УДК 519.688, 519.876.5

РАЗРАБОТКА ФОРМАЛЬНОГО ЯЗЫКА, ЛЕКСИЧЕСКОГО И СИНТАКСИЧЕСКОГО АНАЛИЗАТОРОВ ДЛЯ ОПИСАНИЯ МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ ДИНАМИЧЕСКИХ СИСТЕМ

Кудашев Т.Р.

Набережночелнинский институт ФГАОУ ВПО Казанский (Приволжский) федеральный университет, 423810, г. Набережные Челны, пр. Мира, д.68/19

e-mail: kudashevtim@gmail.com

поступила в редакцию 08 декабря 2014 года

Аннотация

В статье представлен способ разработки формального языка описания математических моделей и описание принципов работы синтаксического и лексического анализаторов. Данный формальный язык позволяет реализовать механизм генерации кода математических моделей для последующей работы с ними на компьютере.

Ключевые слова: *формальный язык, лексический анализатор, синтаксический анализатор.*

Введение. В настоящее время одним из основных способов проведения научных исследований является построение математических моделей реальных объектов и явлений.

Математические модели изначально строятся в аналитическом виде. Для того чтобы иметь возможность проводить исследование и решать прикладные задачи, модель должна быть реализована на ЭВМ с помощью какого-либо программного пакета. Многие пакеты используют графические средства, однако для некоторых задач более предпочтительным в плане удобства является аналитическое задание модели. Поэтому существует потребность в инструменте, позволяющем формировать цифровые математические модели, задавая их в аналитическом виде.

Для достижения наибольшего быстродействия работы цифровой модели, инструмент должен обеспечивать возможность генерации компилируемого программного кода на языке C/C++.

При реализации такого инструмента необходимо, в первую очередь, разработать формальный язык для промежуточного представления компонентов математической модели. Запись модели на этом языке генерируется на основе данных, вводимых пользователем с применением средств пользовательского интерфейса. Описание на промежуточном языке должно дополняться средствами лексического, синтаксического и семантического анализа. На основе представления модели на промежуточном языке осуществляется генерация программного кода на языке C/C++ с последующей компиляцией. Скомпилированный код может быть использован для реализации расчетов в приложениях различного назначения.

Одним из предложенных раннее способов задания промежуточного представления модели является ее представление в виде структуры XML документа. Однако данный способ является неудобным в плане реализации и не оптимальным в плане быстродействия.

В данной работе рассматривается задача разработки формального языка описания математических моделей и реализация синтаксического анализатора на основе созданного языка.

Для достижения цели работы необходимо решить следующие задачи:

- разработать формальный язык описания математических моделей;
- построить лексический и синтаксический анализатор;
- разработать приложение для демонстрации работы лексического и синтаксического анализаторов.

Основная часть. Для описания языка математических моделей используется расширенная форма Бэкуса – Наура (EBNF). Введены следующие обозначения:

- нетерминалы обозначаются произвольной символьной строкой, заключенной в угловые скобки "<" и ">";
- нетерминалы, состоящие из нескольких слов, разделены пробелами.

Базовые элементы языка:

<цифра> = 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

<буква> = [a-zA-Z]

Алфавит V языка определен следующим образом:

V = {<буква><цифра>_ , . ‘ + – * / =^ ( )@#}

Язык должен позволять записывать числа с плавающей точкой. Число с плавающей точкой представляется в следующем виде:

<число> = {/<цифра>/} [(.{/<цифра>/})]

Основной единицей формул является параметр. Параметр представляет из себя буквенное обозначение физических величин. Он должен содержать в себе хотя бы одну букву, а также дополнительно любое количество букв и цифр:

<параметр> = <буква> {[<буква><цифра>]} |<буква>{[<буква><цифра>]}<индекс>

<индекс>=_(<параметр>|<цифра>)_(<параметр>|<цифра>)

|_(<параметр>|<цифра>)

|__(<параметр>|<цифра>)

Если за параметром следует символ "_", то цифра или параметр, расположенный за ним, определяется как нижний индекс. В случае если после нижнего индекса снова расположен символ "_", то цифра или параметр, следующий за ним, определяется как верхний символ. Параметры, имеющие верхний индекс, но не имеющие нижнего индекса, определяются с помощью знака "__".

