Weak Call-by-Value Lambda Calculus as a Model of Computation in Coq

ITP 2017

Yannick Forster   Gert Smolka

Saarland University,

Programming Systems Lab

Related Work

- Michael Norrish
  Mechanised computability theory
  ITP 2011
- J. Xu, X. Zhang and C. Urban
  Mechanising Turing Machines and computability theory in Isabelle/HOL
  ITP 2013
- Andrea Asperti and Wilmer Ricciotti
  A formalization of multi-tape Turing machines
  TCS 2015
- Andrej Bauer
  First steps in synthetic computability theory
  ENTCS 2006

Cutland: Computability, an introduction to recursive function theory

1.7. *Theorem* (Rice's theorem)

Suppose that $\mathscr{B} \subseteq \mathscr{C}_1$, *and* $\mathscr{B} \neq \emptyset, \mathscr{C}_1$. *Then the problem '*$\phi_x \in \mathscr{B}$*' is undecidable.*

Proof. From the algebra of decidability (theorem 2-4.7) we know that '$\phi_x \in \mathscr{B}$' is decidable iff '$\phi_x \in \mathscr{C}_1 \backslash \mathscr{B}$' is decidable; so we may assume without any loss of generality that the function $f_\emptyset$ that is nowhere defined does not belong to $\mathscr{B}$ (if not, prove the result for $\mathscr{C}_1 \backslash \mathscr{B}$).

Kozen: Automata and Computability:

Proof of Rice's theorem. Let P be a nontrivial property of the r.e. sets. Assume without loss of generality that $P(\varnothing) = \bot$ (the argument is symmetric if $P(\varnothing) = \top$). Since P is nontrivial, there must exist an r.e. set A such that $P(A) = \top$. Let K be a TM accepting A.

Wikipedia:

Let us now assume that $P(a)$ is an algorithm that decides some non-trivial property of $\mathbf{F}_a$. Without loss of generality we may assume that P(*no-halt*) = "no", with *no-halt* being the representation of an algorithm that never halts. If this is not true, then this holds for the negation of the property. Since P decides a non-trivial

INGREDIENTS

- Take terms s, t, u, call closed normal forms *procedures*,
- take evaluation $s \triangleright t$ (functional, t procedure),
- define $\mathcal{E}s := \exists t.\ s \triangleright t$,
- take procedures $\mathrm{T} \neq \mathrm{F}$ such that $\mathrm{T}st \triangleright s$ and $\mathrm{F}st \triangleright t$,
- take retraction $\bar{s}$ into procedures to encode terms,
- do computability theory.

Definitions

u decides p if

$$\forall s.\ \ ps \wedge u\bar{s} \rhd \mathrm{T} \ \vee\ \neg ps \wedge u\bar{s} \rhd \mathrm{F}$$

u recognises p if

$$\forall s.\ \ ps \leftrightarrow \mathcal{E}(u\bar{s})$$

$$u \text{ } \textit{decides } p \text{ if}$$

$$\forall s.\ \ ps \land u\bar{s} \rhd \mathrm{T} \ \lor\ \neg ps \land u\bar{s} \rhd \mathrm{F}$$

Fact

$\lambda s.\neg(s\bar{s} \rhd \mathrm{T})$ *is not decidable.*

Proof.

u decides $\lambda s.\neg(s\bar{s} \rhd \mathrm{T})$:

$$\forall s.\ \ \neg(s\bar{s} \rhd \mathrm{T}) \land u\bar{s} \rhd \mathrm{T} \ \lor\ \neg\neg(s\bar{s} \rhd \mathrm{T}) \land u\bar{s} \rhd \mathrm{F}$$

$$\neg(u\bar{u} \rhd \mathrm{T}) \land u\bar{u} \rhd \mathrm{T} \ \lor\ \neg\neg(u\bar{u} \rhd \mathrm{T}) \land u\bar{u} \rhd \mathrm{F}$$

Contradiction! $\square$

SELECTED RESULTS

- *Self-interpreter.* There is a procedure U such that for all terms s, t:
    1. If $s \rhd t$, then $\mathrm{U}\,\bar{s} \rhd \bar{t}$.
    2. If $\mathrm{U}\,\bar{s}$ evaluates, then s evaluates.
- *Rice's theorem.* Every nontrivial extensional class of procedures is undecidable.
- *Modesty.* L-decidable classes are functionally decidable.
- *Post's Theorem.* A class is decidable if it is recognisable, corecognisable, and logically decidable.

