



Зачем биологам синтаксический анализ

Автор: Артём Горохов

Санкт-Петербургский государственный университет
Лаборатория языковых инструментов JetBrains

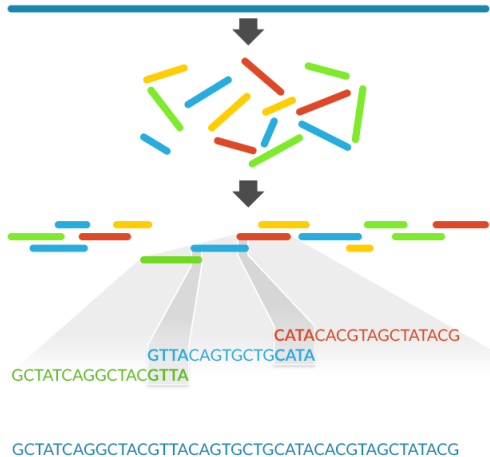
15 октября 2016г.

- 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 $\{A, C, G, U\}$

Получение данных

- Из биологического материала читаются короткие строчки
- Эти кусочки склеиваются в более длинные строки
- Множество строчек — сборка
- Данных очень много, поэтому строится граф, содержащий множество полученных строк



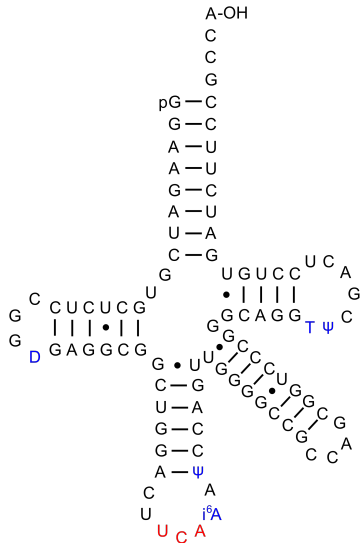
Метагеномная сборка

- Данные из окружающей среды
- Изучаем набор генов всех микроорганизмов в образце

Что ищем

- Хочется понять что у нас в сборке
- Такие последовательности как тРНК, рРНК позволяют провести классификацию организма
- У этих последовательностей есть вторичная структура, которая может быть описана КС-грамматикой

GGAAGAUCG...GCA... =>



Грамматика для кусочка тРНК

START = *STEM*

STEM = *a STEM u*

| *u STEM a*

| *c STEM g*

| *g STEM c*

| *g STEM u*

| *u STEM g*

| *ANY*[4..7]*

ANY = *a | u | g | c*

G • U

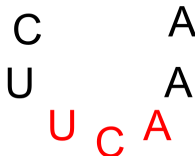
C — G

U — A

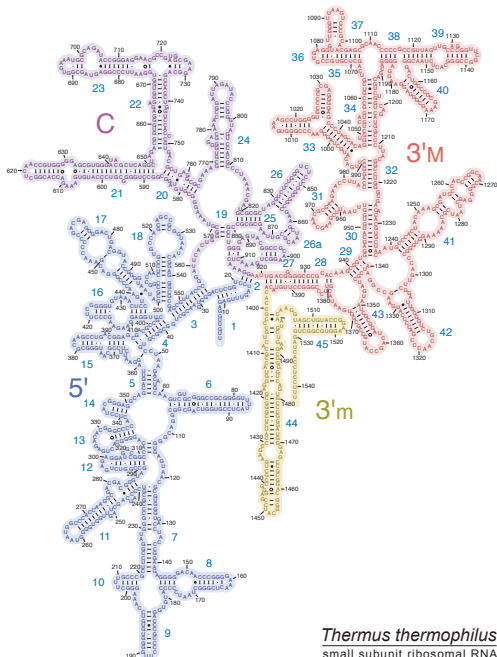
G — C

G — C

A — U



Вторичная структура 16s рРНК



Thermus thermophilus
small subunit ribosomal RNA

- Задача поиска линейных цепочек, удовлетворяющих КС-грамматике, в графе

- В лаборатории созданы алгоритмы
- Реализован инструмент, основанный на алгоритме GLL
- Умеет решать задачу поиска линейных цепочек в графе, удовлетворяющих КС-грамматике

Цель работы — научиться классифицировать организмы в метагеномной сборке

Задачи:

- Адаптировать существующий алгоритм под специфику задачи
- Провести экспериментальные исследования работы алгоритма

