СПОСОБЫ ЗАДАНИЯ ИССКУСТВЕННЫХ ЯЗЫКОВ: СИНТАКСИС И СЕМАНТИКА ЯЗЫКА

Элова Дилрабо Кудратиллаевна
Независимый исследователь Университета узбекского языка и литературы имени Алишера Навои

https://doi.org/10.5281/zenodo.6975110

Аннотация. *В данной статье расскрываются способы задания исскуственных языков, обозначается синтаксис и семантика языка. Итак, грамматика – это описание способа построения предложений языка. Иными словами, грамматика – это математическая система, определяющая язык. Синтаксис языка – это набор правил, определяющий допустимые конструкции языка. Синтаксис определяет «форму языка» – задает набор цепочек символов, которые принадлежат языку. Поскольку языки программирования не являются чисто формальными языками и несут в себе некоторый смысл (семантику), то задача разбора для создания реальных компиляторов понимается несколько шире, чем она формулируется для чисто формальных языков.*

Ключевые слова: *цепочек языка, анализатор, лексический конструкция, синтаксис языка, лексема, грамматика, генератор цепочек языка, виды распознавателей, общая схема анализатора.*

WAYS TO DEFINE ARTIFICIAL LANGUAGES: SYNTAX AND SEMANTICS OF THE LANGUAGE

Abstract. *This article discusses the ways of defining artificial languages, indicates the syntax and semantics of the language. So, grammar is a description of a way of constructing sentences of a language. In other words, grammar is the mathematical system that defines language. Language syntax is a set of rules that define the allowed constructs of a language. Syntax defines the "form of the language" - defines the set of character strings that belong to the language. Since programming languages are not purely formal languages and carry some meaning (semantics), the problem of parsing for creating real compilers are understood somewhat broader than it is formulated for purely formal languages.*

Keywords: *language chains, parser, lexical construction, language syntax, lexeme, grammar, language chain generator, types of recognizers, general scheme of the analyzer.*

ВВЕДЕНИЕ

Каждый язык – это множество цепочек символов над некоторым алфавитом. Но кроме алфавита язык предусматривает также правила построения допустимых цепочек, поскольку обычно далеко не все цепочки над заданным алфавитом принадлежат языку. Символы могут объединяться в слова или лексемы – элементарные конструкции языка, на их основе строятся предложения – более сложные конструкции. И те и другие в общем виде являются цепочками символов, но предусматривают некоторые правила построения. Таким образом, необходимо указать эти правила, или, строго говоря, задать язык. В общем случае язык можно определить тремя способами [5].

1. Перечислением всех допустимых цепочек языка.
2. Указанием способа порождения цепочек языка (заданием грамматики языка).
3. Определением метода распознавания цепочек языка.

МАТЕРИАЛЫ И МЕТОДЫ

Особенность логического устройства (анализатор) – автомат, который на входе получает цепочку символов, а на выходе выдает ответ: принадлежит или нет эта цепочка заданному языку. Например, читая сейчас этот текст, вы в некотором роде выступаете в роли анализатора (надеюсь, что ответ на вопрос о принадлежности текста русскому языку будет положительным).

Говоря о любом языке, можно выделить его синтаксис и семантику. Кроме того, трансляторы имеют дело также с лексическими конструкциями (лексемами), которые задаются лексикой языка. Ниже даны определения всех этих понятий.

Синтаксис языка – это набор правил, определяющий допустимые конструкции языка. Синтаксис определяет «форму языка» – задает набор цепочек символов, которые принадлежат языку. Чаще всего синтаксис языка можно задать в виде строгого набора правил, но полностью это утверждение справедливо только для чисто формальных языков. Даже для большинства языков программирования набор заданных синтаксических конструкций нуждается в дополнительных пояснениях, а синтаксис языков естественного общения вполне соответствует общепринятому мнению о том, что «исключения только подтверждают правило» [5].

Известно, что в традиционной лингвистике лексика – это совокупность слов (словарный запас) языка. Слово или лексическая единица (лексема) языка – это конструкция, которая состоит из элементов алфавита языка и не содержит в себе других конструкций. Иначе говоря, лексическая единица может содержать только элементарные символы и не может содержать других лексических единиц. Например, лексическими единицами (лексемами) узбекского языка являются слова узбекского языка, а знаки препинания и пробелы представляют собой разделители, не образующие лексем. Лексическими единицами алгебры являются числа, знаки математических операций, обозначения функций и неизвестных величин. В языках программирования лексическими единицами являются ключевые слова, идентификаторы, константы, метки, знаки операций; в них также существуют и разделители (запятые, скобки, точки с запятой и т. д.).

Грамматика и анализатор. Грамматика – это описание способа построения предложений некоторого языка. Иными словами, грамматика – это математическая система, определяющая язык. Фактически, определив грамматику языка, мы указываем правила порождения цепочек символов, принадлежащих этому языку. Таким образом, грамматика – это генератор цепочек языка. Она относится ко второму способу определения языков – порождению цепочек символов. Грамматику языка можно описать различными способами. Например, грамматика русского языка описывается довольно сложным набором правил, которые изучают в начальной школе. Для некоторых языков (в том числе для синтаксических конструкций языков программирования) можно использовать формальное описание грамматики, построенное на основе системы правил (или продукций).