Syntax of L

De Bruijn Terms:

$$s, t \ ::= n \mid s\ t \mid \lambda s \quad (n \in \mathbb{N})$$

$$\begin{array}{lllll} \mathrm{I} = \lambda x.x & \mathrm{T} = \lambda xy.x & \mathrm{F} = \lambda xy.y & \omega = \lambda x.xx & \mathrm{D} = \lambda x.\omega\omega \\ \quad := \lambda 0 & \quad := \lambda(\lambda 1) & \quad := \lambda(\lambda 0) & \quad := \lambda(00) & \quad := \lambda(\omega\omega) \end{array}$$

"Procedure" := closed abstraction

SEMANTICS OF L

Reduction:

$$\frac{}{(\lambda s)(\lambda t) \succ s^0_{\lambda t}} \qquad \frac{s \succ s'}{st \succ s't} \qquad \frac{t \succ t'}{st \succ st'}$$

implemented using capturing single-point substitution

$\equiv$ equivalence closure of $\succ$

$\rhd$ big-step evaluation to abstraction

1. Equational reasoning: $s \equiv s' \to t \equiv t' \to st \equiv s't'$
2. Church Rosser: If $s \equiv t$, then $s \succ^* u$ and $t \succ^* u$ for some u.
3. Unique nfs: If $s \rhd^m t$, $s \rhd^n u$, then $t = u$, $m = n$.

[Plotkin, 1975], [Niehren, 1996], [Dal Lago & Martini, 2008]

Scott encodings and recursion

Encodings

T, F for booleans

$\widehat{n}$ for natural numbers

$\overline{s}$ for terms

Scott constructors

- $\mathrm{Succ}\ \widehat{n} \equiv \widehat{Sn}$
- $\mathrm{A}\ \overline{s}\ \overline{t} \equiv \overline{st}$

Recursion combinator

- $(\rho u)v \equiv u(\rho u)v$

[Mogensen, 1990], [Jansen, 2013]

Verification

Functional specification:

$$\forall mn.\ \ \text{Add}\ \widehat{m}\ \widehat{n}\ \equiv\ \widehat{m+n}$$

By induction from:

$$\text{Add}\ \widehat{0}\ \widehat{n} \equiv \widehat{n} \qquad\qquad \text{Add}\ \widehat{Sm}\ \widehat{n} \equiv \text{Succ}\,(\text{Add}\ \widehat{m}\ \widehat{n})$$

$$\text{Add} := \rho(\lambda amn.mn(\lambda m_0.\text{Succ}(am_0n)))$$

$$\text{Add}\ \widehat{m}\ \widehat{n} \equiv \text{Add}\ \widehat{n}\ \widehat{m}$$

If u decides p and v decides q then $\lambda s.ps \wedge qs$ is decidable.

