ТЕХНОЛОГИЧЕСКИЙ КОМПЛЕКС РАЗРАБОТКИ ЯЗЫКОВЫХ ПРОЦЕССОРОВ

Б.К.Мартыненко

Введение

Многие проблемы применения ЭВМ для обработки текстовой информации представляются как проблемы спецификации и реализации трансляций, т.е. отображений из одного (входного) языка е другой (выходный) язык. Одной из наиболее часто используемых Форм реализации таких отображений являются языковые процессоры,

В данном сообщении описываются методологические основы технологического комплекса, предназначенного для разработки и построения языковых процессоров различного назначения. В частности, такие процессоры могут использоваться в качестве компонент автоматизированных систем информационного обслуживания, автоматизированных систем обучения и инструментальных средств автоматизации программирования. Описываемый технологический комплекс может быть полезен, прежде всего, при разработке трансляторов языков программирования, в особенности, для реализации фаз лексического и синтаксического анализа, при разработке программ ввода данных сложной структуры с контролем их правильности и первичной обработкой, а также в учебном процессе при изучении вопросов технологии трансляции. Для построения языковых процессоров применяется трансформационный подход.

Трансляционные грамматики

Для первоначальной спецификации трансляций в рассматриваемой технологии используются трансляционные КС-грамматики. Трансляционная КС-грамматика отличается от обычной КС-грамматики тем, что помимо алфавитов нетерминалов и терминалов, в нее включается еще один алфавит - алфавит семантик, интерпретируемых как преобразования некоторого окружения. Трансляционная грамматика порождает синтаксическое управление - множество управляющих цепочек над

терминалами и семантиками. Управляющая цепочка задает как предложение входного языка (если в ней игнорировать семантики), так и способ его обработки (трансляции), определяемый ее семантиками. Задача состоит в том, чтобы по заданной трансляционной КС-грамматике построить процессор, который по предложению входного языка восстанавливал бы всю управляющую цепочку и выполнял определяемые ею семантические преобразования.

Языковые процессоры

В рассматриваемой технологии для реализации трансляций используются детерминированные МП-процессоры челночного типа. Каждый такой процессор представляет собой пару детерминированных МП—автоматов, наделенных способностью преобразовать окружение, в котором они работают, фактически, это окружение есть то пространство, в котором определены упомянутые выше семантические преобразования. Состояния этих (управляющих) автоматов несут определенную грамматическую информацию и используются для представления результата бесконтекстного синтаксического анализа входной строки. Окружение представляет исходный контекст, в котором интерпретируется входная строка, а также служит для передачи и хранения информации, извлекаемой из самой входной строки е процессе ее анализа.

Первый из названных автоматов решает задачу распознавания входного языка. Читая входную строку, он обнаруживает и диагностицирует ошибки, определяет ее бесконтекстную синтаксическую структуру с точностью до имен составляющих подконструкций (нетерминалов), представляя ее на выводе как цепочку своих собственных состояний.

Второй автомат, сканируя в обратной последовательности выход первого автомата, завершает синтаксический анализ входной строки, уточняя имена составляющих подконструкций. К этому моменту полностью воссоздается управляющая цепочка, определяющая способ трансляции входной строки. Однако исполнение семантических преобразований, определяемых полученной управляющей цепочкой, произво-

дится не после того, как она построена, а в процессе ее воссоздания обоими автоматами. Соответственно семантики подразделяются на прямые и обратные в зависимости от того, какой из упомянутых автоматов инициирует их исполнение, и это подразделение отражается уже в исходной трансляционной КС-грамматике. Окончательное состояние окружения представляет результат трансляции входной строки. Во многих практически полезных случаях для обработки входной строки оказывается достаточно одного первого автомата.

Эквивалентные преобразования трансляционных грамматик

Для того, чтобы существовал процессор рассматриваемого- класса, реализующий трансляцию, специфицированную данной трансляционной КС-грамматикой, необходимо, чтобы последняя удовлетворяла некоторым условиям. Технологический комплекс содержит компоненту, проверяющую выполнение этих условий. В случае их нарушения она выдает информацию о характере этих нарушений, по которой, как правило, удается найти эквивалентные (т.е. сохраняющие трансляцию) преобразования исходной трансляционной грамматики к требуемому классу.

