Unifying Lazy and Strict Computations

Walter Guttmann
University of Canterbury

1. Overview
2. Lazy Computations
3. Recursion
4. Iteration

Computation models

- mathematical descriptions
- based on real computing systems
- simplified, generalised
  - sequential non-deterministic computations
  - relational model, ignore intermediate states
- varying precision
  - infinite, aborting executions
  - partial, total, general correctness
- unify models
  - so far: strict computations
  - new: lazy computations

Lazy computations

- in functional programming
  - compute only the necessary parts
  - support infinite data structures
  - improve modularity
- in imperative programming
  - state with variables
  - values of variables not always needed
  - avoid unnecessary infinite, aborting executions
  - relational model

Unifying models

- structure diversity of models
  - understand their connections
  - reuse theories
  - characterise individual models
  - discover new models
- algebraic structures
  - elements: computations, programs, specifications
  - operations: program constructs
  - axioms, theorems: laws, program transformations

What is unified?

- non-deterministic choice, refinement order, intersection
- sequence, iteration
- conditions, preconditions
- recursion
  - approximation order
  - infinite executions
  - loops are a special case
- neutral, absorbing, least, greatest elements

How are these unified?

- bounded distributive lattices
- variants of semirings, Kleene algebras, omega algebras
- test, domain semirings
- reduce approximation order to semilattice order
  - use domain to extract states with infinite executions
  - axioms about endless loop
- iterings
  - unary operation for unifying iteration
  - simulation axioms instead of induction axioms

Unifying lazy and strict computations

- reduce approximation order to semilattice order
  - use domain to extract states with infinite executions
  - axioms about endless loop
  - problem: previous axioms hold only in strict models

- iterings
  - unary operation for unifying iteration
  - simulation axioms instead of induction axioms
  - problem: previous axioms hold only in strict models

Unifying lazy and strict computations

- reduce approximation order to semilattice order
  - use domain to extract states with infinite executions
  - axioms about endless loop
  - problem: previous axioms hold only in strict models
  - solution: use weaker axioms
- iterings
  - unary operation for unifying iteration
  - simulation axioms instead of induction axioms
  - problem: previous axioms hold only in strict models

Unifying lazy and strict computations

- reduce approximation order to semilattice order
  - use domain to extract states with infinite executions
  - axioms about endless loop
  - problem: previous axioms hold only in strict models
  - solution: use weaker axioms
- iterings
  - unary operation for unifying iteration
  - simulation axioms instead of induction axioms
  - problem: previous axioms hold only in strict models
  - problem: cannot use unary operation in lazy model

Unifying lazy and strict computations

- reduce approximation order to semilattice order
  - use domain to extract states with infinite executions
  - axioms about endless loop
  - problem: previous axioms hold only in strict models
  - solution: use weaker axioms
- iterings
  - unary operation for unifying iteration
  - simulation axioms instead of induction axioms
  - problem: previous axioms hold only in strict models
  - problem: cannot use unary operation in lazy model
  - solution: axiomatise binary operation of omega algebra

Unifying Lazy and Strict Computations

Walter Guttmann
University of Canterbury

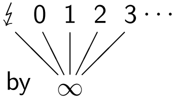
Lazy computations

- assignment $(x_1 := {}^1\!/_0)$ aborts only if value of x_1 needed
  - $(x_1 := {}^1\!/_0) \,;\, (x_1 := 2) = (x_1 := 2)$
  - $(x_1, x_2 := {}^1\!/_0, 2) \,;\, (x_1 := x_2) = (x_1, x_2 := 2, 2)$
- similarly for non-terminating statements
  - $(\text{while } \mathit{true} \text{ do skip}) \,;\, (x_1, x_2 := 2, 2) = (x_1, x_2, \vec{x}_{3..n} := 2, 2, \vec{\infty})$
- construct infinite list $\mathit{ones} = 1{:}\mathit{ones}$
  - $P = f(P) = P \,;\, (\mathit{xs} := 1{:}\mathit{xs})$
  - $\nu f = \bigcap_{n \in \mathbb{N}} f^n(\top) = \bigcap_{n \in \mathbb{N}} (\mathit{xs} := (1{:})^n \infty) = (\mathit{xs} := \mathit{ones})$

