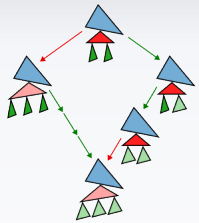


Лабораторная работа 2: Обучение МАТом

Обучение регулярных языков по запросам к «минимально адекватному учителю».
Вариант определяется последней цифрой в зачётке.

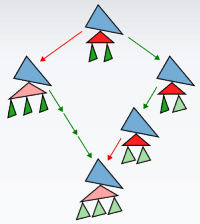
(Номера 3,4) Планарный лабиринт

(Номера 0,1,9) Ортогональный лабиринт

(Номера 6,8) Brainfuck-Рефал

(Номера 2,5,7) Brainfuck-Лисп

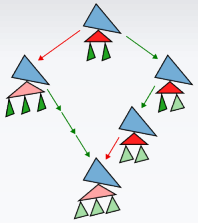
В каждом варианте от 2 до 3 человек — МАТ, остальные — угадыватели.



Техническое задание

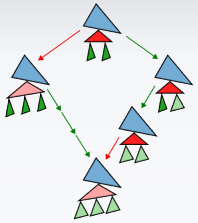
- Со стороны MAT — сгенерировать случайный автомат, соответствующий задаче, и реагировать на запросы об эквивалентности и включении. Реакция на запрос о включении должна быть с минимальной задержкой (парсинг по детерминированному автомату). Реакция на запрос об эквивалентности допускается не оптимальная по скорости. Кроме того, MAT не обязан возвращать минимальный контрпример (может и произвольный). Также MAT должен визуализировать итоговый автомат.

- Со стороны угадывателя — построить полное описание автомата, используя запросы двух видов. Следует учесть, что запрос об эквивалентности может обрабатываться достаточно медленно, особенно для автоматов большого размера.



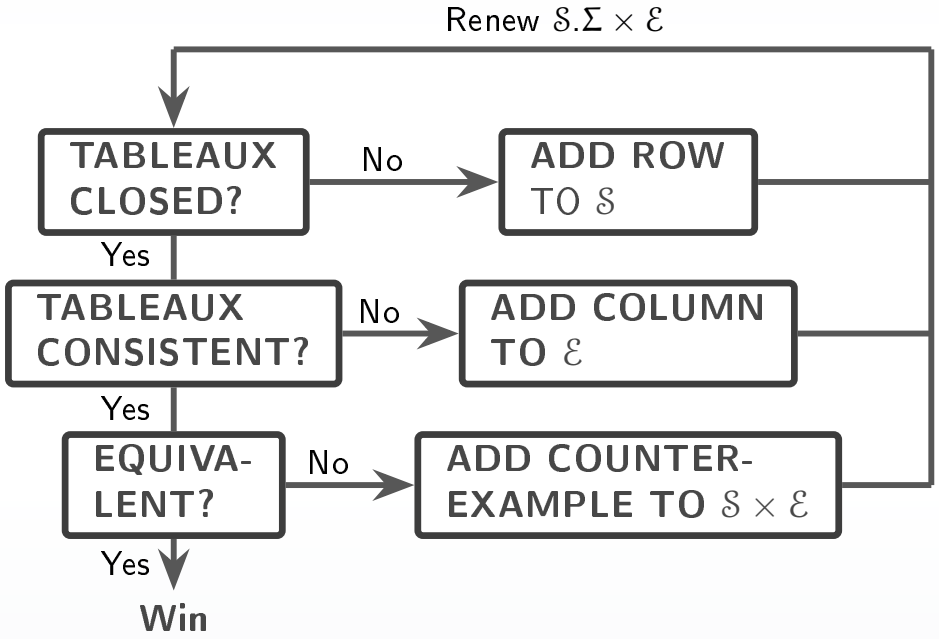
Необходимые инструменты МАТ

- Генератор случайных входных данных (согласно ограничениям ТЗ);
- Минимизация и каноническая нумерация автомата (допускается и другое однозначное внутреннее представление автомата, например, в форме матрицы для ортогонального лабиринта).
- Для построения контрпримера —- пересечение автомата и его дополнения и генерация случайной строки в языке пересечения, либо альтернативный алгоритм.
- Визуализация — либо лабиринт (варианты с лабиринтами), либо автоматы для лексем, и общий автомат для лексера.