К сильно эквивалентным (т.е. сохраняющим синтаксическое управление) преобразованиям относятся: вынесение общих префиксов или суффиксов из нескольких альтернатив правила для одного нетерминала, подстановка правой части правила для некоторого нетерминала вместо его вхождения в правую часть правила для некоторого другого нетерминала, введение правил для новых вспомагательных нетерминалов, замена лево- (право-) рекурсивных определений нетерминалов итеративными. В общем случае такие преобразования приводят трансляционную КС-грамматику к обобщенной регулярной форме Бзкуса-Наура, в которой каждое правило имеет вид: $A \rightarrow R$, где A - нетерминал, а R - некоторое регулярное выражение над тремя алфавитами грамматики.

Другие (не сильно) эквивалентные преобразования связаны с изменением расстановки семантик в правилах грамматики или даже с заменой всей системы семантик.

Во многих случаях удается ограничиться сильно эквивалентными преобразованиями.

Трансляционные граф-схемы

являются еще одним аппаратом задания трансляций, который аналогичен трансляционным КС-грамматикам, но обладает лучшими технологическими качествами благодаря тому, что они непосредственно допускают автоматную интерпретацию, ведущую к конструктивной реализации трансляций.

Трансляционная граф-схема есть ориентированные псевдограф, в общем случае состоящий из нескольких компонент, вершины которых помечены нетерминалами и терминалами, а дуги - цепочками семантик» Каждая компонента имеет две выделенных вершины: начальную, помеченную меткой "начало-А", в которую не входит ни одна дуга, и конечную, помеченную меткой "конец-А", из которой не выходит ни одна дуга, где А-некоторый нетерминал. Считается, что такая компонента опредялет нетерминал А.

Рассмотрим произвольный полный маршрут в компоненте, определяющей начальный нетерминал. Под полным маршрутом подразумевается маршрут, проходящий от начальной до конечной вершины компоненты. Определим след маршрута как цепочку меток вершин и дуг в порядке их следования в рассматриваемом маршруте. Каждый нетерминал полученного следа заменим на след некоторого полного маршрута в компоненте для соответствующего нетерминала. Множество всех следов, образованных посредством повторения таких подстановок, состоящих из терминалов и семантик, образует синтаксическое управление, определяемое данной трансляционной граф-схемой. Трансляция, задаваемая данной трансляционной граФ-схемой, определяется через синтаксическое управление так же, как для трансляционных КС-грамматик.

Существует алгоритм сильно эквивалентного преобразования любой трансляционной КС-грамматики в обобщенной регулярной Форме Бэкуса-Наура и соответствующей трансляционной граФ-схеме. Комплекс содержит модуль, реализующий этот алгоритм.

Далее трансляционная граф-схема используется для построения управляющих таблиц процессора или для непосредственного управления его действиями. В первом случае обеспечивается экономия времени, во втором - экономия памяти во время работы самого процессора.

Технология предусматривает автоматическую минимизацию как управляющих таблиц, так и трансляционных граф-схем.

Управляющие таблицы челночного МП-процессора

В отличие от МП-преобразователя, управляемого одной таблицей с тремя входами (состояние, входной символ, верхний символ магазина), первый управляющий автомат челночного МП-процессора управляется двумя таблицами, с двумя входами каждая. Первая из этих таблиц по текущему состоянию и текущему входному символу (независимо от верхнего символа магазина) определяет семантическую цепочку, магазинную цепочку и переходное состояние, либо указывает на необходимость использования второй таблицы. Вторая таблица по текущему состоянию и верхнему символу магазина (независимо от текущего входного символа) определяет переходное состояние. Автомат инициирует исполнение семантической цепочки, магазинную цепочку помещает в магазин и переходит в указанное переходное состояние, продвигая входную головку к следующему входному символу. При использовании второй таблицы верхний символ магазина удаляется, и автомат переходит в определенное второй таблицей переходное состояние, не продвигая входной головки.

Второй автомат челночного МП-процессора управляется одной таблицей, которая по текущему состоянию и входному символу, т.е. некоторому состоянию первого автомата, определяет магазинную цепочку (состоящую из одного магазинного символа или пустую) и переходное состояние, либо указывает, что переходное состояние нужно взять с вершины магазина. Второй автомат записывает магазинную цепочку в магазин и переходит в указанное состояние, продвигая входную головку в обратном направлении, либо снимает верхний символ магазина и использует его в качестве нового текущего состояния. В последнем случае входная головка не продвигается.

