Курс с++

Царьков Олег

Описание тикета OTSN-17

Конечные мультиавтоматы. Рассмотрим некий граф из вершин и ребер. Среди множества вершин выделим одну, которую назовем *стартовой*, и выделим несколько, которые назовем *терминальными*. Такую констукцию назовем *конечным мультиавтоматом*. Заметим, что стартовая вершина тоже может быть терминальной – ничего плохого в этом нет. Какие-то вершины могут быть не стартовой, не терминальной.

Состоянием конечного мультиавтомата назовем какое-то произвольное множество его вершин.

Ребра мультиавтомата могут быть одними из двух типов:

- дельта-ребро или пустое ребро;
- именованное ребро, на котором написана маленькая латинская буква от “a” до “z”.

Правило перехода из одного состояния в другое. Пусть мы находимся в неком состоянии, то есть имеется какое-то множество вершин, в которых мы находимся (нас много). Тогда дельта-переход – это замена текущего множества вершин на множество вершин, достижимых из наших по дельта-ребрам. Иными словами, мы идем всюду куда можем ходить только по дельта-ребрам и везде, куда можем дойти, там начинаем находиться. При дельта переходе множество вершин, в которых мы находимся, расширяется (потому что из исходных вершин можно попасть в них самих по дельта-ребрам, не совершая никаких переходов вообще).

Пусть зафиксирована некая буква. Именованный переход — это переход по ребру с буквой, которая зафиксирована. Причем, в этот раз в отличие от дельта-перехода, мы должны искать те вершины, до которых путь по именованным ребрам имеет длину ровно 1. То есть, мы уходим от старых вершин к новым.

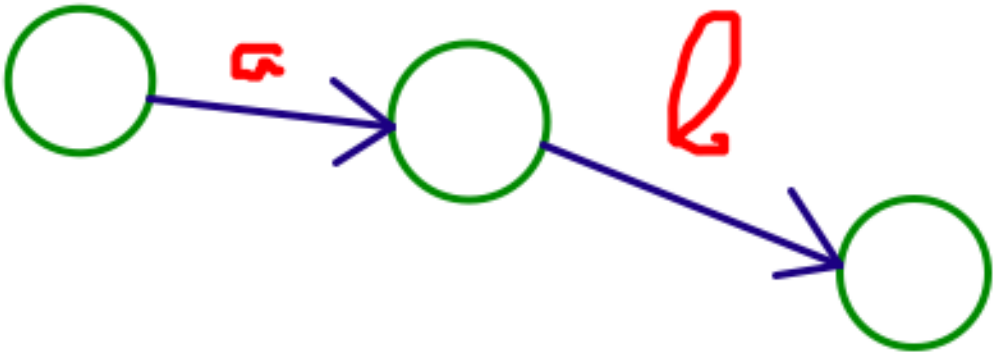
Почему эти два перехода такие разные? Потому что дельта-переход – это как бы переход по пустому ребру, на котором нет буквы, можно написать сколько угодно пустых строк, в том числе и 0 пустых строк – ничего не изменится; а вот если мы делаем переход по букве, то нужно именно использовать одно ребро, что будет соответствовать тому, что мы прочитали эту букву. Ее уже нельзя использовать 0 раз или больше 1 раза, как это было в дельта-переходе.

Как автомат задает язык. Мы будем говорить фразу *автомат задает слово* s, если начиная от стартовой вершины, можно совершать именнованные переходы с символами $s[0], s[1], \ldots, s[n-1]$, между которыми можно совершать любое количество любых дельта-переходов так, чтобы закончить свой путь в терминальной вершине.

Набор слов, которые задает автомат – это язык автомата.