Уравнения и формулы в математических моделях можно представить как совокупность операндов и операций. Операторы разделяются на унарные и бинарные:

<унарная операция> = sin|cos|exp|ln|tan|atan|ctan|actan|sign|sh|ch|th|cth|-|’|+

Уравнения могут содержать производные. Производная от параметра "X" по параметру "t" может быть определена как бинарная операция, первым операндом которой является "X", а вторым "t" (diff(X,t)). Логарифм от числа "A" по основанию "B" также может быть определен как бинарная операция, первым операндом которой является "A", а вторым "B" (log(A,B)):

<бинарная операция> = +|–|/|*|^|diff|log

Операторы применяются над операндами:

<операнд> = <число>|<параметр>|<выражение>

Выражения и операнды определяются рекурсивно:

<выражение> = <унарная операция><операнд>

|<операнд><бинарная операция><операнд>

Равенства используются для определения параметров:

<равенство> = #<параметр>=<выражение>

Уравнения записываются следующим образом:

<уравнение> = <выражение>=<выражение>

Из уравнений состоят системы уравнений:

<система> = @<уравнение>{/(,<система>)/}

Совокупность уравнений, равенств и систем представляют собой математическую модель некоторого реального процесса или явления:

<модель> = <уравнение>{<равенство>}

| <система>{<уравнение>}{<равенство>}

Для группировки выражений используются скобки "()".

Математические модели могут содержать параметр, применяемый в разных уравнениях и системах. Для обеспечения связи между одними и теми же параметрами каждому из них присваивается идентификатор (натуральные числа от 1 до N, где N - количество параметров в математической модели). Для систем уравнений необходимо подсчитывать количество неизвестных параметров и количество уравнений в системе, проверив таким образом

возможность определения значения данных параметров. Помимо параметров, математическая модель может содержать переменные, которые изменяют свое значение с течением времени. Переменные для каждой отдельной математической модели могут иметь разное обозначение, которое должно быть задано пользователем. Если символ переменной не задается, то в качестве него по умолчанию выступает время *t*.

В грамматике языка математических моделей выполнена левая факторизация, поэтому она пригодна для построения синтаксического анализатора на основе метода предиктивного, или нисходящего, анализа [1]. Каждому из нетерминалов грамматики ставится в соответствие функция-обработчик. Принцип работы анализатора заключается в том, чтобы однозначно определить следующую вызываемую функцию-обработчик на основе текущей лексемы [2]. Данный анализатор использует метод рекурсивного спуска.

Под входной строкой понимается совокупность уравнений, систем и равенств математической модели, подаваемой на вход анализатору.

Нисходящий синтаксический анализ можно рассматривать как задачу построения дерева разбора для входной строки, начиная с корня и создавая узлы дерева разбора в прямом порядке обхода (обход в глубину). Или, иначе, нисходящий синтаксический анализ определяется как поиск левого порождения входной строки [3].

Программа синтаксического анализа методом рекурсивного спуска (recursive-descent parsing), состоит из набора процедур, по одной для каждого нетерминала [4].

В данной работе реализовано объединение работы лексического и синтаксического анализаторов в один проход. В случае реализации предварительного прохода лексического анализатора в оперативную память помещается вектор лексем, который идентичен входной строке. В случае сложных систем с большим количеством параметров и уравнений, данный фактор имеет существенное значение. Синтаксический анализатор не требует обозрения нескольких лексем одновременно, поэтому объединение лексического и синтаксического анализаторов является более эффективным подходом.

