ФПМИ МФТИ

Теория формальных языков
Алгоритм LR(k)

LR(k) алгоритм или заклинание, которое позволяет изгнать строгого преподавателя на экзамене. Часть вторая

Докажем оставшиеся нерасмотренные факты для LR(k) разбора, чтобы сделать рассказ полным.

Определение. Грамматика является LR(k) грамматикой, если из условий:
1) $S' \Rightarrow^* \alpha A w \Rightarrow \alpha\beta w$
2) $S' \Rightarrow^* \gamma B x \Rightarrow \alpha\beta y$
3) $First_k(w) = First_k(y)$

Следует, что $\alpha A y = \gamma B x$, то есть $\alpha = \gamma$, A=B, x=y.

Обратите внимание, что не обязательно $w = y$, так как выводы могут быть для разных слов. Более того, $x = y$ может содержать нетерминалы, если последнее правило второго вывода было $B \to \eta_1\beta\eta_2$.

Более неформально, определение говорит, что какое бы слово не выводилось, по стеку и следующим k буквам можно однозначно восстановить один шаг вывода.

Теорема 5.9

Грамматика является LR(k) грамматикой т. и т.т. когда следующие две ситуации не бывают одновременно допустимы для некоторого активного префикса $\alpha\beta$: $A \to \beta\cdot$, u и $A_1 \to \beta_1 \cdot \beta_2$, v и $u \in EFF_k(\beta_2, v)$. Напомним, что EFF_k — это такие префиксы выводимых слов, что если первый символ является нетерминальным, то он не раскрывается как пустое слово. При этом $beta_2$ может быть пустым (тогда эта ситуация порождает конфликтный reduce).

Теорема 5.10 Напомним, что ситуация $A \to \beta_1 \cdot \beta_2, u$ допустима для префикса $\alpha\beta_1$, если и только если существует вывод $S \Rightarrow^* \alpha A w \Rightarrow \alpha\beta_1\beta_2 w$ и $u = FIRST_k(w)$.

Как построить множества допустимых ситуаций? Для этого построим множество $V_k(\epsilon)$ следующим образом: включим ситуацию $S' \to \cdot\S, \epsilon$. Далее, пока множество ситуаций меняется, если есть ситуация $A \to \cdot B\alpha, u$ и правило $B \to \cdot\beta$, то добавим ситуацию $B \to \cdot\beta, x$, $x = FIRST_k(\alpha u)$.

Далее нужно научиться строить ситуации для непустых префиксов. Отметим, что если некоторая строка является активным префиксом, то и её префикс тоже является (нужно рассмотреть соответствующий шаг вывода). То есть, двигаясь по одному символу, можно построить всю систему множеств допустимых ситуаций. Пусть построено $V_k(X_1, ..., X_{i-1})$, где X_j — терминальный или нетерминальный символы. Построим $V_k(X_1, ..., X_i)$. Для этого находим в $V_k(X_1, ..., X_{i-1})$ ситуации вида $A \to \alpha \cdot X_i\beta, u$ и добавляем по ним ситуации $A \to \alpha X_i \cdot \beta, u$ в $V_k(X_1, ..., X_i)$. Далее замыкаем ситуации так же, как и в построении допустимых ситуаций по пустому слову. GOTO строится тривиальным образом: $GOTO(V_k(X_1, ..., X_{i-1}), X_i) = V_k(X_1, ..., X_i)$. Докажем, что V_k это множества допустимых ситуаций и только они.

Очевидно, что алгоритм построения конечен, так как конечно множество ситуаций. Также очевидно, что $V_k(\epsilon)$ строится корректно (при замыкании явно перебираются все правила, оставляющие префикс пустым). Доказываем индукцией по длине префикса. Пусть ситуация $A \to \beta_1 \cdot \beta_2, u$ допустима для префикса $X_1...X_i = \alpha\beta_1$. Тогда, по определению, есть вывод $S \Rightarrow^* \alpha A w \Rightarrow \alpha\beta_1\beta_2 w$. Если $beta_1$ непуст, то $\beta_1 = \beta_1' X_i$ и, по предположению индукции, $A \to \beta_1' \cdot X_i\beta_2, u$ допустима для префикса $X_1...X_{i-1}$ и попадает в $V_k(X_1...X_{i-1})$. Тогда при построении $V_k(X_1...X_i)$ ситуация $A \to \beta_1 \cdot \beta_2, u$ будет добавлена.