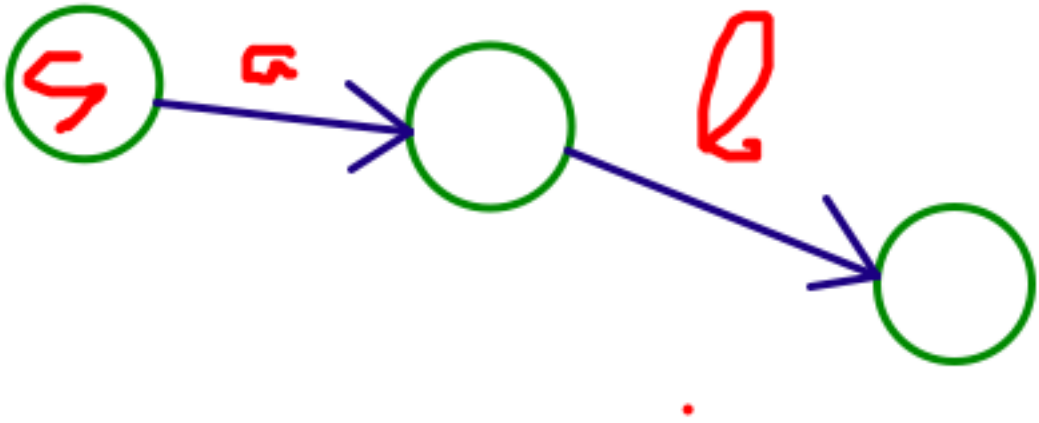
Примеры с автоматами. Будем обозначать стартовую вершину S, терминальные T.

Первый пример:



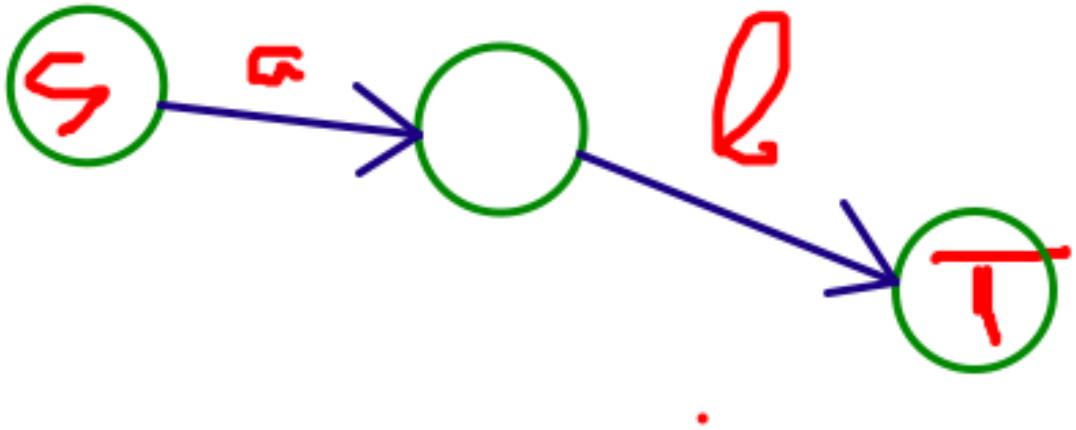
Какие слова задает этот автомат? Никаких, это не автомат, тут не ровно одна стартовая вершина, тут их нет.

Второй пример:

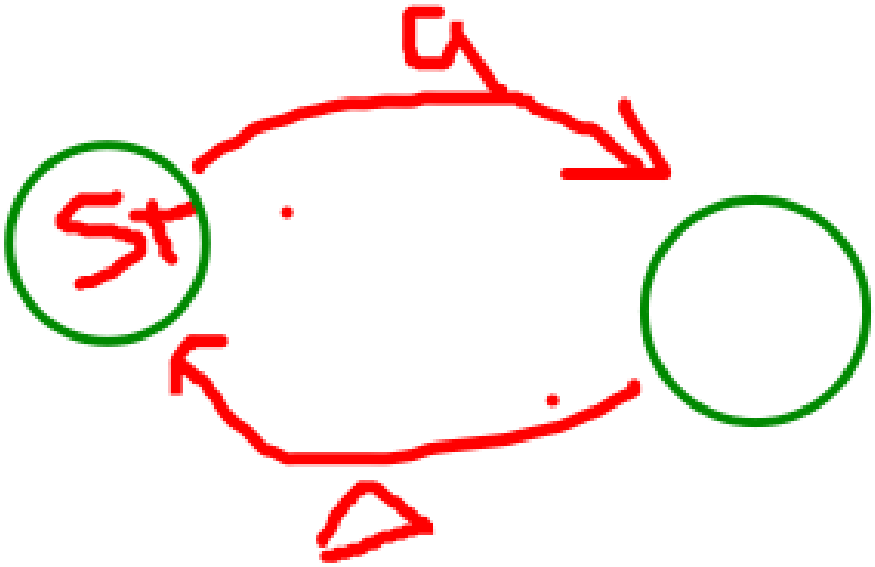


Какие слова задает этот автомат? Никаких, стартовая вершина есть, зато нет никаких терминальных и мы никогда не сможем до них добраться никаким способом.

Третий пример:



Данный автомат задает только одно слово ab.
Четвертый пример:



Одна вершина является и стартовой, и терминальной. Она ведет в другую ребром a, другая возвращается дельта-ребром. В итоге подходят все слова состоящие из буквы a, повторенной 1 или более раз.

Регулярное выражение. Регулярное выражение — это некая строка, которой должен *соответствовать* целый набор строк.

Регулярные выражения определяются просто и индуктивно:

- буква — это регулярное выражение. Единственная строка, которая ему соответствует – это строка из этой самой буквы.
- если A и B – два регулярных выражения, то можно записать их друг рядом с другом, получив выражение AB. Ему соответсвует те строки ab, которые

можно разбить на две части a и b так, что a соответсвует A, b соответствует B.

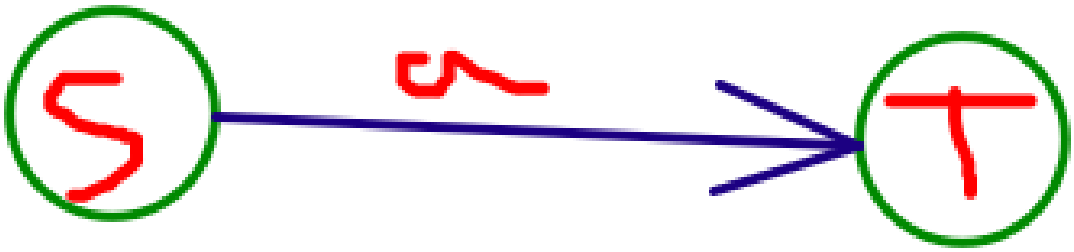
- если A и B – два регулярных выражения, то можно записать выражение $A|B$, которому соотвествуют строки, которые соответствуют либо регулярному выражению A, либо регулярному выражению B, либо обоим сразу;
- если A — регулярное выражение, то $A+$ – тоже регулярное выражение, которому соотвествуют такие строки $a_1 \dots a_n$, которые можно разбить на строки $a_1, \dots, a_n$, каждая из которых соотвествует выражению A.
- если A — регулярное выражение, то $A* = (A + |)$ – ему соотвествует либо то, что соотвествует $A+$, либо пустая строка.

Например, рассмотрим выражение $(a|b)*$. Выражению a соотвествует строка a. Выражению b строка b. Значит, выражению $a|b$ соответсвуют строки a и b. Дальше идет звездочка, то есть всему регулярному выражению соотвествует пустая строка и строки из букв a и b.

Соотвествие регулярных выражений и мультиавтомата. Для каждого регулярного выражения можно построить мультиавтомат, который задает ровно те строки, которые соотвествуют этому регулярному выражению.

