Синтаксис и семантика. Абстрактные синтаксические деревья

Косарев Дмитрий

матмех СПбГУ

20 сентября 2021 г.

Дата сборки: 1 сентября 2022 г.

В этих слайдах

Посмотрим на понятия *синтаксиса* и *семантики*. Примеры языков и построение AST. Операционная семантика

Если кратко...

Синтаксис — это язык, на котором мы записываем что-либо

Семантика$_1$ — смысл, который имеем в виду, записывая слова на языке

Семантика$_2$ — смысл, конкретного предложения в языке

Интерпретация — сопоставление некоторому синтаксису некоторой семантики.

Пример. Синтаксис арифметических выражений

Неформальное описание: *арифметические выражения с 4мя бинарными операциями* $(+, *, -, /$ *и стандартными приоритетами) от целых чисел и переменной* x

Какие выражения являются или не являются арифметическими?

- $1 + 2 * 3$
- $(1 + 2) * (3 - 4))$
- $(-x)/2$
- $\dfrac{1 + 2 * x + x * x}{2}$
- $x^0 + 42$
- 3.1415
- $*$

Пример: Тип для языка арифметических выражений

Неформальное описание: *арифметические выражения с 4мя бинарными операциями* $(+, *, -, /$ *и стандартными приоритетами) от целых чисел и переменной* x.

Это другой, также вполне допустимый вариант

Пример: формулы с целыми числами и кванторами

Неформальное описание:

- константы-числа и инфиксные бинарные операции $(+, /, *, -)$
- кванторы $\forall, \exists$ и переменные $(x, y, z, \dots)$
- (так называемые) *предикатные символы* для чисел $(>, <, \leqslant, \equiv, \neq)$
- логические связки: $\vee$ – или, $\wedge$ – и, $\Rightarrow$ – если ... то ..., $\neg$ – отрицание
- иногда добавляют логические константы: $\top$ (истина, true, top) и $\bot$ (ложь, false, bottom)

Примеры формул и не формул

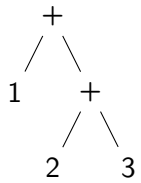
- $\forall x\ \exists y\ (x > y)$
- $\exists x\ \forall y\ (x > y)$
- $\forall x\ \forall y\ \big((x \equiv y) \Leftrightarrow (y \equiv x)\big)$
- $\exists y\ (x > y)$
- $\exists x\ y\ (x + 1 > y)$
- $\exists 1\ (2 > 1)$

Пример: тип для формул

Абстрактные синтаксические деревья (AST)

Вспомним синтаксис арифметических выражений. Пример: $1 + (2 + 3)$

Это выражение будет выглядеть в AST как



Мы абстрагировались в том числе от:

- префиксности и инфиксности
- приоритетов и скобочек
- конкретных позиций в файле

Что может пойти не так с AST

- Может получиться AST не того языка
  - Рекомендуется в комментариях писать, что имелось в виду под тем или иным конструктором в AST

Что может пойти не так с AST

- Может получиться AST не того языка
  - Рекомендуется в комментариях писать, что имелось в виду под тем или иным конструктором в AST
- Надо избегать случаев, когда некоторым значениям AST сложно придать смысл. Пример

Что может пойти не так с AST

- Может получиться AST не того языка
  - Рекомендуется в комментариях писать, что имелось в виду под тем или иным конструктором в AST
- Надо избегать случаев, когда некоторым значениям AST сложно придать смысл.
- Лучше избегать случаев, когда одну и ту же информацию можно представить двумя различными способами

Интерпретация

$$\text{Интерпретация} : \text{Синтаксис} \longrightarrow \text{Семантика}$$

В домашнем задании будут подзадачи

- Задать синтаксис (AST) мини-языка
- Сделать интерпретатор (задать "адекватную" семантику)
- ...

Семантики бывают

1. Заданы в операционном стиле (*операционные семантики*)
   - назначение: смоделировать как что-то вычисляется
   - большого или малого шага
2. В денотационном стиле для некоторого домена $\mathcal{D}$ (*денотационные семантики*)
   - Объяснить смысл синтаксиса в уже известных терминах из $\mathcal{D}$

Операционная семантика для арифметических выражений (одна из)

Операционная семантика для арифметических выражений (одна из)

У нас в синтаксисе присутствуют целые числа, но нам никто не запрещает считать их вещественными...

Операционная семантика для арифметических выражений (одна из)

У нас в синтаксисе присутствуют целые числа, но нам никто не запрещает считать их вещественными...

Возможное улучшение данной семантики:

Малого и большого шага

Семантика **малого** шага

- каким-то образом демонстрирует промежуточные этапы

Семантика **большого** шага

- выдает сразу результат
- более эффективная (но менее познаваемая), чем малого

P.S.

Там ниже дополнительные примеры про написание типов

JSON

- Числа
- Строки
- Массивы
- Объекты как набор пар “ключ-значение”

JSON

- Числа
- Строки
- Массивы
- Объекты как набор пар “ключ-значение”

Пример про почту (1/2)

Пример про почту (1/2)

Хочется, чтобы у контакта был *хотя бы один* адрес: либо электронной, либо физической почты.

Пример про почту (1/2)

Хочется, чтобы у контакта был *хотя бы один* адрес: либо электронной, либо физической почты.

Что вы думаете о вот таком?

Пример про почту (2/2)

Если, посмотрев на тип, сразу понятно какие состояния корректные, а какие нет, то это считается хорошим дизайном.

Пример взят отсюда.

Содержательный пример (1/2) из [?]

Содержательный пример (2/2)

Содержательный пример (2/2)

Лозунг: “плохие” состояния (значения) должны быть непредставимы в типах.

Ссылки

OCaml for the Masses
Yaron Minsky
ссылка