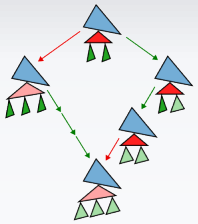


Инструменты угадывателя

Рекомендуется использовать фреймворк алгоритма L^*, допускающего вывод произвольного регулярного языка в форме таблицы классов эквивалентности Майхилла–Нероде.

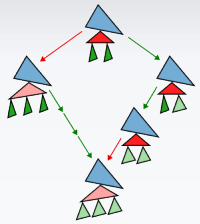


$\mathcal{S}$ — классы эквивалентности (определяющие состояния ДКА);
$\mathcal{E}$ — различающие суффиксы;
$\mathcal{S}.\Sigma$ — расширенные префиксы: префиксы, в конец которых приписаны буквы из Σ.



Алгоритм L^*

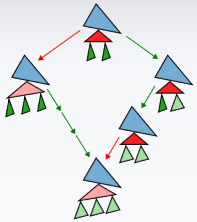
- Условие полноты — отсутствие в $S.\Sigma \times E$ строк, которые отличаются от строк в $S \times E$. Иначе дополняем S новой строкой.
- Условие непротиворечивости — отсутствие в $S.\Sigma \times E$ таких позиций i, j, k, что $u_i v_k \neq u_j v_k$, притом что $u_i = u_i'\gamma$, $u_j = u_j'\gamma$, и строки в таблице $S \times E$ для u_i' и u_j' совпадают. Иначе дополняем E столбцом γv_k.
- При получении контрпримера дополняем либо S им и всеми его префиксами, либо E им и всеми его суффиксами.
- ДКА по таблице строится следующим образом.
  - Состояния ДКА — кратчайшие слова из S, порождающие разные строки в $S \times E$.
  - Начальное состояние соответствует префиксу ε.
  - Конечные состояния — те, которые содержат 1 в столбце, помеченном ε.
  - Если $u\gamma \equiv u'$ (т.е. $u\gamma$ и u' соответствуют одинаковым строчкам в таблице), то $\langle u, \gamma \rangle \to u'$ добавляется в ДКА как переход.



Вариации L*

- Блочный L*: если вы можете разбить автомат над подавтоматы однозначно, можно попробовать выводить их по отдельности, а собирать в последнюю очередь.
- Опережающий L*: если запросы эквивалентности очень дорогие, и есть высокий шанс, что эквивалентность ещё не достигнута, то можно расширять таблицу не на один шаг, а больше.
- Осторожный L*: пробовать добавлять больше различающих суффиксов.

Кроме того, можно использовать вспомогательные запросы, не следующие структуре общего алгоритма (например, если вы пытаетесь угадать алфавит `[eol]`-лексем из BF-вариантов, это может быть выгодно сделать в качестве предварительного шага).



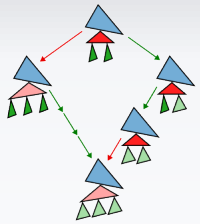
Интерфейс MAT и угадывателя

- Запросы на включение передаются просто как строка. Ответ на вопрос о включении — 0 или 1.
- Запросы на эквивалентность передаются как таблица классов эквивалентности:

```
         | epsilon
-----------------
epsilon |  val1
class1  |  val2
.....
classk  |  valk
```

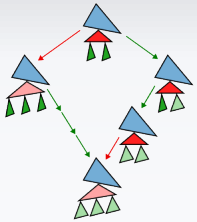
  Здесь `epsilon` — указание на пустую строку, имена строк — классы эквивалентности, `valk` — 0 или 1, показывающие, является ли класс входящим в язык (т.е. соответствующим финальному состоянию). Ответ на вопрос об эквивалентности: TRUE или строка, являющаяся контрпримером.