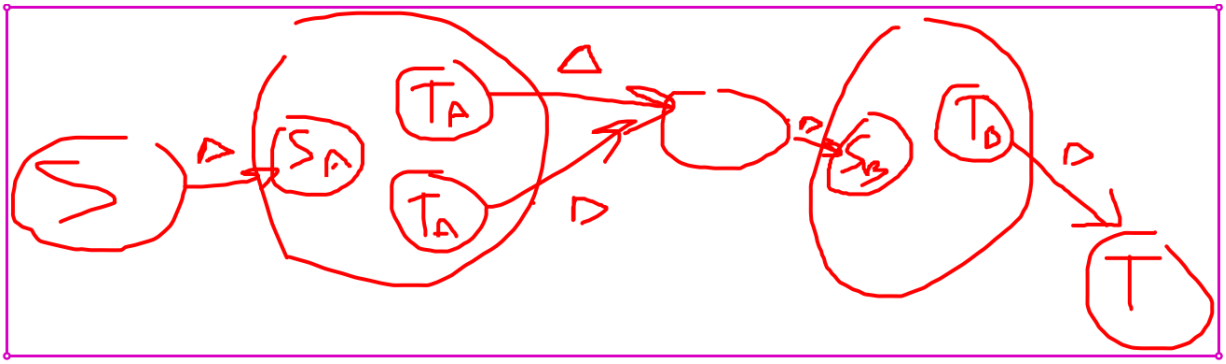
Это построение вполне явное. Вначале для самых маленьких однобуквенных выражений научимся строить мультиавтомат.

Пусть есть регулярное выражение a. Ему соответсвует слово a. Нужно построить такой автомат, который принимает только слово a. Вот он:



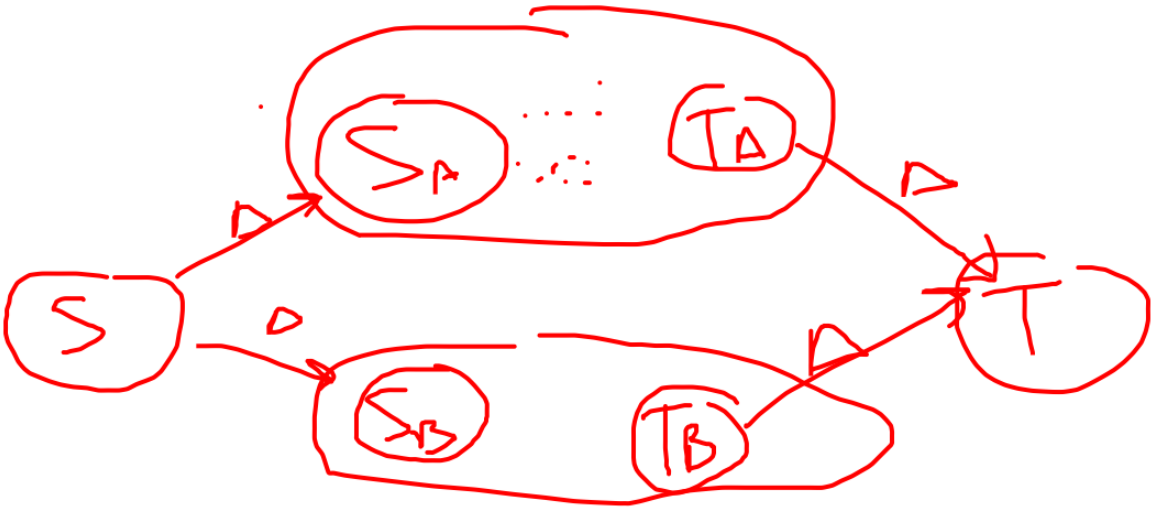
Далее, по индукции, нужно научиться строить граф, зная, что для более маленьких регулярных выражений он построен. Но это очень просто:

Для AB:



