ГЕНЕРАЦИЯ ТЕСТОВ ДЛЯ СИНТАКСИЧЕСКОГО АНАЛИЗА МЕТОДАМИ СУПЕРКОМПИЛЯЦИИ

Студент: Головань С. М.

Руководитель ВКР: Коновалов А. В.

МГТУ им. Н.Э. Баумана

2018

Постановка задачи

Пусть имеется некоторая LL(1)-грамматика, распознаваемая синтаксическим анализатором - автоматом с магазинной памятью.

Необходимо построить конечный набор тестов, покрывающих всю логику работы синтаксического анализатора для заданной LL(1)-грамматики.

Методы тестирования парсеров | 3

Позитивные тесты – предложения языка

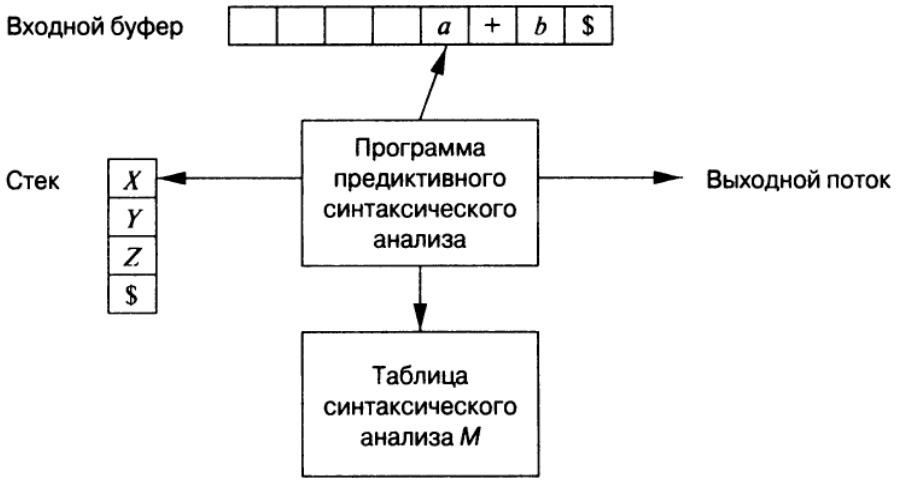
- Стохастические алгоритмы
- Критерий Пардома
- Критерий покрытия всех пар

Негативные тесты – содержат одну синтаксическую ошибку

- Метод мутации позитивных тестов
- Метод мутации грамматики

Распознавание LL(1)-грамматик

$$
\begin{aligned}
E &::= TE' ; \\
E' &::= '+' \ TE' \mid \varepsilon ; \\
\\
T &::= FT' ; \\
T' &::= '*' \ FT' \mid \varepsilon ; \\
F &::= n \mid '(' \ E \ ')' ;
\end{aligned}
$$



	+	*	n	(	)	$
E	$+E$	$*E$	$\underline{\boldsymbol{T\,E'}}$	$\underline{\boldsymbol{T\,E'}}$	ε	ε
E'	$\underline{\boldsymbol{+\,T\,E'}}$	$*E'$	nE'	$(E'$	$\underline{\boldsymbol{\varepsilon}}$	$\underline{\boldsymbol{\varepsilon}}$
T	ε	$*T$	$\underline{\boldsymbol{F\,T'}}$	$\underline{\boldsymbol{F\,T'}}$	ε	ε
T'	$\underline{\boldsymbol{\varepsilon}}$	$\underline{\boldsymbol{*\,F\,T'}}$	nT'	$(T'$	$\underline{\boldsymbol{\varepsilon}}$	$\underline{\boldsymbol{\varepsilon}}$
F	ε	ε	$\underline{\boldsymbol{n}}$	$\underline{(\,\boldsymbol{E}\,)}$	ε	ε

