Data-Driven Detection of Recursive Program Schemes

Martin Hofmann and Ute Schmid
martin.hofmann@uni-bamberg.de, ute.schmid@uni-bamberg.de

Inductive Programming (IP)

- Inductive Synthesis of Functional Programs
  - not only concepts, as in ILP, but recursive programs
  - from incomplete specifications (IO examples)
  - 100% accuracy on training data
  - challenging subclass of machine learning
  - still little researched niche

- Potential Applications
  - End-user programming
  - Programming-by-example
  - Test-Driven Development
  - AI & Cognitive Psychology (planning, human learning)

Approaches

- Analytical
  - data-driven, detect regularities in IOs
  - fold them into recursive definition
- Generate & Test
  - use IOs for test only
  - systematic, enumerative search
  - random search, genetic algorithms

- ICoT II
  - combined approach
  - guide search by three analytical operators

Exponential Complexity

IP is a search problem in the space of all candidate programs.

Alternative strategies to deal with comlexity:

- Search full space
- Hard coded a priori knowledge reduces expressiveness.
- Extensive specification requires expert knowledge.

Type morphisms as program schemes allow to reduce complexity without losing expressiveness!

Type Morphisms as Program Schemes

Exist uniquely for any inductive type [3]

Structural Recursion on lists (reduce-map-filter)

Recursive higher-order schemes:

- foldr (x:xs) = foldr x xs
- map f (x:xs) = f(x) : map f xs
- filter f (x:xs) = if f x then (x:xs) else ()

Universal Property of foldr:

- foldr is unique for any inductive type

Cata morphism generalise structural recursion for arbitrary inductive data types

Paramorphisms generalise primitive recursion.

⇒ Detect applicability in IOs! ⇐

Empirical Results

Iterations of the ICoT-II algorithm

add 13 2 4
add1 3 3
and 2 2
or 2 2
not 2
fib 173 2
fib1 173 2
map 3 3
map1 3 3
filter 0 0
filter1 0 0
length 3 3
length1 3 3
mirror 4 4
mirror1 4 4
odd 2 2
even 0 0
rev 1 1
rev1 1 1
zeros 2 2
zeros1 2 2

References


Empirical Results

Iterations of the ICoT-II algorithm

add 13 2 4
add1 3 3
and 2 2
or 2 2
not 2
fib 173 2
fib1 173 2
map 3 3
map1 3 3
filter 0 0
filter1 0 0
length 3 3
length1 3 3
mirror 4 4
mirror1 4 4
odd 2 2
even 0 0
rev 1 1
rev1 1 1
zeros 2 2
zeros1 2 2

⇒ Detect applicability in IOs! ⇐

Empirical Results

Iterations of the ICoT-II algorithm

add 13 2 4
add1 3 3
and 2 2
or 2 2
not 2
fib 173 2
fib1 173 2
map 3 3
map1 3 3
filter 0 0
filter1 0 0
length 3 3
length1 3 3
mirror 4 4
mirror1 4 4
odd 2 2
even 0 0
rev 1 1
rev1 1 1
zeros 2 2
zeros1 2 2

⇒ Detect applicability in IOs! ⇐