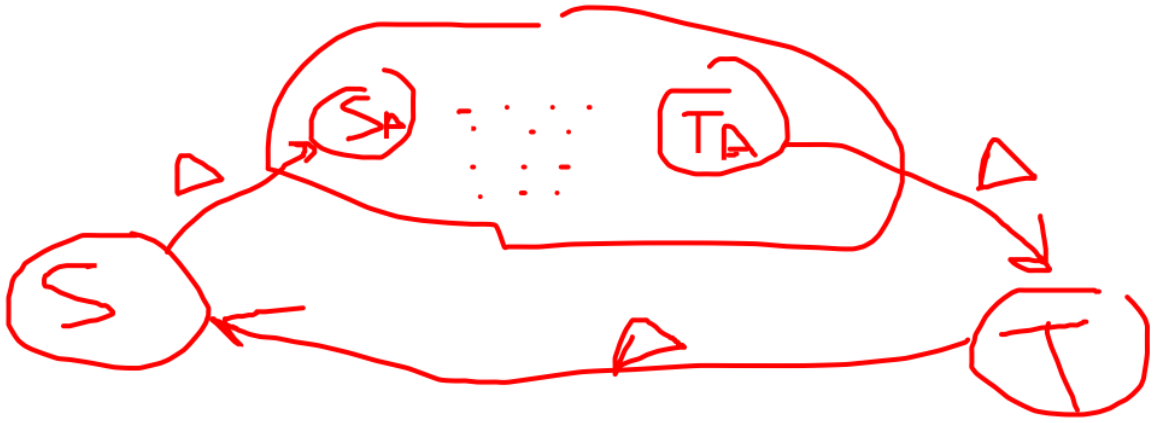
На картинке схематично изображено как из автомата A с стартовой вершиной S_A и с терминальными вершинами T_A и из автомата B с стартновой вершиной S_B и терминальными вершинами T_B построить один большой автомат со стартовой вершиной S и терминальной вершиной T, который будет принимать ровно те строки, которые вначале пройдут через автомат A, потом через автомат B, то есть ровно те строки, которые состоят из двух частей. Первая часть соответствует регулярному выражению A, для которого мы уже построили автомат A по предположению индукции. Вторая часть соответствует регулярному выражению B, для которого автомат B тоже уже построен.

Для $A|B$:



Очевидно, что этот автомат примет те строки, которые пройдут либо через A, либо через B.

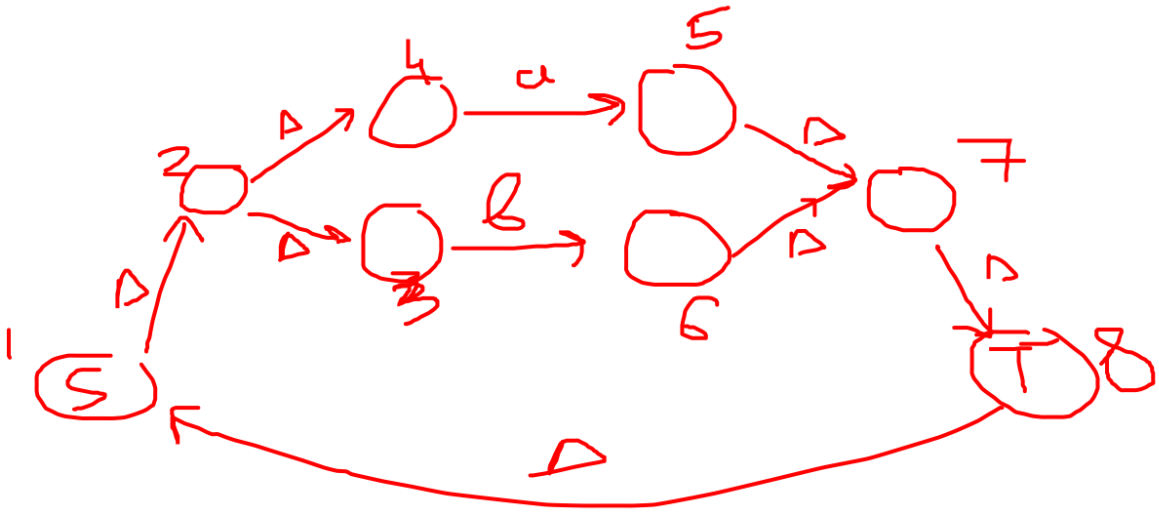
Для $A+$:



Понятно, что этот автомат будет принимать те строки, которые сколько-то раз прокрутятся через автомат A.