Таким образом, челночный МП-процессор использует три таблицы с двумя входами каждая, что требует значительно меньше памяти чем две таблицы с тремя входами для обычных МП-автоматов.

Построение управляющих таблиц прямого просмотра

Управляющие таблицы первого автомата челночного МП-процессора строятся по трансляционной граф-схеме. Попутно проверяются условия, гарантирующие эквивалентность МП-процессора, использующего эти таблицы, трансляционной граф-схеме.

Первоначально в множество состояний автомата включается одно начальное состояние, представленное начальной вершиной компоненты трансляционной граф-схемы для начального нетерминала. Каждое из остальных состояний представляется как некоторое подмножество однородных (нетерминальных или терминальных) вершин граф-схемы. Пополнение множества состояний производится следующим образом. Рассматривается каждое состояние из множества состояний. Для каждой вершины рассматриваемого состояния находятся все смежные вершины, которые в совокупности образуют множество вершин нулевого уровня. Если множество вершин текущего уровня содержит нетерминальные вершины, строится множество вершин следующего уровня. В него включаются все вершины, смежные с начальными вершинами компонент граф-схемы для нетерминалов, помечающих вершины текущего уровня, и т.д. Совокупность множеств вершин всех уровней называется разложением рассматриваемого состояния.

Разложение любого состояния должно удовлетворять следующим четырем условиям: (1) Число уровней разложения должна быть конечным. (2) Любой терминал может быть меткой вершин одного уровня. (3) Конечные вершины могут встречаться лишь на одном уровне. (4) Дуги, идущие из вершин состояния к терминальным вершинам в разложении этого же состояния, помеченным одинаковыми терминалами, должны быть помечены одинаковыми цепочками прямых семантик; аналогичное условие должно выполняться для дуг, ведущих из вершин состояния в конечные вершины его же разложения.

Состояние назовем подавляемым, если его разложение содержит конечные вершины.

По построенному разложению рассматриваемого состояния для каждого терминала, встречающегося среди меток его вершин, определяется: переходное состояние, равное множеству вершин, помеченных таким

терминалом; семантическая цепочка, равная идентичным цепочкам прямых семантик соответствующих дуг и магазинная цепочка, элементами которой являются множества нетерминальных вершим предшествующих уровней, взятые в порядке возрастания уровней. Если рассматриваемое состояние подавляемое, то для всех остальных терминалов: планируется обращение ко второй таблице с предварительной загрузкой магазинной цепочки, состоящей из множеств нетерминальных вершин, предшествующих уровню конечных вершин, и инициированием семантической цепочки, определяемой соответствующими дугами из вершин рассматриваемого состояния в конечные вершины его же разложения. В противном случав для всех остальных терминалов констатируется ошибочная ситуация. При этом множество состояний пополняется новыми переходными состояниями. Один из входов второй таблицы пополняется новыми элементами магазинных цепочек, а на другой ее вход записываются подавляемые состояния. Разумеется, фактически вместо множеств и цепочек используются лишь их имена (номера). После того, как рассмотрены все состояния первой таблицы определяются элементы второй таблицы. Рассматриваются все пары ее ВХОДОВ. Для каждой пары (подавляемое состояние - элемент магазина) планируется переход в состояние, равное множеству вершин элемента магазина, помеченных нетерминалами, входящими в состав меток конечных вершин, входящих в разложение подавляемого состояния. Каждое такое состояние, называемое возвратным, включается в множество состояний первой таблицы. После этого цикл просмотра состояний пополненного множества продолжается до тех пор, пока при очередном определении элементов второй таблицы не появится ни одного нового возвратного состояния. После этого остается проверить еще одно (пятое) условие, гарантирующее адекватность получаемых управляющих таблиц. Сформулируем это условие. Для каждого подавляемого состояния построим множество состояний, включив в него данное подавляемое состояние и все возвратные состояния, соответствующие ему во второй таблице. Если какое-либо из этих возвратных состояний окажется подавляемым, включим в искомое множество также возвратные состояния для него, и

т.д., до тех пор пока не найдется ни одного не рассмотренного подавляемого состояния среди состояний построенного множества.

