СРАВНЕНИЕ LR И GLR СИНТАКСИЧЕСКИХ АНАЛИЗАТОРОВ

Юрченко Н.А., Будницкий Е.Г.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Савёнок В.А. – инженер-программист ООО «Незабудка Софтвер»,
магистр технических наук

В данной работе рассмотрены современные алгоритмы синтаксических анализаторов, которые применяются для разбора текстов формальных и естественных языков. Приведено подробное описание восходящих анализаторов: LR, SLR и GLR, сравнение их возможностей и ограничений, связанных с реализацией.

В настоящее время количество информации, представленной в электронном виде, растет с каждым днем. Все больше текстов с бумажных носителей переносится на электронные, для общения используются социальные платформы и электронная почта. Такой большой объем данных невозможно обработать за рациональное время не только одному человеку, но и группе специалистов, что приводит к потребности в автоматизации обработки и анализа текстовой информации.

Синтаксический анализатор – это программа или подпрограмма, производящая сопоставление линейной последовательности лексем формального или естественного языка его формальной грамматике [1]. Формальный язык - множество строк, из определенного алфавита, удовлетворяющее правилам. Для проверки корректности грамматики используются терминальные и нетерминальные символы. Терминал – элемент грамматики, для которого отсутствуют правила подстановки. Нетерминал – элемент грамматики, для которого есть правила подстановки. Синтаксический анализатор можно построить практически по любой грамматике. Результатом работы синтаксического анализатора является синтаксическое дерево, которое характеризует структуру выражения. На сегодняшний день существует ряд алгоритмов разбора предложений. Эти алгоритмы делятся на две группы в зависимости от способа построения дерева разбора: нисходящие и восходящие анализаторы. Нисходящий разбор строит дерево путем развёртки нетерминалов до соответствия с исходной строкой, восходящий - путем свертки терминалов в нетерминалы пока не останется один нетерминал. Методы восходящего разбора более предпочтительны, так как не требуется левофакторизованная грамматика. В данной работе рассматриваются анализаторы, входящие во вторую группу: семейства LR и GLR анализаторов.

Анализаторы LR детерминированы, они читают входной поток слева направо и преобразуют строку путем замены самого правого нетерминала. LR-анализаторы производят единственный правильный синтаксический анализ, в случае неправильного конечного ответа отката к предыдущим сверткам для выбора другого сворачивания не будет, так как на каждом шаге имеется только определенная схема свертки, в связи с этим алгоритм занимает линейное время. Существует множество вариантов LR-анализаторов, например LR(0), SLR(1), LR(1), LALR(1). Число в скобках показывает количество символов непрочитанной цепочки, используемых анализатором для принятия решения на каждом шаге. Однако при работе LR анализатора имеется проблема, что в некоторых ситуациях нельзя однозначно решить, какое действие следует предпринять в данных момент для достижения верного результата, так как имеется несколько вариантов сверток по различным правилам. При выборе ошибочного решения верная грамматика может оказаться ошибочной из-за неправильного свертывания. В связи с этим строятся таблицы LR разбора, в которых сохраняют только однозначное решение для данного и следующего символа.

LR(0) – восходящий алгоритм синтаксического разбора контекстно-свободных грамматик, один из видов LR. Данный алгоритм редко используется в связи с малым размером грамматик, которые он может разобрать, но он является основой для более эффективных алгоритмов разбора, таких как SLR(1) и LALR(1). Данный алгоритм не требует предпросмотр, что уменьшает время на проверку его возможных грамматик.

SLR(1) – улучшение восходящего алгоритма синтаксического разбора LR(0), которое позволяет разбирать более сложные грамматики. Однако сложность разбираемых им грамматик недостаточно высока, что ограничивает область использовать данного алгоритма только в учебных целях.

LR(1) – восходящий алгоритм синтаксического разбора, для принятия решения которого требуется прочтение одного символа. Главное отличие от LR(0) алгоритма заключается в наличии предпросмотра. Основная идея данного алгоритма заключается в том, чтобы не производить некорректные свертки. Алгоритм затрачивает больше оперативной памяти, но из-за увеличения возможной грамматики его возможностей хватает на различные языки, к примеру C, C++ [2].

LALR(1) (LA от англ. lookahead – предпросмотр) представляет собой восходящий алгоритм синтаксического разбора; является расширением SLR(1). Он позволяет сохранить наиболее подходящие ветви разбора для устранения неоднозначности. Несмотря на то, что данный алгоритм менее мощный, чем LR(1), его возможностей вполне достаточно для разбора большинства языков, к

примеру C, C++ [2].