Сложнее случай, если $beta_1$ пуст. Тогда $\alpha = X_1...X_i$. Так как вывод $S \Rightarrow^* \alpha A w$ правый, то есть некоторый шаг, на котором появится X_i. Выделим его: $S \Rightarrow^* \alpha' B y \Rightarrow \alpha'\gamma X_i\delta y \Rightarrow^* \alpha A w$ и $\alpha'\gamma = X_1...X_{i-1}$ и вывод $\alpha'\gamma X_i\delta y \Rightarrow^* \alpha A w$ раскрывает только символы правее X_i. Тогда, по предположению индукции, ситуация $B \to \gamma \cdot X_i\delta, v$ $v = FIRST_k(y)$ допустима для $X_1...X_{i-1}$ и лежит в $V_k(X_1...X_{i-1})$. И тогда ситуация $B \to \gamma X_i \cdot \delta, v$ будет добавлена в $V_k(X_1...X_i)$. Теперь рассмотрим вывод $\delta y \Rightarrow^* Aw$. Так как в правой части первая буква — нетерминальная, то все шаги вывода применяют правило $D_i \to D_{i+1}\theta_i$, где $D_m = A$ — последний нетерминал. Тогда ситуация $A \to \cdot\beta_2, u$ будет добавлена после $m-1$ замыкания. Так как $\delta y \Rightarrow^* Aw$, то $u = FIRST_k(w)$, то есть и θ_i — это строки из терминальных букв, причём не противоречащие u.

То есть достаточность установлена. Установим необходимость. Пусть ситуация $A \to \beta_1 \cdot \beta_2, u$ лежит в $V_k(X_1, ..., X_i)$. Докажем, что она допустима индукцией по шагам алгоритма добавления. Когда множество $V_k(X_1, ..., X_i)$ пустое, то ситуация могла быть добавлена только не как замыкание, значит $\beta_1 = \beta_1' X_i$ и в $V_k(X_1, ..., X_{i-1})$ лежала ситуация $A \to \beta_1' \cdot X_i\beta_2, u$. Тогда, по предположению индукции, есть вывод $S \Rightarrow^* \alpha A w \Rightarrow \alpha\beta_1' X_i\beta_2 w$ и $\alpha\beta' = X_1...X_{i-1}$. Значит, этот же вывод подходит под определение ситуации $A \to \beta_1' X_i \cdot \beta_2, u$ и префикса $X_1...X_i$. Переход индукции. Пусть множетсво $V_k(X_1, ..., X_i)$ непусто. Тогда, если несмотря на это, очередная ситуация добавляется проносом точки, то применимо рассуждение, как для пустого множества ситуаций. Пусть ситуация теперь добавляется как замыкание. Тогда $\beta_1 = \epsilon$ и есть ситуация $B \to \gamma \cdot A\delta$, порождающая текущую ситуацию. По предположению индукции, порождающая ситуация была добавлена раньше и, значит, она допустима для $X_1...X_i$. Тогда есть вывод $S \Rightarrow^* \alpha' B y \Rightarrow \alpha'\gamma A\delta y$, что $\alpha'\gamma = X_1...X_i$. Тогда добавим к этому выводу раскрытие нетерминала A по правилу $A \to \beta_2$ и обозначим $u = FIRST_k(\delta v)$. Заметим, что ровно такой u и добавляется при замыкании, значит ситуация допустима.