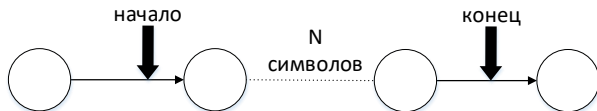
- Разбор осуществляется при помощи дескрипторов
- Дескриптор — четвёрка (слот, позиция во входе, дерево, вершина стека)
- На каждом шаге достаём дескриптор из очереди и разобрав очередной символ создаём новые дескрипторы

Метагеномные сборки довольно большие, поэтому их необходимо предварительно обрабатывать

- Infernal позволяет распознавать структуры в линейном входе
- Рёбра, длинее искомым структур можно делить на части и проверять с помощью Infernal
- После фильтрации рёбер граф распадается на компоненты связности, на которых алгоритм можно запускать анализатор независимо

Отказ от построения дерева

- Синтаксический анализатор возвращает лишь границы и длину найденных цепочек
- Восстановление цепочки идёт путём извлечения подграфа, состоящего из путей заданной длины
- Ложные фильтруются с помощью Infernal



- Грамматика для 16s рРНК сильно неоднозначная и довольно большая
- Из-за этого количество слотов в грамматике очень много
- В процессе разбора создаётся огромное количество дескрипторов

Преобразование грамматики к автомату

Грамматика

$START = STEM$

$STEM = a STEM u$

$| u STEM a$

$| c STEM g$

$| g STEM c$

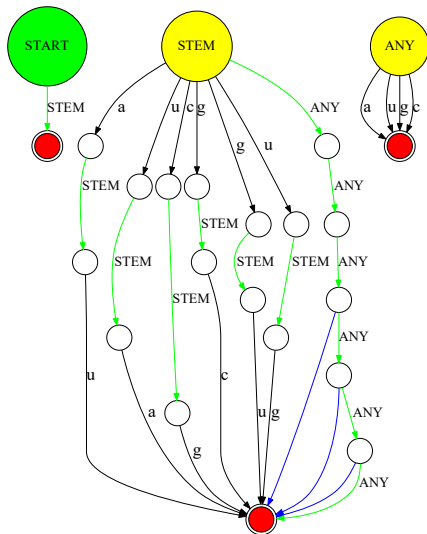
$| g STEM u$

$| u STEM g$

$| ANY^*[3..6]$

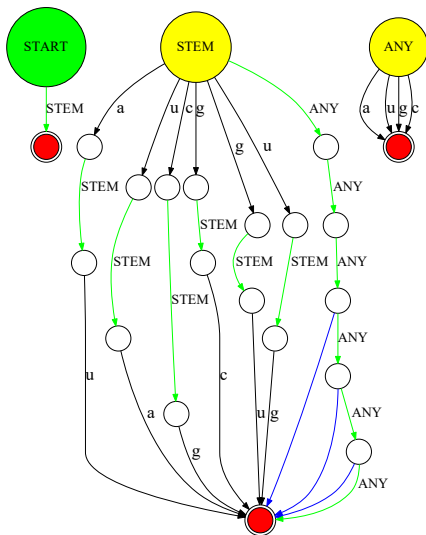
$ANY = a | u | g | c$

Автомат

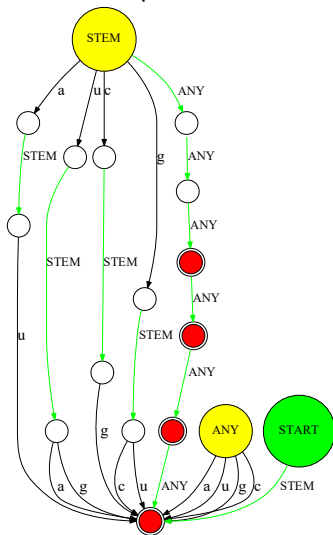


Минимизация автомата

Изначальный автомат



Минимизированный автомат



Результаты работы на сборке, состоящей из 59000 вершин и 87000 рёбер и грамматике кусочка 16s pPНК, длиной около 300 символов

	начальная грамматика	мин. автомат
Время работы	10 ч.	3 ч. 40 мин.
Кол-во слотов /состояний	41	17

Эксперименты проводились на машине с 32 ГБ RAM и CPU core i7-4790

- Разработан механизм подготовки сборок к синтаксическому анализу
- GLL адаптирован под распознавание по грамматике в форме EBNF
- Проведены эксперименты на части грамматики 16s рРНК

- Детальный анализ качества результата
- Исследовать возможность сильнее фильтровать граф, например применяя Infernal
- Поиск полноразмерных 16s рРНК