Граф состояний анализатора

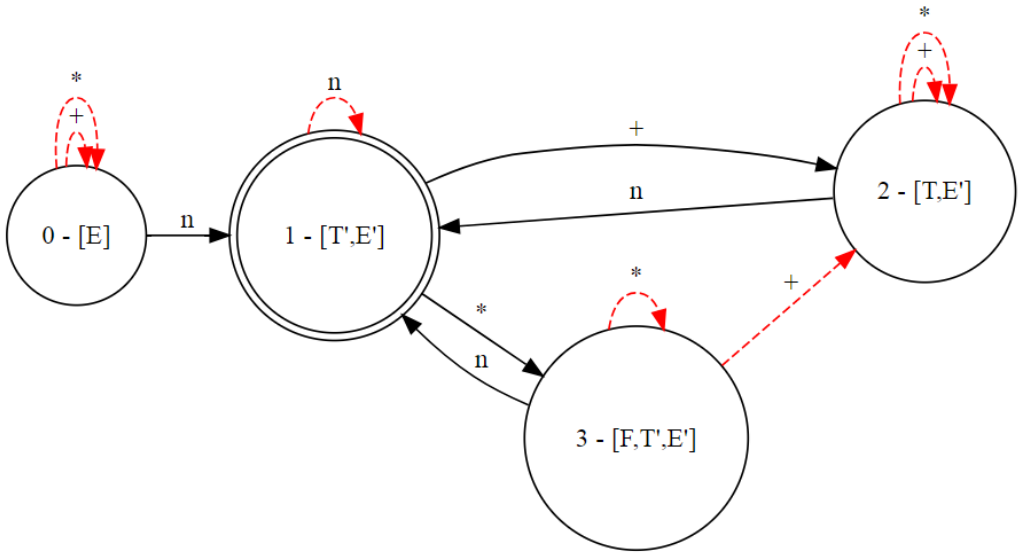
$E \quad ::= TE';$

$E' \quad ::= '+' \; TE' \mid \varepsilon \; ;$

$T \quad ::= FT';$

$T' \quad ::= '*' \; FT' \mid \varepsilon \; ;$

$F \; ::= n$



В общем случае граф состояний является бесконечным.

Суперкомпиляция

Суперкомпиляция – процесс анализа и преобразования программ, основанный на схеме метасистемного перехода.

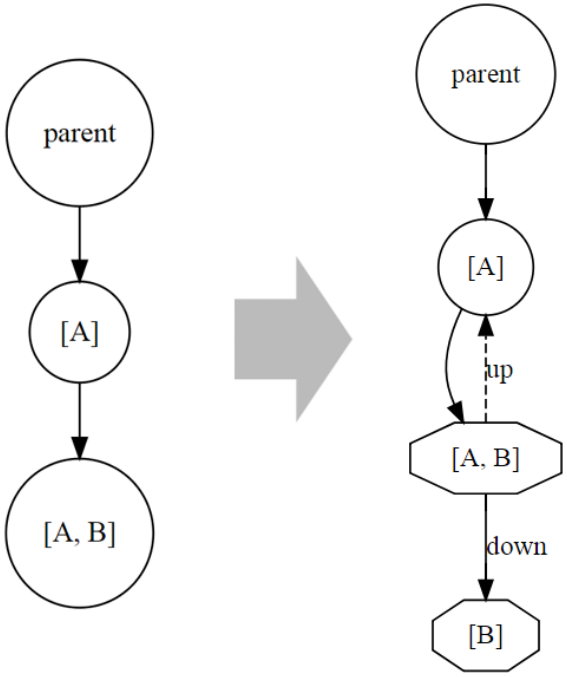
Этапы:

1. Построение **дерева конфигураций**
2. Свертка дерева в **граф конфигураций** (вложение, обобщение)
3. Построение остаточной программы

