Лабораторная работа №1

Распознавание цепочек регулярного языка

1. Цель и задачи работы

Цель работы: приобретение практических навыков реализации важнейших элементов лексических анализаторов на примере распознавания цепочек регулярного языка.

Задачи работы:

1) Ознакомиться с основными понятиями и определениями, лежащими в основе построения лексических анализаторов.
2) Прояснить связь между регулярным множеством, регулярным выражением, праволинейным языком, конечно-автоматным языком и недетерминированным конечно-автоматным языком.
3) Разработать, тестировать и отладить программу распознавания цепочек регулярного или праволинейного языка в соответствии с предложенным вариантом грамматики.

2. Материал для изучения и ознакомления

Перед выполнением работы рекомендуется ознакомиться со следующими материалами:

Регулярный язык. URL: https://ru.wikipedia.org/wiki/Регулярный_язык
Регулярная грамматика. URL: https://ru.wikipedia.org/wiki/Регулярная_грамматика
Праволинейная грамматика. URL: https://ru.wikipedia.org/wiki/Праволинейная_грамматика
Конечный автомат. URL: https://ru.wikipedia.org/wiki/Конечный_автомат
Минимальная форма автомата. URL: https://ru.wikipedia.org/wiki/Минимальная_форма_автомата
Regular language. URL: https://en.wikipedia.org/wiki/Regular_language
Regular expression. URL: https://en.wikipedia.org/wiki/Regular_expression
Finite-state machine. URL: https://en.wikipedia.org/wiki/Finite-state_machine

3. Теоретическая часть

Распознавание цепочек регулярного языка является центральной задачей лексического анализа, который образует первый этап процесса компиляции. На этом этапе символы, составляющие исходную программу, считываются и группируются в отдельные лексические элементы, называемые лексемами.

Большую часть того, что происходит в течение лексического анализа можно моделировать с помощью конечных преобразователей, работающих последовательно или параллельно. Например, лексический анализатор может состоять из ряда последовательно соединенных конечных преобразователей. Первый преобразователь в этой цепи может устранять из исходной программы все несущественные пробелы, второй ликвидирует комментарии, третий ищет константы и т. д. Другая возможность - завести набор конечных преобразователей, каждый из которых ищет определенную лексическую конструкцию.

Лексические анализаторы бывают по существу двух видов - прямые и непрямые. Можно показать как по регулярным выражениям, описывающим соответствующие лексемы, строятся анализаторы обоих видов.

При непрямом лексическом анализе требуется, прочитав цепочку знаков, определить, появилась ли подцепочка, образующая некоторую конкретную лексему. Если множество возможных цепочек, которые могут образовывать эту лексему, обозначается, как это обычно бывает, регулярным выражением, то проблему построения непрямого лексического анализатора для данной лексемы можно представить себе как проблему реализации конечного преобразователя. Конечный преобразователь - это почти конечный автомат (распознаватель) в том смысле, что он

читает вход, не производя выхода, пока не обнаружит присутствие лексемы данного типа (т. е. достигнет заключительного состояния). Тогда он сигнализирует о том, что эта лексема появилась, и выдает на выходе цепочку символов, образующих эту лексему.

Проблема непрямого лексического анализа является, таким образом, по существу проблемой построения детерминированного конечного автомата (ДКА) по заданному регулярному выражению и его программной реализации. В простейшем случае сначала по регулярному выражению строят недетерминированный конечный автомат (НКА). Затем этот НКА превращают в ДКА, либо моделируют его работу, прослеживая параллельно всевозможные последовательности тактов.

Пусть множество лексем данного типа обозначается регулярным выражением или задается в виде праволинейной грамматики. Во втором случае требуется преобразование праволинейной грамматики в регулярное выражение, которое выполняется за два шага:
Шаг 1. Построение стандартной системы уравнений с регулярными коэффициентами по праволинейной грамматике.
Шаг 2. Решение стандартной системы уравнений с регулярными коэффициентами.

Построение НКА по регулярному выражению может выполняться либо с помощью алгоритма «Конструктор Томпсона», либо с помощью алгоритма построения НКА по расширенному регулярному выражению (см. Алгоритм 3.2. АУ1).

