Отладка грамматик простого предшествования

Метод синтаксического разбора для грамматик простого предшествования самый простой, но, тем не менее, он иллюстрирует многие приемы синтаксического анализа, которые встречаются в других алгоритмах синтаксического анализа для грамматик других классов. С теоретической точки зрения рассмотренный метод кажется безупречным и эффективным. Однако на практике он не всегда хорош. Более того, почти любая другая техника разбора оказывается лучше метода простого предшествования. Дело в том, что очень часто между двумя символами определено более чем одно отношение. Единственное, что можно сделать в этом случае – это обработать и изменить грамматику так, чтобы обойти этот конфликт. Но в результате может измениться вся структура языка, не говоря о том, что грамматика станет неудобочитаемой. Далее перечисляются причины возникновения конфликтов и способы их ликвидации (обхода).

Как указывалось, для того, чтобы грамматика являлась грамматикой простого предшествования, она должна удовлетворять двум условиям:

* между любыми двумя символами из словаря должно быть определено не более чем одно отношение;
* ни у каких двух продукций не должно быть одинаковых правых частей.

Несоответствие грамматики этим условиям – это **конфликт**.

Конфликт может возникнуть как следствие левосторонней рекурсии. Предположим, что существует правило U::=U…. Если есть другое правило вида V::=…SU…, то получается, что одновременно **S=U и S<U**.

Можно избавиться от такого конфликта вводя еще один нетерминал W и промежуточное правило. Заменим

V::=…SU…

на

V::=…SW…, W::=U,

где W –новый символ. Тогда получим, что S=W и S<U. Этот прием называется **стратификацией** или **разделением**.

При правосторонней рекурсии такие же проблемы возникают с отношениями **> и =**. Предположим, что существует правило U::=…U. Если есть другое правило вида V::=…SU…, то получается, что одновременно S=U и S>U.

Можно избавиться от такого конфликта также, как в случае с левосторонней рекурсией, вводя еще один нетерминал W и промежуточное правило. Заменим

V::=…SU…

на

V::=…SW…, W::=U,

где W –новый символ. Тогда получим, что S=W и S>U.

К сожалению, применение стратификации может привести к конфликтам иного рода и поэтому помогает не во всех ситуациях.

От **правил с одинаковыми правыми** частями можно попробовать избавиться, соответствующим образом изменив грамматику, так как, обычно, такие правила избыточны и не являются остро необходимыми.

Что касается конфликта, когда **R<S и R>S** одновременно, то лучший выход – применить другую технику.

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Данное руководство описывает принципы работы, формат сообщений, файлов и ошибки программы для анализа грамматик на несоответствия (конфликты) ограничениям класса грамматик простого предшествования GRAMMAR.EXE.

Системные требования.

Модуль GRAMMAR.EXE является консольным приложением Win32 и поэтому для его функционирования необходим компьютер семейства x86 под управлением ОС семейства Microsoft Windows с поддержкой Win32-приложений.

Формат командной строки.

Модуль GRAMMAR.EXE принимает данные от пользователя через аргументы командной строки:

GRAMMAR.EXE <grammar\_file> [/NODETAIL] [/TAB:<ASCII\_code>]

<grammar\_file> - файл с грамматикой в НБФ, данный формат описан далее. Это обязательный параметр.

/NODETAIL – при использовании этого параметра в файле-отчете не выводятся матрицы и правила – выводятся только ошибки и сообщения об успешной работе. Это не обязательный параметр. Данный параметр полезен при отладке грамматики, когда достаточно только сведений об ошибках и не нужна прочая информация (которая к тому же может мешать).

/TAB:<ASCII\_code> - данный параметр заменяет разделитель элементов в выводимых матрицах (по умолчанию это символ ;) на символ соответствующий ASCII-коду <ASCII\_code>. Например /TAB:9 определяет в качестве разделителя символ табуляции (имеющий ASCII-код 9), что позволяет удобно копировать матрицы через буфер обмена в программу электронных таблицы Microsoft Excel. Данный параметр не обязателен.