Анализаторы. Общая схема анализатора. Анализатор (или распознаватель) – это специальный автомат, который позволяет определить принадлежность цепочки символов некоторому языку. Задача распознавателя заключается в том, чтобы на основании исходной цепочки дать ответ на вопрос, принадлежит ли она заданному языку или нет.

Анализаторы, как было сказано выше, представляют собой один из способов определения языка.

Виды распознавателей. Анализаторы можно классифицировать в зависимости от вида составляющих их компонентов: считывающего устройства, устройства управления (УУ) и внешней памяти. По видам считывающего устройства анализаторы могут быть двусторонние и односторонние. Односторонние анализаторы допускают перемещение считывающей головки по ленте входных символов только в одном направлении. Поскольку все языки программирования подразумевают нотацию чтения исходной программы “слева направо”, то так же работают и все анализаторы. Двусторонние анализаторы допускают, что считывающая головка может перемещаться относительно ленты входных символов в обоих направлениях: как вперед, от начала ленты к концу, так и назад, возвращаясь к уже прочитанным символам [5].

По видам устройства управления анализаторы бывают *детерминированные* и *недетерминированные*.

По видам внешней памяти анализаторы бывают следующих типов:

1) анализаторы без внешней памяти;
2) анализаторы с ограниченной внешней памятью;
3) анализаторы с неограниченной внешней памятью. У распознавателей без внешней памяти внешняя память полностью отсутствует. В процессе их работы используется только конечная память.

У анализаторов с ограниченной внешней памятью размер внешней памяти ограничен в зависимости от длины входной цепочки символов. Эти ограничения могут предписываться некоторой зависимостью объема памяти от длины цепочки – линейной, полиномиальной, экспоненциальной и т.д. Анализаторы с неограниченной внешней памятью предполагают, что для их работы может потребоваться внешняя память неограниченного объема (вне зависимости от длины входной цепочки). У таких анализаторов предполагается память с произвольным методом доступа.

Вместе эти три составляющих позволяют организовать общую классификацию анализаторов. Например, в этой классификации возможен такой тип: *двусторонний недетерминированный анализатор с линейно ограниченной стековой памятью*.

Чем выше в классификации стоит анализатор, тем сложнее создавать алгоритм, обеспечивающий его работу. Разрабатывать двусторонние анализаторы сложнее, чем односторонние.

Задача разбора. Для каждого языка программирования важно не только уметь построить текст программы на этом языке, но и определить принадлежность имеющегося текста к данному языку. Именно эту задачу решают компиляторы в числе прочих задач (компилятор должен не только распознать исходную программу, но и построить эквивалентную ей результирующую программу). В отношении исходной программы компилятор выступает как анализатор, а человек, создавший программу на некотором языке программирования, выступает в роли генератора цепочек этого языка.

Грамматики и анализаторы – два независимых метода, которые реально могут быть использованы для *определения* какого-либо языка. Однако при создании компилятора для некоторого языка программирования возникает задача, которая требует связать между собой данные методы задания языков.

Разработчики компилятора всегда имеют дело с уже определенным языком программирования. Грамматика для синтаксических конструкций этого языка известна. Она, как правило, четко описана в стандарте языка. Задача разработчиков заключается в том, чтобы построить анализатор для заданного языка, который затем будет основой синтаксического анализатора в компиляторе.

Таким образом, задача разбора в общем виде заключается в следующем: на основе имеющейся грамматики некоторого языка построить анализатор для текущего языка. Заданная грамматика и анализатор должны быть эквивалентны, то есть определять один и тот же язык (часто допускается, чтобы они были почти эквивалентны, поскольку пустая цепочка во внимание обычно не принимается).

РЕЗУЛЬТАТЫ

Поскольку языки программирования не являются чисто формальными языками и несут в себе некоторый смысл (семантику), то задача разбора для создания реальных компиляторов понимается несколько шире, чем она формулируется для чисто формальных языков. Компилятор должен не просто установить принадлежность входной цепочки символов заданному языку, но и определить ее смысловую нагрузку. Для этого необходимо выявить те правила грамматики, на основании которых была построена цепочка. Если же входная цепочка символов не принадлежит заданному языку – исходная программа содержит ошибку, – разработчику программы не интересно просто узнать сам факт наличия ошибки. В данном случае задача разбора также расширяется: анализатор в составе компилятора должен не только установить факт присутствия ошибки во входной программе, но и по возможности определить тип ошибки и то место во входной цепочке символов, где она встречается.