$\lambda x.ux(vx)$F does the job

(Step-indexed) Interpreter

$$
\begin{aligned}
\mathit{eval} &: \mathbf{N} \to \mathbf{T} \to \mathbf{T}_\perp \\
\mathit{eval}\ n\ k &= \perp \\
\mathit{eval}\ n\ (\lambda s) &= \lfloor \lambda s \rfloor \\
\mathit{eval}\ 0\ (st) &= \perp \\
\mathit{eval}\ (Sn)\ (st) &= \textsf{match}\ \mathit{eval}\ n\ s,\ \mathit{eval}\ n\ t\ \textsf{with} \\
&\quad \mid\ \lfloor \lambda s \rfloor, \lfloor t \rfloor \Rightarrow \mathit{eval}\ n\ s^0_t \\
&\quad \mid\ \_\ \_ \Rightarrow \perp
\end{aligned}
$$

$$s \rhd t \leftrightarrow \exists n.\ \mathit{eval}\ n\ s = \lfloor t \rfloor$$

$$\mathrm{E}\ \widehat{n}\ \overline{s} \equiv \overline{\mathit{eval}\ n\ s}$$

If $s \rhd t$, then $\mathrm{U}\ \overline{s} \rhd \overline{t}$.
If $\mathrm{U}\ \overline{s}$ evaluates, then s evaluates.

Minimisation and Interpreter

$$\text{If } s \rhd t \text{, then } \mathrm{U}\,\overline{s} \rhd \overline{t}.$$
$$\text{If } \mathrm{U}\,\overline{s} \text{ evaluates, then } s \text{ evaluates.}$$

Theorem

There is a procedure C *such that for every unary* u*:*

1. *If* u *is satisfiable, then* $\mathrm{C}u \rhd \widehat{n}$ *for some* n *satisfying* u*.*
2. *If* $\mathrm{C}u$ *evaluates, then* u *is satisfiable.*

$$\mathrm{U} := \lambda x.\mathrm{E}\,(\mathrm{C}(\lambda y.\mathrm{E}\,y\;\,x\;(\lambda z.\mathrm{T})\;\mathrm{F}))\,x$$

Rice in Reality

Kozen:

Proof of Rice's theorem. Let P be a nontrivial property of the r.e. sets. Assume without loss of generality that $P(\varnothing) = \bot$ (the argument is symmetric if $P(\varnothing) = \top$). Since P is nontrivial, there must exist an r.e. set A such that $P(A) = \top$. Let K be a TM accepting A.

Wikipedia:

Let us now assume that $P(a)$ is an algorithm that decides some non-trivial property of $\mathbf{F}_a$. Without loss of generality we may assume that $P(\textit{no-halt})$ = "no", with *no-halt* being the representation of an algorithm that never halts. If this is not true, then this holds for the negation of the property. Since P decides a non-trivial

RICE & SCOTT

Scott: Every class p satisfying the following conditions is undecidable.

1. There are closed terms s_1 and s_2 such that ps_1 and $\neg ps_2$.
2. If s and t are closed terms such that $s \equiv t$ and ps, then pt.

Rice: Every class p satisfying the following conditions is undecidable.

1. There are procedures s_1 and s_2 such that ps_1 and $\neg ps_2$.
2. If s and t are procedures such that $\forall uv.\ s\overline{u} \triangleright v \leftrightarrow t\overline{u} \triangleright v$ and ps, then pt. ("p is extensional")

[Barendregt, 1984]

Rice's theorem

Fact

The class of closed terms s such that $\neg\mathcal{E}(s\overline{s})$ is not recognisable.

Lemma (Reduction)

A class p is unrecognisable if there exists a function f such that:

1. $p(fs) \leftrightarrow \neg\mathcal{E}(s\overline{s})$ *for every closed terms s.*
2. *There is a procedure v such that $v\overline{s} \equiv \overline{fs}$ for all s.*

Rice's theorem

Lemma

Let p be an extensional class such that D is in p and some procedure N is not in p. Then p is unrecognisable.

Proof.