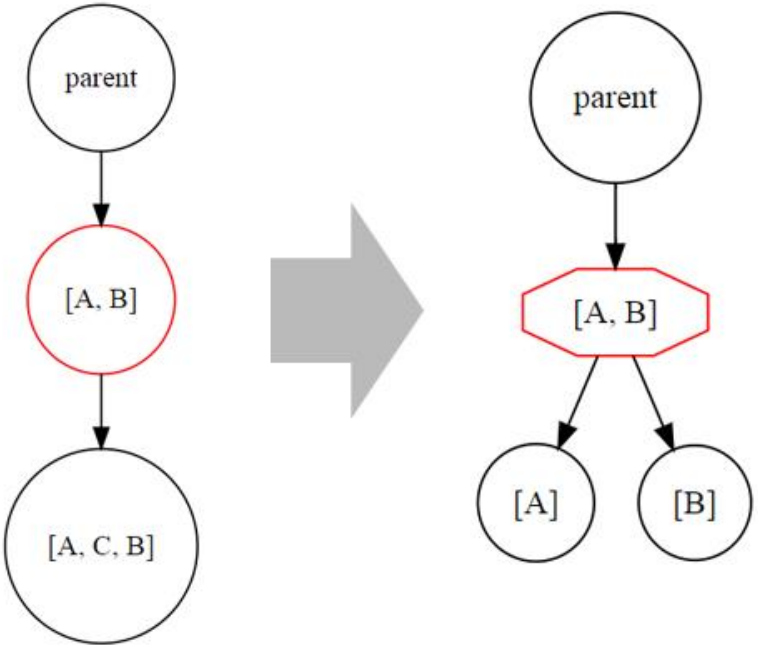
Применительно к синтаксическому анализу, **граф конфигураций** есть **свернутый граф состояний** автомата с магазинной памятью при использовании **расширенных правил восстановления**.

Суперкомпиляция

Вложение



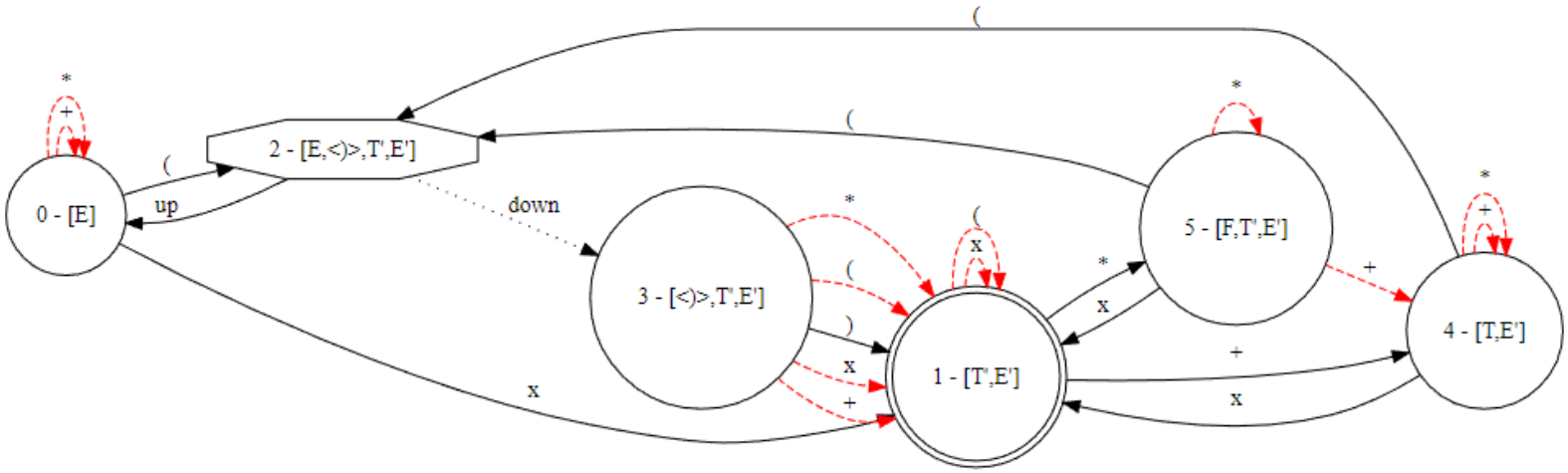
Обобщение



Суперкомпиляция

С учетом сказанного ранее, сформулируем **критерий полноты тестирования**:

Тестирование предсказывающего синтаксического анализатора является полным тогда и только тогда, когда в процессе вывода цепочек из тестового набора окажутся посещены все ребра графа конфигураций.