Формат файла грамматики (НБФ).

Входные данные для программы – файл с грамматикой в НБФ, имя которого указывается в командной строке. Это текстовый файл. Строки файла имеют вид:

<имя\_нетерминала> ::= <имя\_нетерминала> или

”имя\_терминала”…<имя\_нетерминала> или ”имя\_терминала”

Таким образом слева находится нетерминал, справа список из нетерминалов и терминалов. Левая и правая части разделены символами ::= без пробелов между этими тремя символами.

Нетерминалы заключаются в угловые скобки: < и >.

Терминалы заключаются в двойные кавычки: ” и ”.

Вне или между терминалов (огранич. ””), нетерминалов (огранич. < >) и символов ::= допускается любое количество пробелов и символов табуляции.

Если есть необходимость использовать в именах терминалов или нетерминалов символы <,>,” или последовательность символов ::=, то они заключаются в одиночные кавычки ', например – ”'<'нетерминал'>'”. Если есть необходимость использовать в именах терминалов или нетерминалов одиночные кавычки, то три подряд записанные кавычки (''') интерпретируются как одна кавычка (').

Строка, начинающаяся с символа #, интерпретируется как комментарий. Символ # так же может предваряться символами табуляции и пробелами.

Нетерминал стоящий в левой части самого первого по порядку в файле правила считается начальным символом Z.

Файлы результатов и вывод программы.

В результате работы программа выводит на экран последовательность действий, сведения об ошибках и сообщение об успешном построении матрицы предшествования (что означает, что грамматика из входного файла является грамматикой простого предшествования).

Кроме этого, программа сохраняет сведения об ошибках и различную информацию о грамматике в создаваемом ею текстовом файле:

<имя\_входного\_файла\_без\_расширения>.LOG.

Данный файл содержит в следующем порядке:

* Кодировку нетерминалов целыми числами.
* Кодировку терминалов целыми числами.
* Кодировку правил целыми числами (представляющими терминалы и нетерминалы).
* Сообщения об ошибках разбора грамматики из файла (если такие ошибки были).
* Порядок соответствия терминалов и нетерминалов строкам и столбцам далее выводимых матриц.
* Матрицы FIRST, FIRST+, LAST, LAST+, WITHIN, WITHIN+.
* Ошибки, выявленные при проверке грамматики на приведенность.
* Матрицы отношений предшествования =, <, >.
* Ошибки, выявленные при анализе грамматики на конфликты.
* Матрицу предшествования.

Описание форматов ошибок приведено далее. После первой ошибочной стадии выполнение программы прекращается.

Также программа создает бинарный файл:

<имя\_входного\_файла\_без\_расширения>.TAB,

в котором она сохраняет данные разобранной грамматики (таблицу правил, таблицу терминалов и матрицу предшествования) для модуля синтаксического разбора. Данный файл создается только в случае удачного построения матрицы предшествования для грамматики.

Описание ошибок.

Ошибки разбора грамматики из файла.

При нарушении выше указанного формата файла с грамматикой в НБФ программа выдает список найденных при его разборе ошибок. Данное сообщение имеет следующий формат:

(*строка\_в\_файле*) : error *код\_ошибки*: *Текст\_разъясняющий\_ошибку*

Например:

(2) : error 10: Метасимволы в символе без переопределяющих символов '

Далее приводятся возможные ошибки. Коды ошибок перечислены слева, их описание справа:

1 – Нетерминал в левой части должен начинаться с <,

2 – Левая и правая части правила должны быть разделены ::=,

3 – Пустая правая часть правила,

4 – Нарушение формата правой части правила,

5 – Пустой символ,

6 – Нет завершающего метасимвола,

7 – Нет символа ' завершающего последовательность спец-символов,

8 – Пустая последовательность спец-символов,

9 – Одна из правых частей пустая,

10 – Метасимволы в символе без переопределяющих символов '

Ошибки, выявленные при проверке грамматики на приведенность.