Наконец, осуществляют детерминированное моделирование НКА для заданной входной цепочки знаков. При моделировании конечного автомата необходимо учитывать следующие обстоятельства. Работа автомата завершается, если обнаруживается хотя бы одна допускающая конфигурация, достижимая из начальной конфигурации автомата. Если входная цепочка построена синтаксически неправильно, то придется рассмотреть все возможные последовательности тактов автомата. Если исчерпаны все возможные последовательности тактов, а допустимая конфигурация не обнаружена, то надо выдать сообщение об ошибке. **Внимание! При моделировании НКА или ДКА необходимо показать все конфигурации алгоритма.**

4. Практическая часть

Для решения задач лабораторной работы необходимо обратиться к следующим алгоритмам:

1) Построение стандартной системы уравнений с регулярными коэффициентами по праволинейной грамматике: Теорема 2.2. [2], Пример 2.10. [2].
2) Решение стандартной системы уравнений с регулярными коэффициентами: Алгоритм 2.1. [2], Пример 2.9. [2].
3) Построение НКА по регулярному выражению: 7.5.2. [1], Алгоритм 3.23. [3], Пример 3.24. [3], From regular expressions to NFA [4].
4) Thompson's construction. URL: https://en.wikipedia.org/wiki/Thompson%27s_construction
5) Моделирование НКА: Алгоритм 3.22. [3].
6) Преобразование НКА в ДКА: 7.6. [1], Алгоритм 3.20. [3], Пример 3.21. [3], From NFA to FA via subset construction [4].
7) Построение по НКА эквивалентного ДКА, алгоритм Томпсона. URL: http://neerc.ifmo.ru/wiki/index.php?title=Построение_по_НКА_эквивалентного_ДКА,_алгоритм_Томпсона
8) Моделирование ДКА: Алгоритм 3.18. [3], Пример 3.19. [3].
9) Минимизация количества состояний ДКА: 7.7. [1], Алгоритм 2.2. [2], Пример 2.15. [2], Алгоритм 2.6. [2], Пример 3.38. [3], Алгоритм 3.39. [3].
10) Минимизация ДКА, алгоритм за O(n^2) с построением пар различимых состояний. URL: http://neerc.ifmo.ru/wiki/index.php?title=Минимизация_ДКА,_алгоритм_за_O(n%5E2)_с_построением_пар_различимых_состояний
11) Минимизация ДКА, алгоритм Хопкрофта (сложность O(nlogn)). URL: http://neerc.ifmo.ru/wiki/index.php?title=Минимизация_ДКА,_алгоритм_Хопкрофта_(сложность_O(n_log_n))
12) Алгоритм Бржозовского. URL: http://neerc.ifmo.ru/wiki/index.php?title=Алгоритм_Бржозовского
13) Построение ДКА по регулярному выражению: Алгоритм 3.36. [3], Пример 3.27. [3].

5. Варианты заданий на лабораторную работу

Вариант 1

Напишите программу, которая в качестве входа принимает произвольную праволинейную грамматику, и выполняет следующие преобразования:

1) По праволинейной грамматике строит стандартную систему уравнений с регулярными коэффициентами, неизвестными которой служат нетерминалы исходной грамматики.
2) Решает стандартную систему уравнений с регулярными коэффициентами.
3) По регулярному выражению, являющемуся решением стандартной системой уравнений с регулярными коэффициентами, строит НКА.
4) Детерминировано моделирует НКА.

Вариант 2

Напишите программу, которая в качестве входа принимает произвольное регулярное выражение, и выполняет следующие преобразования:

1) По регулярному выражению строит НКА.
2) По НКА строит эквивалентный ему ДКА.
3) По ДКА строит эквивалентный ему КА, имеющий наименьшее возможное количество состояний.
   Указание. Воспользоваться алгоритмом, приведенным по адресу
   http://neerc.ifmo.ru/wiki/index.php?title=Минимизация_ДКА,_алгоритм_за_O(n%5E2)_с_построением_пар_различимых_состояний
4) Моделирует минимальный КА для входной цепочки из терминалов исходной грамматики.

Вариант 3

Напишите программу, которая в качестве входа принимает произвольное регулярное выражение, и выполняет следующие преобразования:

1) По регулярному выражению строит НКА.
2) По НКА строит эквивалентный ему ДКА.
3) По ДКА строит эквивалентный ему КА, имеющий наименьшее возможное количество состояний.
   Указание. Воспользоваться алгоритмом, приведенным по адресу
   http://neerc.ifmo.ru/wiki/index.php?title=Минимизация_ДКА,_алгоритм_Хопкрофта_(сложность_O(n_log_n))
4) Моделирует минимальный КА для входной цепочки из терминалов исходной грамматики.

Вариант 4

Напишите программу, которая в качестве входа принимает произвольное регулярное выражение, и выполняет следующие преобразования:

1) По регулярному выражению строит НКА.
2) По НКА строит эквивалентный ему ДКА.
3) По ДКА строит эквивалентный ему КА, имеющий наименьшее возможное количество состояний.
   Указание. Воспользоваться алгоритмом, приведенным по адресу
   http://neerc.ifmo.ru/wiki/index.php?title=Алгоритм_Бржозовского
4) Моделирует минимальный КА для входной цепочки из терминалов исходной грамматики.

