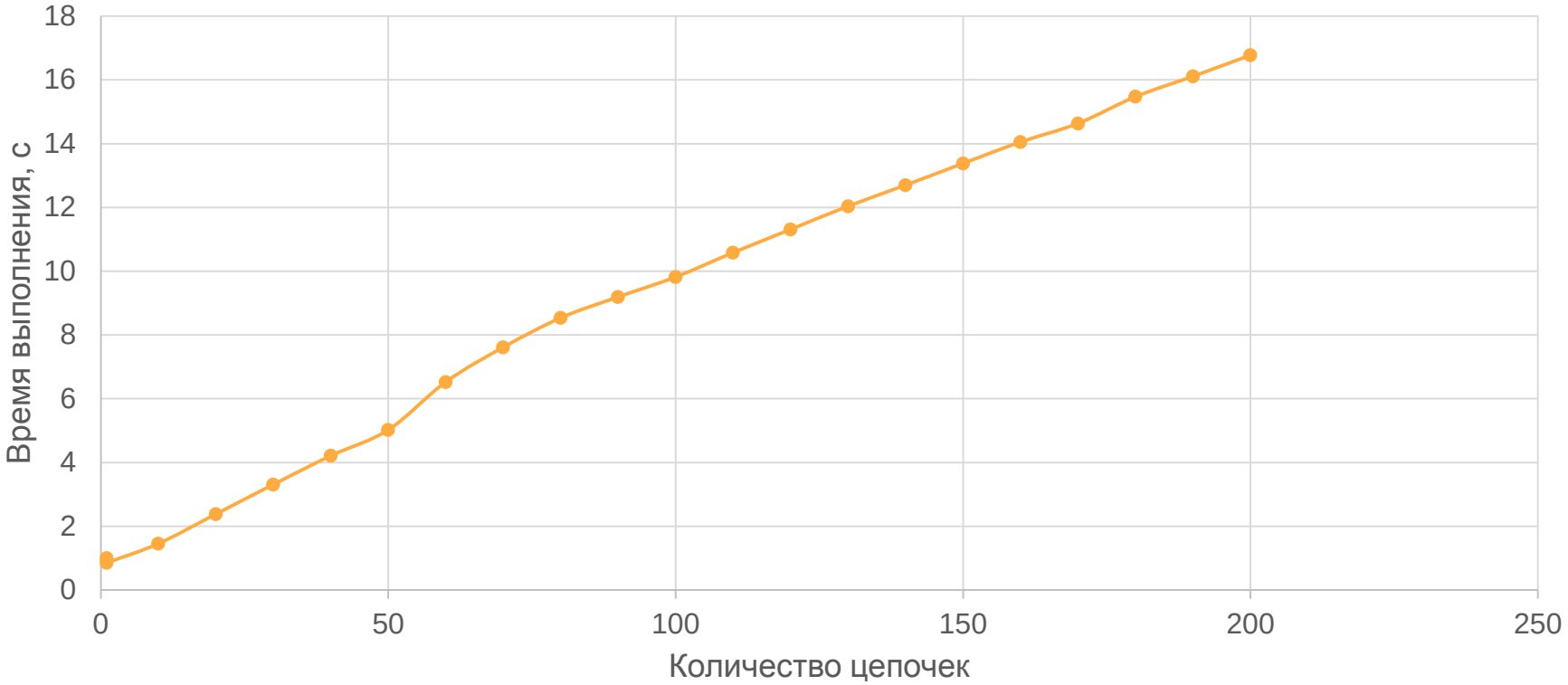
Эксперимент

- Для проверки эффективности решения из SILVA database была взята последовательность 16s РНК бактерий
- Каждая цепочка состоит примерно из 1500 нуклеотидов, то есть последовательность символов из алфавита {A; C; G; T}
- Данные последовательности были склеены через специальный символом & и переданы на вход алгоритму.

Эффективность



Производительность



Результаты

- Реализован алгоритм Sequitur
- Реализована возможность построения общей грамматики для нескольких срок
- Оформлен фронтенд к YaccConstructor для построения грамматики из строки в формате YARD.IL
- Проведено тестирование с помощью системы NUnit
- Проверена эффективность и производительность данного алгоритма на последовательностях РНК