Прежде, чем строить синтаксическое дерево для математической модели, необходимо удостовериться, что входные данные удовлетворяют синтаксису разработанного языка. В случае успешной проверки, для входной строки строится синтаксическое дерево разбора, в противном случае выводится сообщение об ошибке с описанием вида ошибки, а также позиции лексемы во входной строке. Обнаружение и обработка ошибок (исключений) реализована с помощью операторов try–except.

Во входной строке могут содержаться лексические, синтаксические и семантические ошибки. Лексические и синтаксические ошибки определяются исходя из грамматики языка и связаны, в основном, с формой записи входных данных. Однако корректно введенные входные данные, не содержащие лексических и синтаксических ошибок, могут содержать семантические ошибки. Простейшим примером семантической ошибки является деление на ноль. В случае математических моделей среди семантических ошибок можно выделить следующие:

- несоответствие количества уравнений в системе количеству неизвестных параметров;
- параметр, не определяемый через уравнения, не задан численно – недостаточность исходных данных.

На рисунке 1 показан пример работы приложения для корректно введенной строки с выражением.

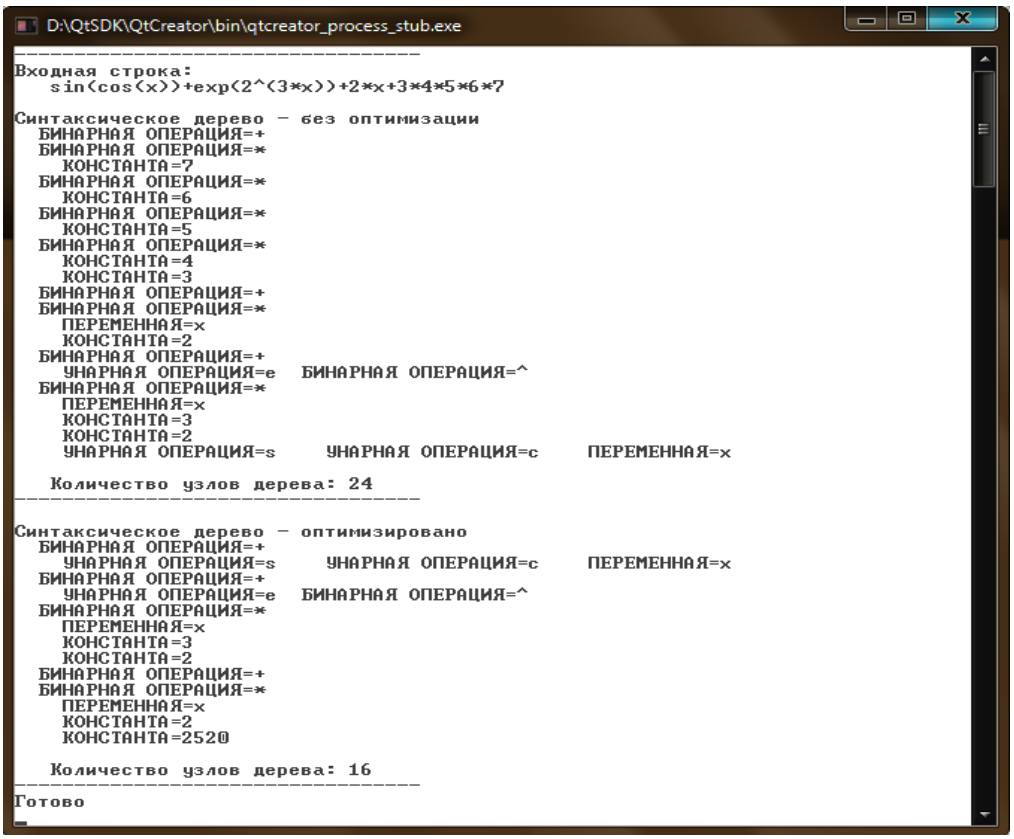

Рисунок 1. – Пример работы приложения для корректно введенной строки с выражением.