Многократное использование корпуса большим количеством пользователей обусловило унифицированные и разметки, и программное обеспечение. Лингвистические и экстралингвистические разметки должны основываться на широко распространенных и общепринятых принципах описания текстов и языковых единиц. Параметры разметки, а также их значения должны соответствовать универсальным научным классификациям. Программное обеспечение, в свою очередь, должно заниматься обработкой типовых запросов и решением типовых задач. Огромное значение уделяется унификации форматов, как их наполнения, так и структуры. Обмен корпусными данными происходит на основе единых форматов представления данных, т. е. единого программного обеспечения. Работа пользователей с корпусом осуществляется с помощью специальных программных средств – корпусных менеджеров, которые предоставляют разнообразные пути получения из корпуса нужной информации:

«– поиск конкретных словоформ;

– поиск словоформ по леммам;

– поиск группы словоформ в виде разрывной или неразрывной синтагмы;

– поиск словоформ по набору морфологических признаков;

– отображение информации о происхождении, типе текста и т. п.;

– вывод результатов поиска с указанием контекста заданной длины;

– получение различных лексико-грамматических статистических данных;

– сохранение отобранных строк конкорданса в отдельном файле на компьютере пользователя и др.» [4].

ОБСУЖДЕНИЕ

Полученные результаты поиска представлены в виде конкорданса, где искомая единица представлена в ее контекстном окружении и в виде статистических данных. Они, фиксируя частотные характеристики отдельных языковых единиц, или граммем, характеризуют совместную встречаемость нескольких лексических единиц. Составители корпусов в работе сталкиваются с рядом проблем. Прежде всего следует отметить проблему репрезентативности корпусов, т.е. способности отражать все свойства проблемной области. Репрезентативность определяется фонетическими, морфологическими, синтаксическими, стилевыми параметрами. Создатель корпуса сначала ставит вопрос, какой корпус и для кого он создает. Невозможно представить компьютеру все тексты или все разговоры данного языка, поэтому создатели корпуса ориентируются на исследователей, которым этот корпус предназначен.

ВЫВОДЫ

Итак, способы задания исскуственных языков заключается в определении синтаксиса и семантики языка. Синтаксис языка – это набор правил, определяющий допустимые конструкции языка. Синтаксис определяет «форму языка» – задает набор цепочек символов, которые принадлежат языку. Слово или лексическая единица (лексема) языка – это конструкция, которая состоит из элементов алфавита языка и не содержит в себе других конструкций. Иначе говоря, лексическая единица может содержать только элементарные символы и не может содержать других лексических единиц.

После определения синтакисиса и семаники в процесс приступают грамматики и анализаторы – два независимых метода, которые реально могут быть использованы для определения какого-либо языка. Однако при создании компилятора для некоторого языка программирования возникает задача, которая требует связать между собой данные методы задания языков.

REFERENCES

1. Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции. В двух томах. пер. С англ. – М.: Мир, 1978.
2. Болтаев Т.Б., Ибрагимов С.И. О проекте программной системы морфологического анализа узбекского языка // “Тафаккур ва талқин” (Магистратура талабалари мақолалар тўплами). – Бухоро: Дурдона, 2018. – Б. 86-89. (http://ziyonet.uz/uploads/books/473012/5afbccb8e9f55.pdf (31.10.2020)
3. Болтаев Т.Б., Кузьминов Т.В., Поттосин И.В. О структурном конструирование программ и инструментах его поддержки // Среда программирования: методы и инструменты. – Новосибирск, 1992. – С.22-37.; The Structured Constructing as a Discipline of Safe Programming and Instruments Supporting It /Aniskov M.I., Boltaev T.B., Kochetov D.V. at al//Instrumental Congress on Computer Systems and Applied Mathematics CSAM’93. St-Petersburg. July 19-23.
4. Захаров В.П. Корпусная лингвистика. – СПб., 2005.
5. Место и назначение лингвистического обеспечения в информационных системах. Понятие информационной системы. Лекция 4 // https://gigabaza.ru/doc/116551-pall.html

6. Касьянов В.Н., Поттосин И.В. Методы построения трансляторов. – Новосибирск: Наука, 1986. – С. 344.;
7. Ножов И.М. Морфологическая и синтаксическая обработка текста (модели и программы): диссертация канд. филол. наук. − Москва, 2003. // https://docplayer.ru/26110069-I-m-nozhov-morfologicheskaya-i-sintaksicheskaya-obrabotka-teksta-modeli-i-programmy-1.html
8. Пономарев В.В. Лингвистическое обеспечение и социолингвистическая специфика проблемы автоиндексационной актуализации информационных систем: автореф. диссер. канд. филол. наук. − Москва, 2005.
9. Сысоев П.В. Интегративное обучение грамматике: исследование на материале английского языка // Иностранные языки в школе. 2003. № 6. С. 25–31.
10. Mohri M.A. Finite-state transducers in language and speech processing. Computational Linguistics, 23B), 269-312.
11. Speech and Langauge Processing. Daniel S. Jurafsky and James H. Martin. Contributing writers: Andrew Kehler, Keith Vander Linden, Nigel Ward 2000y. Prentice Hall, Englewood Cliffs, New Jersey 07632. pages: 950.
12. The Lexical Semantics of a Machine Translation Interlingua. RickMorneau // http://www.eskimo.com/~ram/lexical_semantics.html 2006.
13. Хамроева Ш.Ўзбек тили морфологик анализаторининг лингвистик таъминоти. филол. фан докт. диссертацияси автореф. – Фарғона, 2021.