Терминал, помечающий некоторую вершину, входящую в разложение данного состояния, назовем допустимым в этом состоянии. Условие состоит в том, чтобы во множестве состояний, построенном описанным выше оврагом для каждого подавляемого состояния; для каждой пары различных состояний пересечение множеств терминалов, допустимых в каждом из состояний рассматриваемой пары, было пусто.

Построение управляющей таблицы обратного просмотра

Один вход в управляющую таблицу второго автомата челночного МП-процессора составляет множество состояний первого автомата. Другой вход включает множество состояний второго автомата, которое первоначально содержит одно начальное состояние, представленное конечной вершиной компоненты граф-схемы для начального нетермина-ла, а затем пополняется по мере определения элементов этой табли-цы.

Значение элемента управляющей таблицы обратного просмотрам соответствующего паре (состояние второго автомата - состояние первого автомата) определяется следующим образом. Строится под-множество вершин состояния первого автомата, включающие вершины, для которых хотя бы одна смежная вершина принадлежит состоянию второго автомата из рассматриваемой пары.

Если полученное подмножество пусто, планируется выборка пере-ходное состояния из магазина. В противном случае проверяется последнее (шестое) условие, гарантирующее сильную эквивалентность челночного МП-процессора: все дуги, соединяющие вершины найденного подмножества с вершинами состояния второго автомата, должны иметь идентичные метки относительно обратных семантик. В этом случае семантическая цепочка полагается равной инвертированной последовательности обратных семантик, помечающих любую из упомя-нутых дуг.

Если найденное подмножество состоит из терминальных вершин, оно берется в качестве переходного состояния, а магазинная цепочка считается пустой.

Если найденное подмножество состоит из нетерминальных вершин, то магазинная цепочка считается состоящей из одного элемента – магазинного состояния, представленного этим подмножеством, а переходное состояние полагается равным множеству конечных вершин компонент граФ-схемы для нетерминалов, помечающих вершины магазинного состояния. Новые переходные и магазинные состояния включаются во множество состояний второго автомата.

Если среди состояний второго автомата имеются состояния более чем из одной вершины, то трансляционная грамматика, по которой строилась трансляционная граф-схема, синтаксически неоднозначна. Однако это не является помехой детерминированной реализации трансляции, т.к. сформулированные условия относительно семантик гарантируют ее семантическую однозначность (каждому предложению входного языка соответствует ровно одна управляющая цепочка).

Состояния второго автомата, взятые в обратной последовательности, представляют порождающий маршрут в трансляционной граф-схеме для данного входного предложения. Кроме того, в момент помещения состояния в магазин определяется конец, а в момент выборки его из магазина начало терминального порождения одного из нетерминалов, помечающих вершины этого магазинного состояния (в случае синтаксической однозначности такой нетерминал единствен). Таким образом второй автомат своими состояниями представляет также и синтаксическую структуру входного предложения.

Минимизация управляющих таблиц

Опыт показывает, что число входов в управляющие таблицы можно значительно сократить за счет перехода к классам эквивалентных состояний и входных символов. Отношения эквивалентности состояний и входных символов определяются по принципу их неразличимости по реакции управляющих автоматов.

Минимизация управляющих таблиц дает информацию для соответствующей минимизации граф-схем. Минимизация трансляционных граф-схем существенна, если она используется для непосредственного управления работой языкового процессора, а также при генерации тестов для проверки неформальной правильности результирующего процессора.

Тестирование процессоров.

Теория метода гарантирует соответствие получаемого процессора специфицированной трансляции. Для проверки неформальной правильности процессора комплекс содержит генератор тестов, использующий трансляционную граф-схему. Он гарантирует минимальность длины и полноту теста для заданного критерия, который зависит от целочисленного параметра, задающего "силу" теста. Математически дело сводится к решению задачи китайского почтальона на некотором орграфе, производном от исходной трансляционной граф-схемы.

ЛИТЕРАТУРА

1. АЛГОЛ 68. Методы реализации, Под ред. Г.С.Цейтина, Л., 1976.

2. Б. К. М а р т ы н е н к о , И. З.Косинец. К обоснованной технологии реализации языков программирования: построение и тестирование конечных процессоров. Деп. в ВИНИТИ, № 6907-84. Деп., 1984.