Student: Maksim Kryuchkov
Group: M3339
Date: 24 октября 2018 г.

---

17 октября

wow

Рассмотрим свободную грамматику
$E \to TE'$
$E' \to +TE'|\epsilon$
$T \to n|(E)$
Строим дерево, там есть множество нераскрытых нетерминалов.
Будем разбирать нерекурсивно, храня на стеке множество нераскрытых нетерминалов
Изначально на стеке лежит стартовый нетерминал
По какому правилу раскрыть? Смотрим на правые части, смотрим на first от них, выбираем то, правило, где наш первый символ строки входит в first.

Алгоритм

Стековая машина, управляющая таблица

Таблица: строки - терминалы, нетерминалы, дно стека. Столбцы: терминалы, $

- (терминал, терминал) : проверяем равны ли, сдвигаем указатель, убираем символ со стека, иначе ошибка
- (нетерминал, символ) : смотрим на все правила, выбираем то, у которого символ входит в first, иначе если наш нетерминал раскрывается в эпсилон, то смотрим в follow.
- (терминал, конец строки) : ошибка
- дно стека : матчится только с концом строки

Грамматика регулярных выражений
$E \to E|A$
$E \to A$
$A \to AF$
$A \to F$
$F \to Z^*$
$F \to Z$
$Z \to c$
$Z \to (E)$
Правое ветвление
Когда есть два правила: $A \to \alpha\beta$
$A \to \alpha\gamma$
Избавление от правого ветвления: $A \to \alpha A'$
$A' \to \beta, A' \to \gamma$
Избавляемся от правого ветвления и от левой рекурсии
Посчитали first

Восходящий разбор

Возьмем грамматику арифметичексих выражаний. $E \to E + T, E \to T, T \to T * F, T \to F, F \to n, F \to (E)$

2 + 2 + 2 + 2. Для каждого символа понимаем, что что из него раскрылось, хуяк хуяк все получилосб

Def *Перенос-свертка, shift-reduce*
Два стека: левый и правый. Слева будет разбор, справа будет слово.
Изначально справа лежит все выражение
Два возможных варианта:
1) перенос одного символа правого стека в левую
2) Сворачиваем какую-то часть левого стека (сверху, разумеется)

$\alpha \mid t$ - имеем такую строку
$handle = \gamma : \beta\gamma x = \alpha t$, при этом x - суффикс t
LR(0) автомат.
LR(0) ситуация(LR(0) item) - пара из правила и числа i, где $0 \leq i \leq |\alpha|$ - состояния автомата нашего
Переходы: $(A \to \alpha.B\beta) \to_B (A \to \alpha B.\beta)$
$(A \to \alpha.B\beta) \to_\epsilon (B \to .\gamma)$
$(A \to \alpha.c\beta) \to_c (A \to \alpha c.\beta)$
Добавим нетерминал E_0 - триггер того, что мы успешно разобрали слово.
Теорема
Строка α допускается в состояние $(A \to \gamma.), \alpha = \beta\gamma, \exists t \in \Sigma^* : S \Rightarrow^* \beta A t \Rightarrow \beta\gamma t$.
Детерминизация.
Состояниями детерминированного автомата будут замкнутые множества LR(0) ситуаций.
Переходы

Def *LR(0) грамматика*
Если в получившемся автомате нет состояний, которые одновременно содержат терминальную и нетерминальную ситуацию, или содержат два и более терминальных состояния

Def *LR(0) разбор*

(1) Построить автомат
(2) Детерминизировать
(3) Применяем перенос-свертка. Для принятия решения перенос-свертка применяем наш автомат на содержимое левого стека. Если мы оказываемся в состоянии, где лишь одна терминальная ситуация, применяем свертку, иначе перенос

Def *SLR алгоритм*

(1) Строим автомат
(2) Детерминизируем
(3) Нужно принять решение.
(4) Пропускаем левый стек через автомат. Если там только нетерминальные, то делаем перенос.
    Если есть терминальные. Если верхний символ правого стека принадлежит только одному из нетерминалов, которые мб получены сверткой, то делаем этот переход свертки

Коротко о FOLLOW
$A \to Bc, A \to xBb, A \to xC, B \to y, C \to yc$
xyc

Def *LR(1) ситуация*
$\langle A \to \alpha, i, c \in \Sigma \cup \{\epsilon\}\rangle$

Переходы:
$(A \to \alpha.X\beta, c) \to_X (A \to \alpha X.\beta, c)$
$(A \to \alpha.B\beta, c) \to_\epsilon (B \to \gamma, d), d \in FIRST(\beta c)$
Делаем тут еще в $|FIRST(\beta c)|$ раз больше переходов

Def *LR(1) разбор*

(1) Построить автомат
(2) Детерминизировать
(3) Если есть терминальная ситуация с некоторым символом предпросмотра, то делаем тот переход, в котором есть этот символ. Соответственно LR(1) грамматика - такая что в получившемся автомате нет конфликтов по свертке / свертке-переходу

24 октября

КЕК

Def *Атрибут-транслирующая грамматика*
Атрибуты - поля терминалов/нетерминалов
Транслирующие символы - нетерминалы, у которых единственное правило из этого нетерминала в эпсилон

Example
$E \to E + T, E \to T, T \to T * F, T \to F, F \to n, F \to (E)$
$n, E, F, T : val$
$E \to E + T : \{E_0.val = E_1.val + T_3.val\}$ - правила при раскрытии нетерминалов
$E \to T : E_0.val = T_2.val$ - правило копирования атрибутов
Рассмотрим версию без левой рекурсии
E' acc, val
T acc, val - атрибуты
Синтезируемые и наследуемые атрибуты
Синтезируем - если зависит от детей данного нетерминала в дереве разбора, присваиваются в момент выхода из нетерминала в дереве разбора
Наследуем - если зависит от родителей или братьев, присваиваются в момент входы в нетерминал.
val - синт.
acc - наследуемый
Если рассматривать нетерминал как функцию, то синт. атрибуты - это возвращаемые значения, а наследуемые - это параметры
$E \to TE'$
$E' \to +T$, соответственно E' записывает себе в аккумулятор значение T - левого брата и считает val на поддереве