Relational model

- variables $x_1, \ldots, x_n$ with values $x_i \in D_i$
- state $\vec{x} \in D_{1..n} = \prod_{i \in 1..n} D_i$
- computation $P \in D_{1..n} \leftrightarrow D_{1..n}$
- $(\vec{x}, \vec{x}') \in P \Leftrightarrow$ execution of P with input $\vec{x}$ may yield output $\vec{x}'$
- non-deterministic if $|P(\vec{x})| > 1$

  $(x := 3x{+}1) \cup (x := 4x{+}1) = \{(0,1),(1,4),(1,5),(2,7),(2,9),\ldots\}$
- lazy computations
  - 'undefined' value $\lightning \in D_i$
  - 'non-terminating' value $\infty \in D_i$
  - D_i partially ordered, for $D_i = \mathbb{N} \cup \{\infty, \lightning\}$ by



  - image sets $P(\vec{x})$ upward-closed

Algebraic structure

- bounded distributive lattice $(S, +, \curlywedge, 0, \top)$
  - non-deterministic choice $+$
  - refinement $x \leq y \Leftrightarrow x + y = y$
- omega algebra $(S, +, \cdot, ^*, ^\omega, 0, 1)$ without $x0 = 0$
  - sequential composition $\cdot$
  - finite iteration * with $y^* z = \mu(\lambda x.yx + z)$
  - infinite iteration $^\omega$ with $y^\omega + y^* z = \nu(\lambda x.yx + z)$
- domain $d : S \to S$
  - $x = d(x)x$
  - $d(xy) = d(xd(y))$
  - $d(x + y) = d(x) + d(y)$
  - $d(0) = 0$
  - $d(x) \leq 1$

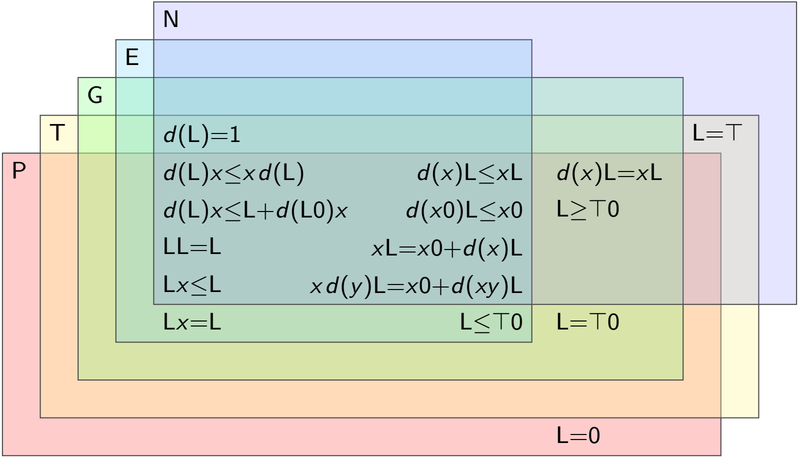
Recursion

- least fixpoint in approximation order
  - $x \sqsubseteq y \;\Leftrightarrow\; x \leq y + \mathsf{L} \;\wedge\; d(\mathsf{L})y \leq x + d(x0)\top$
- use domain for infinite executions of a computation
  - $x0$ eliminates all finite executions of x
  - L endless loop, all infinite executions
  - $x \curlywedge \mathsf{L}$ infinite executions of x
  - $d(x0)$ or $d(x \curlywedge \mathsf{L})$ states with infinite executions
- axioms for L should
  - imply useful properties of $\sqsubseteq$
  - hold in lazy and strict models

Axioms for L

- weaken previous axioms
  - $x\mathsf{L} = x0 + d(x)\mathsf{L}$
  - $d(\mathsf{L})x \leq xd(\mathsf{L})$
  - $d(\mathsf{L})\top \leq \mathsf{L} + d(\mathsf{L}0)\top$
  - $\mathsf{L}x \leq \mathsf{L}$
  - $x0 \curlywedge \mathsf{L} \leq (x \curlywedge \mathsf{L})0$
- $\mathsf{L}x = \mathsf{L}$ in strict models
- consequences
  - $\sqsubseteq$ partial order with least element L
  - $+$, $\cdot$, $\curlywedge\mathsf{L}$, *, $^\omega$ are $\sqsubseteq$-isotone

