УДК 519.687.7
https://doi.org/10.36906/AP-2020/30

ИСТОРИЯ РАЗВИТИЯ ТЕХНОЛОГИИ ПАРСИНГА

Королев Р. И.
Нижневартовский государственный университет
г. Нижневартовск, Россия
Никонова Е. З.
канд. пед. наук
Нижневартовский государственный университет
г. Нижневартовск, Россия

Аннотация. В статье рассматривается история возникновения и развития программ-парсеров, использующих методы синтаксического анализа для автоматизированного синтаксического и лексического анализа текстовых документов (парсинга). Автор отмечает события и авторов, оказавших наиболее значимое влияние на развитие данной технологии.

Ключевые слова: синтаксический анализ информации, анализаторы, синтаксически управляемые алгоритмы, рекурсивный спуск, парсинг, парсеры, алгоритм Эрли.

В настоящее время основным источником информации является интернет, предоставляющий огромное количество ссылок на различные сайты, содержащие текстовые документы. В связи с этим основной трудностью для пользователя становится отбор нужной информации, предполагающий ее первичный анализ. С этой задачей помогают справиться специальные программы — парсеры, использующие методы синтаксического анализа для автоматизированного синтаксического и лексического анализа текстовых документов (парсинга).

К программам, выполняющим задачи анализа текста, относится еще множество различных программ, таких как программы–переводчики, трансляторы языков программирования, транслятор запросов с языка реляционной алгебры в SQLи т.п.

Под методом синтаксического анализа в информатике понимают сопоставление линейной последовательности слов некоторого формального языка с правилами его грамматики.

История автоматического анализа текста, представленного в виде последовательности операторов языка программирования, началась в 1960 г запуском спецификации языка ALGOL 60, в которой был определен язык со структурой блоков. В то время не существовало такого понятия как парсинг, но разработчики были убеждены, что подробная характеристика языка с блочным строением позволит в дальнейшем с легкостью осуществлять анализ программ. И как показало время, их предположения вполне оправдались.

В 1961 г Нед Айронс описал свой анализатор ALGOL, отличающийся следующими особенностями:

–анализ осуществлялся по алгоритму рекурсивного спуска в виде леворекурсивного (рекурсивный спуск — это алгоритм, по которому синтаксический анализ выполняется с помощью взаимного вызова процедур, каждая из которых соответствует определенному правилу грамматики);

–алгоритм носил общий характер, т.е. был способен разобрать синтаксис, представленный в виде формы Бэкуса — Наура (БНФ), в которой синтаксические категории последовательно определяются через последующие);

–анализатор являлся синтаксически-управляемым, т.е. автоматически создающимся из БНФ.

В этом же году были реализованы леворекурсивные алгоритмы с возможностью изменения исходного кода, так называемые «самокодируемые» анализаторы, которые в настоящее время объединяются общим названием «рекурсивный спуск».

Именно эти подходы в построении анализаторов постепенно вытеснили синтаксически управляемые алгоритмы, и причиной этому послужили следующие факторы:

–ограниченность таких ресурсов как память и CPU, вследствие чего «ручное» кодирование алгоритмов было более продуктивным;

–слабые возможности «чистого» леворекурсивного алгоритма в синтаксическом анализе и необходимость использовать «ручное» кодирование.

В 1965 г Дональд Кнут описал алгоритм анализа, названный им LR. Данный подход имел большую ценность с точки зрения математической задачи, чем в его практическом использовании.

В 1968 г Джей Эрли разработал алгоритм, названный в его честь «Алгоритм Эрли». Подобно алгоритму Неда Айронса он был синтаксически-управляемым и имел общий характер, но не использовал метод поиска с возвратом. Ключевая идея Эрли заключалась в использовании таблиц для отслеживания выполнения алгоритма.

Но и этот подход не был лишен некоторых недостатков:

–правила нулевой длины обрабатывались некорректно;

–использование правосторонней рекурсии требовало повторного прохождения алгоритма;

–создание и ведение таблиц должно было вестись с учетом имеющихся системных ресурсов, что в то время было достаточно сложной задачей.

В 1969 г Фрэнк де Ремер модифицировал алгоритм LR Дональда Кнута в алгоритм LALR, для работы которого необходим небольшой объем памяти для размещения стека и таблицы состояний.

В 1972 г Алманн и Аха добились корректной обработки правил нулевой длины в алгоритме Эрли, но это потребовало значительного увеличения системных ресурсов.

В 1975 г Bell Labs переписала свой компилятор языка Си, построенный методом рекурсивного спуска с самокодированием, в компилятор по алгоритму LALR ДеРемера.

