A Fixedpoint Approach to (Co)Inductive Definitions

Lawrence C. Paulson
Computer Laboratory
University of Cambridge
England
lcp@cl.cam.ac.uk

Thanks: SERC grants GR/G53279, GR/H40570; ESPRIT Project 6453 ‘Types’
Inductive Definitions

- datatypes
  - finite lists, trees
  - syntax of expressions, ...

- inference systems
  - transitive closure of a relation
  - transition systems
  - structural operational semantics

Supported by Boyer/Moore, HOL, Coq, ..., Isabelle/ZF
Coinductive Definitions

- **codatatypes**
  - *infinite* lists, trees
  - syntax of *infinite* expressions, ...

- **bisimulation relations**
  - process equivalence
  - uses in functional programming (Abramsky, Howe)

Supported by ...?, ..., Isabelle/ZF
The Knaster-Tarski Fixedpoint Theorem

$h$ a monotone function

$D$ a set such that $h(D) \subseteq D$

The least fixedpoint $\text{lfp}(D, h)$ yields inductive definitions

The greatest fixedpoint $\text{gfp}(D, h)$ yields coinductive definitions

A general approach:

- handles all provably monotone definitions

- works for set theory, higher-order logic, ...
An Implementation in Isabelle/ZF

- **Input**
  - description of *introduction rules* & tree’s *constructors*
  - *theorems* implying that the definition is *monotonic*

- **Output**
  - (co)induction rules
  - case analysis rule and *rule inversion* tools, . . .

*flexible, secure, . . . but fast*
Working Examples

- lists
- terms recursive over lists: \( \text{term}(A) = A \times \text{list}(\text{term}(A)) \)
- primitive recursive functions
- lazy lists
- bisimulations for lazy lists
- combinator reductions; Church-Rosser Theorem
- mutually recursive trees & forests
Other Work Using Fixedpoints

The HOL system:

- Melham’s induction package: special case of Fixedpoint Theorem
- Andersen & Petersen’s induction package
- (no HOL datatype package uses fixedpoints)

Coq and LEGO:

- (Co)induction *almost* expressible in base logic (CoC)
- … inductive definitions are built-in
Limitations & Future Developments

- infinite-branching trees
  - justification requires proof
  - would be easier to build them in!

- recursive function definitions
  - use well-founded recursion
  - distinct from datatype definitions

- port to Isabelle/HOL