Для единообразия примем, что таблицы и строки принимаются MAT-ом во входной поток, и ответ также передаётся в стандартный поток.



Планарный лабиринт

Угадыватель заранее знает оценку сверху на число ветвлений в лабиринте и число выходов из него (читая из файла parameters.txt два соответствующих числа). По этому же файлу MAT генерирует граф, используя данные числа как оценку сверху.
Входной автомат — произвольный планарный граф с бинарным ветвлением. Все тупиковые ветки — выходы из лабиринта. Путь в лабиринте представляет собой последовательность инструкций в алфавите {L, R}: указывающих на каждой развилке, идти направо или налево.
Путь принадлежит языку, если строго выводит к выходу (т.е. следуя инструкциям и начиная из стартового состояния, вы попадаете в финальное состояние). Если путь остаётся в лабиринте или содержит дополнительные инструкции после попадания в финальное состояние, считается, что он языку не принадлежит.



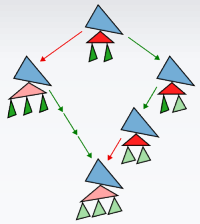
Ортогональный лабиринт

Угадыватель заранее знает размеры прямоугольника, ограничивающего размер лабиринта (читая из файла parameters.txt два числа: его ширину (размеры с востока на запад) и длину (размеры с севера на юг)). По этому же файлу MAT генерирует сам автоматный лабиринт.

Входной автомат — ортогональный лабиринт, помещающийся внутрь прямоугольника (но не обязательно заполняющий его целиком), и имеющий один или больше выходов за пределы прямоугольника. Путь в лабиринте представляет собой последовательность инструкций в алфавите $\{\mathtt{E}, \mathtt{W}, \mathtt{N}, \mathtt{S}\}$: указывающих в каждой клетке, идти на восток, запад, север или юг. Если с соответствующей стороны стена, то шаг не осуществляется.

Стартовое состояние угадывателя: самая северная и западная клетка внутри лабиринта (при этом выхода рядом с ним может и не быть). Гарантируется, что в лабиринте все клетки достижимы из этой (может быть, посредством выхода из лабиринта и повторного захода в него).

Путь принадлежит языку, если выводит за пределы прямоугольника (т.е. следуя инструкциям и начиная из стартового состояния, вы попадаете за пределы прямоугольника, и возможно, делаете шаги вне его). Если путь остаётся в лабиринте или сначала выводит из прямоугольника, а потом возвращает в него, то он языку не принадлежит.



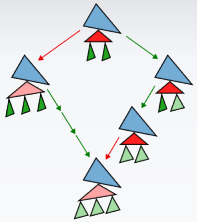
Грамматика Brainfuck Рефал

Автомат, распознающий лексически правильные конструкции обфусцированного рефала, строится по следующей грамматике:

```
[program] ::= [eol]*([definition][eol]+)+
[definition] ::= [const] [lbr-1] ([eol]*[sentence])* [eol]*[rbr-1]
[sentence] ::= [pattern][equal][expression][sep]
[pattern] ::= [lbr-3][pattern][rbr-3]|[pattern][blank][pattern]
              | [var] | [const] |
[expression] ::= [var] | [const] | [expression][blank][expression]
                 [lbr-3] [expression] [rbr-3] |
                 [lbr-2] [const] [blank] [expression] [rbr-2]
```

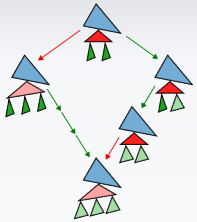
Если для лексемы нет определения в грамматике, значит, она определяется языком некоторого автомата. Алфавит BF-Рефала: $\{a, b, c, 0, 1, 2\}$.

Автоматы могут быть какими угодно, в том числе допускающими недетерминированный и неоднозначный разбор по лексемам. Однако обязаны выполняться базовые правила (см. следующий слайд).