В 1977 г была издана «Книга дракона» — первая книга по синтаксическому анализу, получившая свое название из-за изображения рыцаря на обложке, на копье которого можно было прочесть "LALR".

В 1979 г в лаборатории Bell Labs вышла седьмая версия операционной системы UNIX, в состав которой входил инструментарий для создания компиляторов. И основным среди них является Yacc — генератор синтаксических анализаторов, основанный на алгоритме LALR. Особенностью Yacc была его способность анализировать свой собственный язык ввода, а также язык компилятора Си, что позволило считать задачу синтаксического анализа успешно решенной.

В 1987 г появляется Perl — высокоуровневый интерпретируемый динамический язык программирования общего назначения, созданный Ларри Уоллом и первоначально предназначенный для работы с текстом. Новый язык упростил труд разработчиков, которые

смогли посвятить себя решению более трудные задачи. Существовавшие в то время языки программирования не предоставляли таких возможностей.

В 1991 г Лео Джуп предложил способ ускорения правосторонних рекурсий в алгоритме Эрли, позволяющий применить его для любого вида грамматики, как однозначной, так и неоднозначной, т.е. способного предложить реализацию некоторой строки более чем одним способом. Но, несмотря на то, что требования к системным ресурсам уже не являлись столь критичными, новый алгоритм по своей скорости работы уступал классическому алгоритму Эрли, и поэтому его настоящая реализация смогла осуществиться только 20 лет спустя.

Таким образом, признанным стандартом в разработке анализаторов остался алгоритм LALR. Но все чаще стали выявляться его достаточно существенные «неудобства», одно из которых заключалось в том, что этот алгоритм мог автоматически генерировать коды, но их отладка являлась настолько трудозатратной, что сводила на нет все преимущества автоматической генерации.

Другим недостатком можно считать неполное сообщение пользователю об ошибке, включающее лишь информацию о неверном формате без каких-либо уточнений и пояснений.

И даже достаточно большая скорость работы при корректных входных данных не могла компенсировать указанных неудобств, что дало повод Ларри Уоллу назвать LALR «быстрым, но глупым».

Не случайно в новом издании своего Perl 6 в 2000 году Ларри Уолл полностью отказался от использования LALR.

В 2002 г Хоспул и Эйкок обнародовали итоги собственных исследований в модификации алгоритма Эрли, которая, в частности, как и в разработках Лео Джуп, позволяла устранить ошибки при обработке правил нулевой длины. Следует отметить, что в отличие от предшественника, их метод не требовал учета системных ресурсов, но был достаточно сложным.

В 2006 г проект GNU — объединение разработчиков, создающих бесплатные программы, объявляет о своей разработке компилятора GCC, являющегося, по сути, набором компиляторов для различных языков программирования и использующего метод рекурсивного спуска.

Произошел своеобразный поворот в истории развития синтаксического анализа: алгоритм рекурсивного спуска, вытесненный в 70-ые годы 20-го века алгоритмом LALR, снова возвращается. И это несколько снижает значимость достижений в применении синтаксического анализа, поскольку «перечеркивает» все предыдущие достижения и возвращает к истокам — алгоритму Неда Айрона 1961 года.

Технологии синтаксичесого анализа постоянно совершенствуются, и их практическое применение в виде парсинга становится одним из востребованных инструментов бизнеса, которым успешно пользуются и многие российские компании (18% компаний уже используют в своей работе парсинг, а 23% предполагают использование этого инструмента в ближайшем будущем).

Очень часто парсинг используется в бизнесе, так, например, существуют сайты, которые находят самые дешевые авиа или ж.д. билеты среди большого количества предложений на сайтах транспортных компаний (Aviasales.ru, tutu.ru и другие)

Умная лента новостей Google и Yandex, которая подбирает новости из ссылок на разные, но предпочтительные для пользователя, новостные порталы.

Таким образом, парсинг может стать мощным инструментом решения тактических и стратегических задач предприятия при условии соблюдения соответствующих юридических норм.

Литература

1. Альфред В. Ахо, Моника С. Лам, Рави Сети, Джеффри Д. Ульман. Компиляторы: принципы, технологии и инструментарий, М.: Вильямс. 2001.

2. Хантер Р. Основные концепции компиляторов, М.: Вильямс, 1986.

3. Jeffrey D. Ullman, Monica S. Lam, Ravi Seti, Alfred W. Aho. Compilers: Principles, Techniques, and Tools. Addison-Wesley, 2008.

4. Hanter R. The Essence of Compilers. M.: Williams, 2002, P. 258.