При анализе грамматики на приведенность может возникнуть три типа ошибок:

1 - правило типа U::=U,

2 - нетерминал не выводится из Z,

3 - из нетерминала не выводится цепочка состоящая из терминальных символов.

Слева приведены коды ошибок. Данные ошибки отображаются в следующем формате:

error 1: Обнаружено правило типа U::=U для нетерминала <*имя\_нетерминала*> !,

error 2: Нетерминал <*имя\_нетерминала*> не выводится из начального символа Z !,

error 3: Из нетерминала <*имя\_нетерминала*> не выводится цепочка только из терминалов!.

Ошибки, выявленные при анализе грамматики на конфликты.

При анализе грамматики на конфликты могут быть найдены два типа конфликтов – конфликтные правила (правила с одинаковыми правыми частями) и конфликтные отношения (более одного отношения предшествования для пары символов). Ниже приведен пример вывода конфликтов:

Конфликтные правила (одинаковая правая часть):

<E>::=<E>"+"<T>

<A>::=<E>"+"<T>

<X>::=<E>"+"<T>

<F>::="("<E>")"

<B>::="("<E>")"

<F>::="i"

<B>::="i"

Конфликтные отношения:

"+" = <T> "+" < <T>

"(" = <E> "(" < <E>

Таким образом, правила с одинаковыми правыми частями выводятся одно за другим построчно, группы правил с одинаковыми правыми частями разделяются пустой строкой.

Конфликтные отношения выводятся одно за другим, каждой паре символов соответствует отдельная строка.

Системные ошибки, возникшие в процессе выполнения программы.

* GLOBAL ERROR: Недостаточно памяти! – Данное сообщение указывает на недостаток оперативной памяти.
* GLOBAL ERROR: Параметр вне допустимых пределов! – Данное сообщение скорее всего появляется в результате нарушения приведенных далее ОГРАНИЧЕНИЙ.
* При нарушении формата командной строки программа выводит справку о формате командной строки:

Формат командной строки:

GRAMMAR.EXE <grammar\_file> [/NODETAIL] [/TAB:<ASCI\_code>]

<grammar\_file> - файл с грамматикой в НБФ

/NODETAIL - не показывать матрицы и правила

/TAB:<ASCI\_code> - заменить разделитель в матрицах с ;

на <ASCI\_code>

* ERROR: Ошибка открытия файла:*имя\_файла* – невозможно открыть указанный файл с грамматикой, проверьте пути и существование файла.
* ERROR: Неправильный формат файла! – файл, который вы задали в качестве входного, пустой.
* ERROR: Ошибка открытия лог-файла (*имя\_файла*)! – не возможно создать рассматривавшийся выше файл для вывода отчета (.LOG).
* ERROR: Ошибка открытия файла для сохранения данных (*имя\_файла*)! – не возможно создать рассматривавшийся выше файл для вывода данных для модуля синтаксического разбора (.TAB).
* ERROR! – нераспознанный код ошибки.
* GLOBAL ERROR: Попытка последоват. работы со списком при не заданной позиции!
* GLOBAL ERROR: Несоблюдение типов при работе с бинарной матрицей!
* GLOBAL ERROR: Несоблюдение размеров бинарных матриц при сложении!
* GLOBAL ERROR: Несоблюдение размеров бинарных матриц при умножении!
* GLOBAL ERROR: Транзитивное замыкание реализовано только для квадратных матриц!
* GLOBAL ERROR: Недопустимое количество нетерминалов в бинарной матрице!

Ограничения.

На этапе компиляции, исходя из соображений разумной достаточности, в данной программе задаются следующие жесткие ограничения:

Максимальное количество символов (литералов) в строке файла с грамматикой в НБФ равно 1023.

Максимальная длина цепочки символов (литералов) описывающих терминал или нетерминал (включая ограничивающие символы) равна 255.

Максимальное количество терминалов и нетерминалов задается в результате указания нижней и верхней границ для кодировки терминалов (от 101 до 300) и нетерминалов (от 1 до 100) целыми числами.