Structuring computation models



Recursion theorem

- assume f is $\leq$-, $\sqsubseteq$-isotone and μf, νf exist
  - $\mu f / \nu f / \kappa f$ is $\leq / \geq / \sqsubseteq$-least fixpoint
  - $\sqcap$ is $\sqsubseteq$-meet
- then equivalent
  - κf exists
  - κf and $\mu f \sqcap \nu f$ exist and $\kappa f = \mu f \sqcap \nu f$
  - κf exists and $\kappa f = (\nu f \curlywedge \mathsf{L}) + \mu f$
  - $d(\mathsf{L})\nu f \leq (\nu f \curlywedge \mathsf{L}) + \mu f + d(\nu f\, 0)\top$
  - $d(\mathsf{L})\nu f \leq (\nu f \curlywedge \mathsf{L}) + \mu f + d(((\nu f \curlywedge \mathsf{L}) + \mu f)0)\top$
  - $(\nu f \curlywedge \mathsf{L}) + \mu f \sqsubseteq \nu f$
  - $\mu f \sqcap \nu f$ exists and $\mu f \sqcap \nu f = (\nu f \curlywedge \mathsf{L}) + \mu f$
  - $\mu f \sqcap \nu f$ exists and $\mu f \sqcap \nu f \leq \nu f$

Iteration theorem

- while-loop
  - while p do w = if p then (w ; while p do w) else skip
  - $f(x) = yx + z$
  - $\kappa f = (y^\omega \curlywedge \mathsf{L}) + y^* z = d(y^\omega)\mathsf{L} + y^* z$
- in strict models
  - $\kappa f = y^\circ z$
  - $y^\circ = d(y^\omega)\mathsf{L} + y^*$

Unary itering

- itering $(S, +, \cdot, ^\circ, 0, 1)$
  - $(x + y)^\circ = (x^\circ y)^\circ x^\circ$
  - $(xy)^\circ = 1 + x(yx)^\circ y$
  - $zx \leq yy^\circ z + w \;\Rightarrow\; zx^\circ \leq y^\circ(z + wx^\circ)$
  - $xz \leq zy^\circ + w \;\Rightarrow\; x^\circ z \leq (z + x^\circ w)y^\circ$
- models
  - Kleene algebra $x^\circ = x^*$
  - omega algebra $x^\circ = x^\omega 0 + x^*$
  - demonic refinement algebra $x^\circ = x^\omega$
  - extended designs $x^\circ = d(x^\omega)\mathsf{L} + x^*$

Iterings and lazy computations

- in lazy model
  - $f(x) = x$
  - $\kappa f = \mathsf{L} \neq 0 = 1^\circ 0$
  - cannot use unary $^\circ$ for κf
- omega algebra has binary $y \star z = y^\omega + y^* z$
  - not valid in some strict models
- axiomatise $\star$ instead of definition
  - independent of omega algebra, Kleene algebra

Binary itering

- binary itering $(S, +, \cdot, \star, 0, 1)$
  - $(x + y) \star z = (x \star y) \star (x \star z)$
  - $(xy) \star z = z + x((yx) \star (yz))$
  - $x \star (y + z) = (x \star y) + (x \star z)$
  - $(x \star y)z \leq x \star (yz)$
  - $zx \leq y(y \star z) + w \;\Rightarrow\; z(x \star v) \leq y \star (zv + w(x \star v))$
  - $xz \leq z(y \star 1) + w \;\Rightarrow\; x \star (zv) \leq z(y \star v) + (x \star (w(y \star v)))$
  - $w(x \star (yz)) \leq (w(x \star y)) \star (w(x \star y)z)$
- models
  - itering $x \star y = x^\circ y$
  - omega algebra with $x\top \leq x\top x\top$ and $x \star y = x^\omega + x^* y$
- paper shows properties of $\star$ and Back's theorem

Conclusion

- unifying approach covers
  - lazy computations
  - binary operation for iteration
- future work
  - lazy computations with general correctness
  - independent aborting, finite and infinite executions
  - conditions and aborting, infinite executions