YaccConstructor

Задачи на весенний семестр 2016

Автор: Семён Григорьев

Лаборатория языковых инструментов JetBrains
Санкт-Петербургский государственный университет
Математико-механический факультет

8 февраля 2016г.

YaccConstructor

- Исследования в области лексического и синтаксического анализа
- Открытый исходный код
  - `https://github.com/YaccConstructor`
- Основной язык разработки — F#

Требования к знаниям и навыкам

- Знакомство с функциональным программированием (F# или OCaml)
- Знания в области синтаксического анализа
- Знания в области техник трансляции
- Умение читать научные статьи, чужой код, молитвы

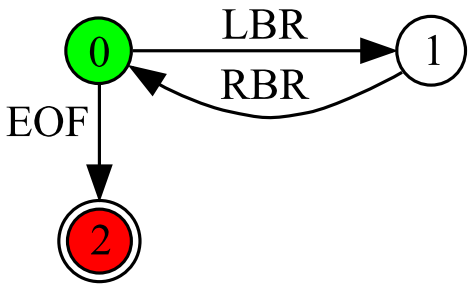
Анализ динамически формируемого кода

Динамически формируемый код

```
string res = "";
for(i = 0; i < l; i++)
    res = "()" + res;
use(res);
```

Аппроксимация

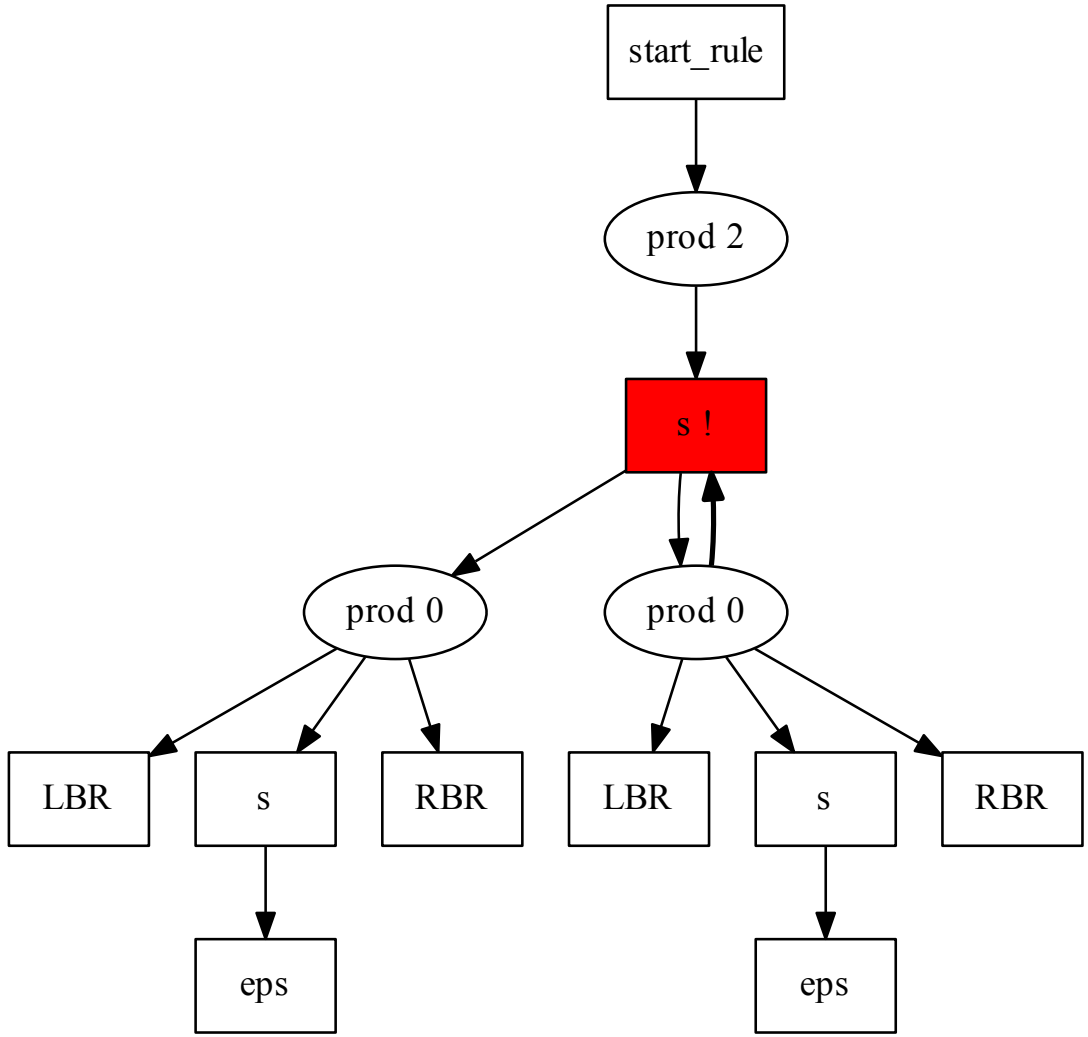
`("()")*`



Грамматика

$$
\begin{aligned}
\mathit{start} &::= s \\
s &::= \texttt{LBR}\ s\ \texttt{RBR}\ s \\
s &::= \epsilon
\end{aligned}
$$

Лес разбора (**SPPF**)



Задача: применение анализа строковых выражений для JavaScript eval

- Цель: трансляция стандартного `eval` в "безопасный"
  - Martin Lester. Information Flow Analysis for a Dynamically Typed Functional Language with Staged Metaprogramming
  - Martin Lester. Analysing Eval using Staged Metaprogramming
- Исследовательская задача: диплом, публикации
- Разбивается на 2 подзадачи
  - Получение аппроксимации
  - Трансляция SPPF в "безопасный `eval`"

Задача: использование SPPF в абстрактном синтаксическом анализе

- Абстрактный синтаксический анализ — один из подходов к анализу динамически формируемого кода