Вариант 5

Напишите программу, которая в качестве входа принимает произвольное регулярное выражение, и выполняет следующие преобразования:

1) Преобразует регулярное выражение непосредственно в ДКА.
2) По ДКА строит эквивалентный ему КА, имеющий наименьшее возможное количество состояний.
   Указание. Воспользоваться алгоритмом, приведенным по адресу
   http://neerc.ifmo.ru/wiki/index.php?title=Минимизация_ДКА,_алгоритм_за_O(n%5E2)_с_построением_пар_различимых_состояний

3) Моделирует минимальный КА для входной цепочки из терминалов исходной грамматики.

Вариант 6

Напишите программу, которая в качестве входа принимает произвольное регулярное выражение, и выполняет следующие преобразования:

1) Преобразует регулярное выражение непосредственно в ДКА.
2) По ДКА строит эквивалентный ему КА, имеющий наименьшее возможное количество состояний. *Указание.* Воспользоваться алгоритмом, приведенным по адресу http://neerc.ifmo.ru/wiki/index.php?title=Минимизация_ДКА,_алгоритм_Хопкрофта_(сложность_O(n_log_n))
3) Моделирует минимальный КА для входной цепочки из терминалов исходной грамматики.

Вариант 7

Напишите программу, которая в качестве входа принимает произвольное регулярное выражение, и выполняет следующие преобразования:

1) Преобразует регулярное выражение непосредственно в ДКА.
2) По ДКА строит эквивалентный ему КА, имеющий наименьшее возможное количество состояний. *Указание.* Воспользоваться алгоритмом, приведенным по адресу http://neerc.ifmo.ru/wiki/index.php?title=Алгоритм_Бржозовского
3) Моделирует минимальный КА для входной цепочки из терминалов исходной грамматики.

6. Порядок выполнения работы

1) Ознакомиться с основными понятиями и определениями по рекомендуемым в п. 2 материалам.
2) Проработать главу 2 учебного пособия [5].
3) Изучить алгоритмы, рекомендованные в п. 4 для предложенного варианта задания на лабораторную работу.
4) Разработать, тестировать и отладить программу распознавание цепочек регулярного языка для предложенного варианта задания на лабораторную работу.
5) Подготовить отчет о проделанной работе.
6) Подготовить ответы на контрольные вопросы.

7. Требования к отчету

Отчет по лабораторной работе выполняется в электронном виде и должен включать:

1) Идентификатор группы, имя и фамилию студента, дату выполнения работы.
2) Название лабораторной работы.
3) Описание задания – постановку задач, подлежащих выполнению в процессе лабораторной работы.
4) Текст программы, в которой решаются поставленные задачи.
5) Набор тестов и ожидаемые результаты для проверки правильности программы.
6) Результаты выполнения программы.
7) Анализ результатов и краткие выводы по работе.
8) Список дополнительной использованной литературы или дополнительных использованных электронных ресурсов.

8. Контрольные вопросы

1) Какие из следующих множеств регулярны? Для тех, которые регулярны, напишите регулярные выражения.
    a. Множество цепочек с равным числом нулей и единиц.
    b. Множество цепочек из {0, 1}* с четным числом нулей и нечетным числом единиц.
    c. Множество цепочек из {0, 1}*, длины которых делятся на 3.
    d. Множество цепочек из {0, 1}*, не содержащих подцепочки 101.

2) Найдите праволинейные грамматики для тех множеств из вопроса 1, которые регулярны.
3) Найдите детерминированные и недетерминированные конечные автоматы для тех множеств из вопроса 1, которые регулярны.
4) Найдите конечный автомат с минимальным числом состояний для языка, определяемого автоматом M = ({A, B, C, D, E}, {0, 1}, d, A, {E, F}), где функция в задается таблицей

Состояние	Вход	
	0	1
A	B	C
B	E	F
C	A	A
D	F	E
E	D	F
F	D	E

9. Рекомендуемая литература

1. БЕЛОУСОВ А.И., ТКАЧЕВ С.Б. Дискретная математика: Учеб. Для вузов / Под ред. В.С. Зарубина, А.П. Крищенко. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2001.
2. АХО А., УЛЬМАН Дж. Теория синтаксического анализа, перевода и компиляции: В 2-х томах. Т.1.: Синтаксичечкий анализ. - М.: Мир, 1978.
3. АХО А.В, ЛАМ М.С., СЕТИ Р., УЛЬМАН Дж.Д. Компиляторы: принципы, технологии и инструменты. – М.: Вильямс, 2008.
4. Notes on lexical analysis.pdf.
5. БУНИНА Е.И., ГОЛУБКОВ А.Ю. Формальные языки и грамматики. Учебное пособие. – М.: Изд-во МГТУ им. Н.Э.Баумана, Москва, 2006. URL: http://iu9.bmstu.ru/data/book/fl.pdf