GLR-анализатор (англ. Generalized Left-to-right Rightmost derivation parser) – обобщенный восходящий магазинный анализатор или параллельный синтаксический анализатор. Этот расширенный алгоритм предназначен для разбора по недетерминированным и неоднозначным грамматикам. Теоретическая база была заложена Бернардом Лэнгом [3] впервые такой синтаксический анализатор описал Масару Томита [4]. Цель создания GLR-анализатора – анализ текстов на естественных языках, с недетерминированностью и неоднозначностью которых LR-анализатор не справляется.

Для разбора недетерминированных грамматик GLR-анализаторы обрабатывают различные ветви разбора параллельно, они используют более сложную стековую структуру, нежели LR-анализаторы. Такая структура называется графовой структурой (GSS – англ. graphstructured stack). Использование графовой структуры позволяет эффективно хранить различные ветви разбора, которые помечаются неактивными по мере сдвига входных токенов или символов. Все неактивные ветви должны быть восстановлены в случае повторного посещения старого состояния, из чего следует задача применения поиска с возвратом, что усложняет работу анализатора.

Работу синтаксического анализатора можно оптимизировать сокращением объема хранимой информации. GLR-анализатор считается самым универсальным из рассмотренных алгоритмов, поскольку он разбирает не только однозначные грамматики, но также способен успешно анализировать естественные языки. Однако у классического GLR-анализатора есть ряд недостатков. В частности, он не имеет средств выхода из скрытой левой рекурсии [5], которая часто присутствует в грамматиках языков программирования. На практике часто используются гибридные анализаторы, в которых GLR комбинируют с алгоритмом LALR(1), что обеспечивает более высокую производительность.

Сравнительный анализ описанных восходящих анализаторов представлен в таблице 1. Для анализа выбраны следующие характеристики: глубина предпросмотра, уровень сложности грамматики, размер таблицы, способность анализировать естественные языки.

Таблица 1 – Сравнительная характеристика синтаксических анализаторов

Тип анализатора	Параметры сравнения			
	Глубина предпросмотра	Уровень сложности грамматики	Размер таблицы	Подходит для анализа естественных языков
LR(0)	0	Низкий	Маленькая	Нет
SLR(1)	1	Низкий	Больше, чем у LR(0)	Нет
LALR(1)	1	Средний	Такая же, как у SLR(0)	Нет
LR(1)	1	Средний	Большая	Нет
GLR	Не ограничена	Высокий	Большая	Да

Развитие анализаторов происходило в порядке: LR(0) → SLR(1) → LALR(1) → LR(1) → GLR. Таким образом, LR(0) поддерживает грамматику только LR(0), SLR(1) поддерживает LR(0) и SLR(1), LALR(1) поддерживает LR(0), SLR(1) и LALR(1) и т. д. Для разбора строятся LR-таблицы, размер которых увешивается с ростом сложности разбираемой грамматики как показано в таблице 1. Размер LR-таблицы также связан с количеством всевозможных состояний, а состояния в свою очередь связаны со сложностью грамматики. Поэтому чем проще грамматика, тем быстрее ее будет разобрать. Использование предпросмотра увеличивает затрачиваемое время на анализ, но не во всех случаях избежать предпросмотр возможно. Из-за простоты некоторых грамматик, LR(0), SLR(1), необходимость предпросмотра отпадает, что уменьшает время на их анализ.

Анализаторы семейства LR подходят для разбора формальных языков, но не пригодны для работы с естественными языками, потому что не способны разрешать недетерминированность и неоднозначность этих языков. Для анализа естественных языков используют семейство GLR-анализаторов.

Таким образом по результатам анализа нельзя выделить лучший или худший синтаксический анализатор. Все анализаторы имеют свои плюсы и минусы. Каждый анализатор разрабатывался для разбора определенных грамматик. При выборе анализатора следует исходить из задачи, предоставленной для решения, и искать анализатор, который справится с этой задачей максимально эффективно.

Список использованных источников:

1. Бионика интеллекта / Борисова Н.В., Канищева О.В. – Изд. 2-е. – 2014
2. Bison GNU / Operating system GNU. – Mode of access: https://www.gnu.org/software/bison/
3. Deterministic techniques for efficient non–deterministic parsers / Lang B. Lang B. – 2nd Colloquium – Saarbrücken: Springer.
4. Efficient parsing for natural language / Tomita M. – Boston : Kluwer Academic Publishers, 1986.
5. Generalized parsing: some costs / Johnstone A., Scott E., Economopoulos G.. – Berlin: Springer, 2004.