  - K. G. Doh, H. Kim, D. A. Schmidt. Static Validation of Dynamically Generated HTML Documents Based on Abstract Parsing and Semantic Processing
  - E. Verbitskaia, S. Grigorev, D. Avdyukhin. Relaxed Parsing of Regular Approximations of String-Embedded Languages
- Реализовать вычисление семантики по статьям
- Сравнить с нашим подходом
- А можно ли использовать SPPF
- Диплом, публикации

Средства для сертификационного программирования

или

$F\# + F^* = ?$

Сертификационное программирование

- F^* (https://www.fstar-lang.org/tutorial/)
- Coq
- Agda
- ...

```
val sort: l:list int ->
          Tot (m:list int{sorted m
                          /\ (forall i. mem i l = mem i m)})
          (decreases (length l))
let rec sort l = match l with
  | [] -> []
  | pivot::tl ->
    let hi, lo = partition (cmp pivot) tl in
    let l' = append (sort lo) (pivot::sort hi) in
    dedup l'
```

Задача: объединение $F\#$ и F^*

- $F\# + F^* = F\#^*$
- Парсер для $F\#^*$
- Транслятор из AST $F\#$ в AST F^*
- Разбивается на подзадачи (до 3 человек)
- Диплом (вся задача), публикации

Задача: поддержка $F\#^*$ в Microsoft Visual Studio

- Поддержать в модели проекта, редакторе, отладчике
- Разбивается на подзадачи (до 4 человек)
- Диплом (вся задача), публикации

Задача: межъязыковое взаимодействие

- Использовать функции, написанные на F^* в $F\#$(.NET)
- Использовать функции, написанные на $F\#$(.NET) в F^*
- Сохранить типизацию/вывод типов
- Разбивается на подзадачи (до 2 человек)
- Диплом (вся задача), публикации

Контакты

- Почта: rsdpisuy@gmail.com
- Исходный код YaccConstructor: https://github.com/YaccConstructor
- Google+ сообщество: https://goo.gl/DuPWkM