Домашнее задание № 4, Формальные языки

Шестаков Денис Владиславович

28.09.2020

1. Немного изменил лексер, теперь он возвращает токены: KW - для ключевых слов terminal, start, points; VER и $WORD$ для вершин и слов соответственно; по токену для каждого оператора ; , $->$: ".
   Для своего языка я создал грамматику со следующими правилами:

$$\begin{cases} discr = discr\ str; \mid str; \\ str = KW\ ver_l \mid ver_{from} -> ver_{to} : "word_l" \\ ver_l = ver_l, VER \mid VER \\ ver_{from} = VER \\ ver_{to} = VER \\ word_l = word_l,\ WORD \mid WORD \end{cases}$$

   Здесь $discr$ - описание всего автомата, str - строка этого описания. $ver_l, word_l$ - это список из вершин и слов через запятую. Впоследствии, чтобы корректно и достаточно быстро обрабатывать ошибки, я решил создать еще два класса ver_{from}, ver_{to} - откуда и куда соответственно ведет ребро. Проблема была в том, что если просто писать VER, то выводится сразу вся строка вместе со словами, т.е. до точки с запятой. Сначала я решил это просто как $ver_l -> ver_l$, но потом оказалось, что такое решение будет работать либо некорректно, либо дольше (линия вместо логарифма для обработки ошибок).
   Парсер строит дерево по автомату, а потом, получая строку проходит по нему и проверяет, что оказался в терминальной вершине. Все автоматы находятся в папке $Automata$ - это автоматы из $HW02$. Тесты в папке $Tests$. $test_X.Y$ - тест для автомата X, он принимает строку если $Y == 1$, и не принимает иначе.

2. Обрабатываю ошибки, которые указаны в задании, тесты для них находятся в папке $Errors$ - в случае ошибки дерево не строится, выводит ошибку и завершает работу. $ErrX, 1 \leq Xleq5$ - обработка ошибок из задания. Я также добавил обработку стандартных ошибок на количество аргументов и возможность открыть файл. $Err6$ - проверка, что стартовая вершина всего одна, $Err7$ - проверка, что автомат полный, но я удалил вообще все ребра.