Ограничения Brainfuck Рефал

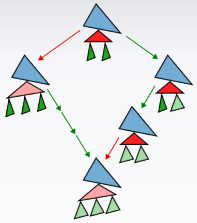
- автоматы для `[eol]` и `[blank]` имеют алфавит, отличный от всех остальных лексем (и друг от друга);
- первые и последние буквы слов, соответствующих автоматам для `[equal]` и `[sep]`, находятся в другом алфавите, чем все прочие автоматы, и языки у этих лексем конечные (но не обязательно синглетоны);
- автоматы для разных типов скобок, левых и правых, все имеют попарно непересекающиеся языки, не пересекающиеся с языками переменных и констант, кроме того, конкатенация любых двух таких автоматов друг с другом будет иметь язык, не пересекающийся ни с каким языком автомата для одной скобки (языки, содержащие единственные различные непустые строки, являются тривиальным случаем, подходящим под это условие, но есть и другие случаи);
- языки переменных и констант не пересекаются друг с другом, оба бесконечные.



Угадывание Brainfuck Рефал

Угадыватель заранее знает максимальную вложенность скобок и максимальный размер автомата для лексемы (читаемые из файла parameters.txt). По этим же параметрам МАТ генерирует автоматы для языка. Вложенность скобок считается отдельно по каждому из трёх типов (т.е. если указана 1, то может быть выражение со всеми тремя типами скобок, вложенными друг в друга). Сгенерированные автоматы могут быть меньше указанного размера (по числу достижимых состояний, не являющихся ловушками), но не больше.

Слово принадлежит языку тогда и только тогда, когда оно распознаётся соответствующим автоматом лексического анализа, с учётом условия непревышения вложенности скобок.

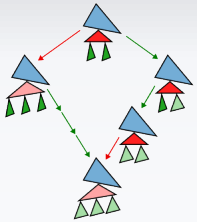


Brainfuck Лисп

Автомат, распознающий лексически правильные конструкции обфусцированного лиспа, строится по следующей грамматике:

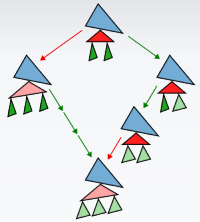
```
[program] ::= [eol]*([expression][eol]*)+
[expression] ::= [atom]| [eol][expression] | [expression][eol]
               | [lbr] [expression] [dot] [expression] [rbr]
               | [list]
[list] ::= [lbr] [expression]+ [rbr]
```

Если для лексемы нет определения в грамматике, значит, она определяется языком некоторого автомата. Алфавит BF-Lisp: $\{0 - 9\}$. Автоматы могут быть какими угодно, в том числе допускающими недетерминированный и неоднозначный разбор по лексемам. Однако обязаны выполняться базовые правила (см. следующий слайд).



Ограничения Brainfuck Lisp

- автомат для `[eol]` имеет алфавит, отличный от всех остальных лексем;
- автоматы для левых и правых скобок имеют попарно непересекающиеся языки, не пересекающиеся с языком атомов, кроме того, конкатенация любых двух таких автоматов друг с другом будет иметь язык, не пересекающийся ни с языком автомата для единственной скобки (языки, содержащие единственные различные непустые строки, являются тривиальным случаем, подходящим под это условие, но есть и другие случаи);
- язык для `[dot]` не пересекается с языком для атомов.
- язык лексемы атомов бесконечный.



Угадывание Brainfuck Lisp

Угадыватель заранее знает максимальную вложенность скобок и максимальный размер автомата для лексемы (читаемые из файла parameters.txt). По этим же параметрам MAT генерирует автоматы для языка. Сгенерированные автоматы могут быть меньше указанного размера (по числу достижимых состояний, не являющихся ловушками), но не больше.

Слово принадлежит языку тогда и только тогда, когда оно распознаётся соответствующим автоматом лексического анализа, с учётом условия непревышения вложенности скобок.