Основная задача. Нужно уметь по регулярному выражению и по строке понимать, соответствует ли строка регулярному выражению. Как это делать: попытаться построить граф – конечный мультиавтомат, который принимает те же строки, что и регулярное выражение. Далее нужно прогнать через него данную строку и понять, подходит она или нет.

Пример: регулярное выражение $(a|b)+$ и строка ab. Если последовать построению выше, получим автомат



Посмотрим, подходит ли в него строка ab. Да, потому что есть путь

$$1 \to 2 \to 4 \to 5 \to 7 \to 8 \to 1 \to 2 \to 3 \to 6 \to 7 \to 8,$$

который начинается в стартовой вершине, кончается в терминальной и вырисовывает строку ab. Значит, эта строка подходит под регулярное выражение и ответ получен.

А если была бы строка ac, то как бы мы поняли что она вообще не подходит? Берем стартовую вершину 1. Ее дельта-расширение – это вершины $1, 2, 3, 4$. Дальше стараемся пойти по a 1 раз. Единственная вершина из которой это можно сделать –это 4. Вершины $1, 2, 3$ отмирают, вершина 4 переходит в вершину 5. Дальше, как только сделали именной ход, снова строим дельта-расширение, получаем $5, 7, 8, 1, 2, 3, 4$ – все вершины, их которых можно дойти от 5 по дельта-путям. Теперь нам предстоит пойти по ребру c, но такого нет, поэтому все вершины отмирают и мы приходим в пустое состояние, в котором мы не находимся ни в одной вершине. Такое как раз и означает, что в терминальную мы никогда не придем и строка ac не подходит под регулярное выражение.

Финальный алгоритм. Дано регулярное выражение и строка. Строим по регулярному выражению автомат. Автомат – это класс из вершин и ребер, причем нам достаточно будет ровно одной стартовой и ровно одной терминальной вершины в каждом автомате.

Пробегаемся по регулярному выражению, если наткнулись на $|$, то надо разбить его на две части – то что до палочки и то, что после. Применить к обоим построение автомата по рекурсии, дальше сочленить два автомата в один по процедуре, описанной выше.

Если мы натыкааемся на скобочку (, надо просканировать все до соответствующей ей закрывающейся скобки (не до первой попавшейся, потому что могут быть скобки внутри скобок ((()), взять выражение в скобках, построить для него автомат по рекурсии. Построить для всего что до (и для всего, что после) автоматы тоже по рекурсии, потом сочленить все автоматы в один.

Если нактнулись на $*$ или на $+$. Если перед ними не было скобочек, то они относятся лишь к последнему символу. Например $ab+$ означает букву a и сколько угодно букв b, а $(ab)+$ означает слово $abababab$.... Тоже вызываем для того, что непосредственно перед $+$ или $*$ рекурсивное построение автомата. Зацикливаем этот автомат, как нужно, и получаем требуемый.

После получения автомата нужно взять строку, которую мы будем проверять и делать следующее: хранить набор из одной вершины – стартовой. Каждый раз делать дельта-расширение + переход по букве. Потом опять дельта-расширение и переход по букве. Дельта-расширение делается через BFS, переход по букве делается тривиально циклом по имеющимся вершинам. Если в итоге множество вершин оказалось пустым, то слово не подходит. Если мы честно дошли до конца слова, и множество не пустое, надо проверить, есть ли в нем терминальная вершина. Если есть – слово подходит, нет – нету.

Инструкции к написанию. Пишем граф с ориентированными именованными ребрами. Добавляем фиксированную стартовую и фиксированную терминальную (одной терминальной вершины достаточно в задаче) вершину. Реализуем функции сочленения нескольких автоматов различными способами.

Строим по регулярному выражению автомат и проверяем строку на нем.