Implementation of static and dynamic semantics for a calculus with algebraic effects and handlers using PLT Redex

Maciej Buszka

Instytut Informatyki UWr

04.10.2019

Formalizacja języków programowania

Wspiera:

- Przewidywalność
- Ortogonalność
- Spójność

Wynik: łatwiejsze utrzymanie i rozwój programów

Narzędzie: rachunki formalne

Efekty obliczeniowe (*side effects*)

- Zjawisko powszechne i znane każdemu programiście `print "Hello World!"`
- Zrozumienie działania programu wymaga nielokalnego wnioskowania
- Wpływ programu na otoczenie
- Dwie ogólne kategorie:
  - Modyfikacja lub odczytanie stanu

Efekty obliczeniowe (*side effects*)

- Zjawisko powszechne i znane każdemu programiście `print "Hello World!"`
- Zrozumienie działania programu wymaga nielokalnego wnioskowania
- Wpływ programu na otoczenie
- Dwie ogólne kategorie:
  - Modyfikacja lub odczytanie stanu
  - Alternatywne ścieżki sterowania

Efekty obliczeniowe - podejścia konwencjonalne

- Ignorowanie efektów
  - C, C++, Java, OCaml, Scala
  - Poważne błędy programów
  - Niemożliwość wnioskowania równościowego
  - Niejawne zależności między modułami

Efekty obliczeniowe - podejścia konwencjonalne

- Ignorowanie efektów
  - C, C++, Java, OCaml, Scala
  - Poważne błędy programów
  - Niemożliwość wnioskowania równościowego
  - Niejawne zależności między modułami
- Podzielenie języka na dwie części
  - Nieprzyjazne dla nowych programistów
  - Łączenie różnych efektów jest skomplikowane (np transformatory monad)
  - Problem z dodaniem efektów *post-factum*

Efekty algebraiczne

- Jednolita składnia
- Pomocny system typów
- Kompozycja efektów
- Modelowanie obu kategorii
- Definicja własnych efektów
- Nowe, aktualnie rozwijane zagadnienie

Cel pracy

- Stworzenie formalnego rachunku
  - Efekty algebraiczne
  - Konstrukcje bazowe
- Implementacja modelu
  - Algorytmiczna inferencja typu i efektów
  - Obrazowanie wykonania krok po kroku

Rachunek

- Abstrakcyjne operacje
  - Nie muszą być zdefiniowane a priori
  - Przyjmują jeden argument, zwracają jedną wartość
- Wyrażenia obsługujące (*handler*)
  - Dostęp do wznowienia
  - Semantyka tzw. głębokiej obsługi
  - Obsługa wielu operacji naraz
  - Klauzula `return`
- Wyrażenia podnoszące (*lift*)
  - 'przeskoczenie' najbliższego wyrażenia obsługującego
- Wartości bazowe oraz operacje na nich

Model

- Zaimplementowany za pomocą biblioteki `PLT Redex`
- Składnia abstrakcyjna
  - Dodatkowo: przyjazny język
- System typów
  - Algorytmiczna inferencja typu i efektów wyrażenia
- Relacja redukcji
  - Wykonanie krok po kroku
- Maszyna abstrakcyjna
  - Jawne środowisko
  - Zgodność z relacją redukcji
- Testy

Podsumowanie

Wynik:

- Formalny rachunek z efektami algebraicznymi
- Wykonywalny model rachunku
  - Wizualizacja przekształceń
  - Inferencja typu i efektów

Możliwości modelu:

- Iteracyjna rozbudowa rachunku
- Testowanie semantyki

Dziękuję za uwagę

Implementation of static and dynamic semantics for a calculus with algebraic effects and handlers using PLT Redex

Maciej Buszka

Instytut Informatyki UWr

04.10.2019