Реализация | 9

В процессе выполнения данной работы был реализован генератор позитивных и негативных тестов, принимающий на вход описание LL(1)-грамматики.

При реализации использовался язык JavaScript спецификации ECMAScript 2016 на базе платформы Node.js.

Были разработаны следующие компоненты:

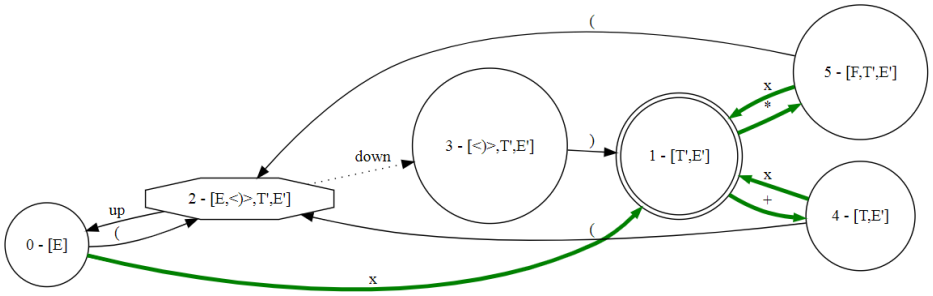
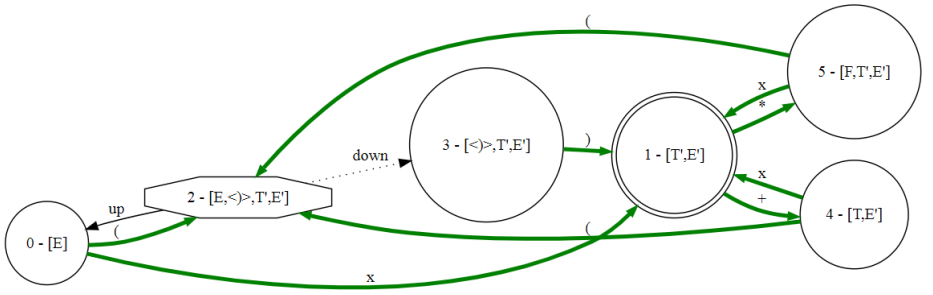
- **Анализатор грамматики** (на основе работы Михаила Макарова, ИУ9)
- **Построитель графа конфигураций**
- **Генератор позитивных тестов**
- **Генератор негативных тестов**

Тестирование

Тестирование проводилось для различных входных грамматик:
Арифметические выражения и операторы, JSON, определения функций Pascal…

Также был успешно проведен тест самоприменимости.

Пример. Грамматика арифметических выражений:

Описание позитивного теста	Состояние ребер в графе конфигураций
1. $x + x * x$ Путь в графе: $[0] \to' x' \to [\mathbf{1}] \to' +' \to [4] \to' x' \to [\mathbf{1}] \to ' * ' \to [5] \to 'x' \to [\mathbf{1}] \to \emptyset$	
2. $(x + (x * (x)))$ *Путь в графе:* $[0] \to '(' \to [0,3] \to 'x' \to [\mathbf{1},3] \to ' + ' \to [4,3] \to' (' \to [0,3,3] \to \cdots \to [3,3,3] \to')' \to [\mathbf{1},3,3] \to [3,3] \to ')' \to [\mathbf{1},3] \to [3] \to ')' \to [\mathbf{1}] \to \emptyset$	

Заключение

Результатом выполнения данной работы является генератор тестов для синтаксических анализаторов LL(1)-грамматик.

Были изучены и применены основные методы суперкомпиляции, техники построения и тестирования синтаксических анализаторов.

В дальнейшем возможно применить использованные приемы для более широких классов грамматик, накладывая дополнительные требования к анализатору грамматики, а также корректируя построение и обход графа конфигураций.

Выражаю благодарность руководителю ВКР **Александру Владимировичу Коновалову**, а также **Антонине Николаевне Непейвода** за помощь и наставления при написании данной работы.

Благодарю за внимание!