Выражение, содержащее несбалансированные скобки, воспринимается как не принадлежащее синтаксису языка. Для таких выражений, как видно из рисунка 2, возникает исключение и выводится соответствующее сообщение об ошибке. Синтаксическое дерево в этом случае не строится.

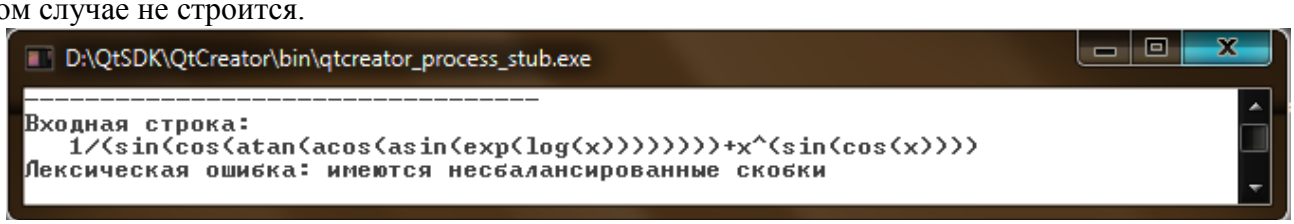

Рисунок 2. – Несбалансированные скобки.

Один из наиболее распространенных типов ошибки – отсутствие одного из операндов (или сразу двух) для бинарных операций (рисунок 3).

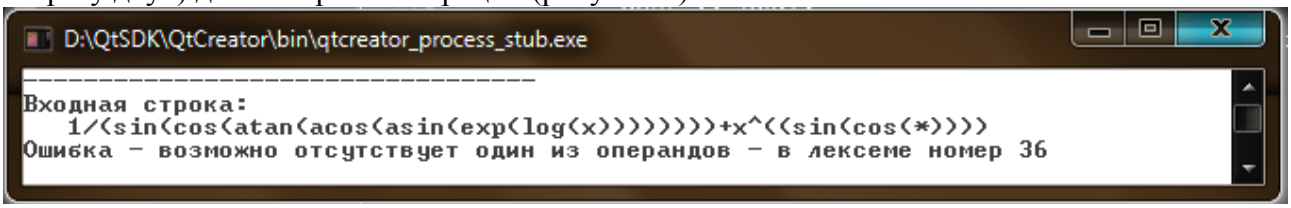

Рисунок 3. – Ошибка с указанием позиции лексемы.

Заключение. В результате выполнения данной работы был создан язык описания математических моделей, адаптированный для построения нисходящего синтаксического анализатора. Кроме того, разработано приложение для демонстрации работы синтаксического и лексического анализаторов.

Для достижения поставленных целей были решены следующие задачи:

- разработан формальный язык описания математических моделей;
- построены лексический и синтаксический анализатор;
- разработано приложение для демонстрации работы лексического и синтаксического анализаторов.

В качестве направления для дальнейшего развития можно выделить следующие задачи:

- модификация языка для работы с матричными и векторными величинами и уравнениями с соответствующими модификациями лексического и синтаксического анализаторов;
- разработка алгоритмов для генерации кода математических моделей;
- создание пользовательского интерфейса для ввода исходных данных и возможности проводить исследования математической модели.

Список литературы

1) Молдованова О.В. Языки программирования и методы трансляции: учеб. пособие. Новосибирск: СубГУТИ, 2012. 134 с.
2) Легалов А.И. Основы разработки трансляторов: учеб. пособие. Красноярск: Сибирский федер. ун-т, 2008.
3) Ахо А.В. Ульман Дж.Д., Агафонова В.Н. Теория синтаксического анализа, перевода и компиляции. В 2 т. Т.1. Синтаксический анализ. ред. В.М. Курочкина. М.: Мир, 1978. 613 с.
4) Ахо А.В. Сети Р., Ульман Дж.Д. Компиляторы: принципы, технологии и инструменты. под. ред. И.В. Красикова. 2-е изд. М.: Вильямс, 2008. 1184 с.