Таким образом, мы построили систему допустимых ситуаций корректно, а значит, LR(k) анализатор корректен (вместе с теоремой 5.9 доказано, что в нём нет противоречий). Побочным продуктом мы получили алгоритм проверки LR(k) определения: построить анализатор и проверить его на непротиворечивость. Рассмотрим некоторые свойства построенного анализатора:

- Если запоминать последовательность правил, по которым сделана свёртка (будем обозначать этот вывод как π, то можно доказать, что на каждом ша-

ге работы анализатора верен факт: $\gamma x \Rightarrow_\pi w$, где γ — строка нетерминалов (без состояний) на стеке, x — ещё непрочитанное слово. Отсюда следует, что если на стеке есть буква S, а слово прочитано до конца, то слово действительно выводится. То есть алгоритм корректен и принимает только выводимые слова. Более того, если на каком-то шаге для префикса на стеке не существует такого суффикса x, содержащего первые k букв, которые алгоритм подсматривает при очередном шаге разбора, что при присоединении x к стеку нельзя вывести слово из языка, то алгоритм выдаёт ошибку (так как не может построить очередное правило вывода). Значит, анализатор выдаёт ошибку при первой возможности её выдать (для этого не требуется знать всё слово, только первые k букв).

- Можно построить ДМП-автомат (МП-автомат, такой, что на каждом шаге можно однозначно определить следующий шаг), эмулирующий LR-анализатор. Для этого нужно хранить очередные k букв в состоянии (а их конечно много), и анализатором "читать" букву, находящуюся на k единиц левее по слову, чем очередная прочитанная ДМП-автоматом буква (автомат намеренно читает больше букв, чтобы сделать выбор для анализатора, который опаздывает на k букв). Если Вы не идёте на отл(10), этот факт Вам не пригодится.

По свойству 1 алгоритма доказано, что он принимает только выводимые слова. Также доказано, что если алгоритм выдаёт ошибку, то слово невыводимо (более того, верен более сильный факт, см. свойство 1). Осталось доказать то, что алгоритм работает за конечное время.

На самом деле, верен более сильный факт о том, что он работает за линейное время. Так как это задача на отл(10) на экзамене, приведём только скелет доказательства.

Введём понятие C-конфигурации. Назовём начальную конфигурацию C-конфигурацией, конфигурацию, получаемую после другой C-конфигурации шагом переноса или шагом свёртки, который укорачивает стек (на стеке меньше символов, чем в предыдущей C-конфигурации. Замечание: короче не чем на предыдущем шаге, а именно на предыдущей C-конфигурации). Заметим, что укорачивание стека играет ключевую роль в анализе. Введём потенциал конфигурации, равный сумме числа терминальных и нетерминальных символов грамматики на стеке и удвоенного числа непрочитанных букв в слове. Тогда такой потенциал уменьшается на 1 от перехода к новой C-конфигурации. У начальной конфигурации он равен $2n$, значит C-конфигураций не более, чем $2n$. Осталось рассмотреть максимальное число конфигураций между C-конфигурациями. По свойству 2 построим ДМП-автомат, который работает со стеком также, как LR-анализатор (правда имеет чуть более сложные состояния). О ДМП-автоматах известно, что существует некоторое число N, зависящее только от автомата, что если он находится в состоянии q и имеет стек длины l, а потом делает N шагов, так что стек не становится короче, чем l, то существует зацикливающийся путь (не важно, удлиняющий стец, или нет). Но LR(k) анализатор не возвращает ошибку, значит существует суффикс, который можно на каждом шаге добавить к строке на стеке и вывести некоторое слово из языка. Так как не происходят переносы, этот суффикс может быть одним для всего множества стеков циклического исполнения. Значит, существует слово, имеющее сколь угодно длинные правосторонние выводы. Значит, грамматика неоднозначна. Покажем, что неоднозначная грамматика не может быть LR(k) грамматикой ни при каком k. Пусть есть два вывода одного и того же слова. Рассмотрим самый последний шаг, на котором очередные строки в выводе не равны: $S \Rightarrow \alpha_1 \Rightarrow \alpha_2 ... \Rightarrow \alpha_n = w$ и $S \Rightarrow \beta_1 \Rightarrow \beta_2 ... \Rightarrow \beta_m = w$ и $\alpha_{n-i} \neq \beta_{m-i}$, причём i минимально. Рассмотрим выводы только до этих шагов и определение LR(k)-грамматики. Оно не выполняется (на следующем шаге стеки равны, так как i было взято минимальным). Более того, так как выводится одно и то же слово, то для любого k определение не выполняется.