- Define function fs such that
  - $fs \approx D$ if $\neg\mathcal{E}(s\bar{s})$
  - $fs \approx N$ if $\mathcal{E}(s\bar{s})$
- $f := s \mapsto \lambda y.\mathrm{F}(s\bar{s})Ny$
  $v := \lambda x.\mathrm{L}(\mathrm{A}(\mathrm{A}(\mathrm{A}\,\overline{\mathrm{F}}(\mathrm{A}x(\mathrm{Q}x)))\overline{\mathrm{N}})\bar{0})$
- $v\bar{s} \equiv \overline{fs}$ and $p(fs) \leftrightarrow \neg\mathcal{E}(s\bar{s})$
- Reduction lemma

$\square$

Rice's theorem

Lemma

Let p be an extensional class such that D is in p and some procedure N is not in p. Then p is unrecognisable.

Theorem

Every nontrivial extensional class of procedures is undecidable.

Proof.

If u decides p then pD or $\neg pD$ and … □

Computable normal forms

Lemma

There is a function of type $\forall s.\ (\exists t.\ s \rhd t) \to \Sigma t.\ s \rhd t.$

Proof.

- $(\exists t. s \rhd t) \leftrightarrow \exists n.\ \mathit{eval}\ n\ s \neq \bot$
- $\lambda n. \mathit{eval}\ n\ s \neq \bot$ is Coq-decidable
- Use constructive choice (constructive indefinite ground description) to obtain n with $\mathit{eval}\ n\ s = \lfloor t \rfloor$
- $s \rhd t$

□

TYPING TOTAL λ-DEFINABLE FUNCTIONS IN COQ

If u decides p then there is f with $fs = \mathsf{true} \leftrightarrow ps$
$\Rightarrow$ L-decidability implies Coq-decidability

$$\forall u. (\forall n \exists m.\ u\,\widehat{n} \rhd \widehat{m}) \to \{f : \mathbf{N} \to \mathbf{N} \mid \forall s.\ u\,\widehat{s} \rhd \widehat{fs}\}$$

[Larchey-Wendling (2017)]

Post's Theorem

Theorem

If u recognises p and v recognises $\lambda s.\neg ps$, then p is decidable if $\forall s.\, ps \vee \neg ps$.

Without restriction: equivalent to $\neg\neg\mathcal{E}s \to \mathcal{E}s$

[Bauer (2006)]

FURTHER RESULTS

- *Totality.* The class of total procedures is unrecognisable.
- *Parallel or.* There is procedure O such that:
  1. If s or t evaluates, then $\mathrm{O}\,\bar{s}\,\bar{t}$ evaluates.
  2. If $\mathrm{O}\,\bar{s}\,\bar{t}$ evaluates, then either $\mathrm{O}\,\bar{s}\,\bar{t} \triangleright \mathrm{T}$ and $\mathcal{E}\,s$, or $\mathrm{O}\,\bar{s}\,\bar{t} \triangleright \mathrm{F}$ and $\mathcal{E}\,t$.
- *Closure under union.* The union of recognisable languages is recognisable.
- *Scott's theorem.* Every nontrivial class of closed terms closed under $\equiv$ is undecidable.
- *Enumerability.* A class is recognisable if and only if it is enumerable.

CONTRIBUTION

- Elegant model of computation, easy to reason about
- Constructive formalisation of basic computability theory, less than 2000 loc
- Self-Interpreter, Rice, Scott, Post, Totality

Future Work

- "L and Turing Machines can simulate each other with a polynomially bounded overhead in time and a constant-factor overhead in space."
  [Dal Lago, Martini (2008)], [Forster, Kunze, Roth (LOLA 2017)]
- Connect L to other models such as recursive functions.
- Use L to show "real-word" problems undecidable (e.g. from logic)
- Do further computability theory in L (Turing degrees, Myhill isomorphism theorem)
- Automate correctness proofs including time complexity
  [Forster, Kunze (CoqWS 2016)]

https://www.ps.uni-saarland.de/extras/L-computability/

LINES OF CODE UP TO . . .

What?	Lines	cumulated
Definition of L	400	400 loc
Rice's theorem	500	900 loc
Step-indexed interpreter	500	900 loc
Full parallel interpreter	300	1200 loc
Enumerable $\leftrightarrow$ recognisable	600	1500 loc