Lecture
Models of Computation
(DIT310, TDA184)

Nils Anders Danielsson

2016-11-21

Today

- X-computability.
- A self-interpreter for χ.
- Reductions.
- More problems that are or are not computable.
- Rice's theorem.

X-computability

χ-computable functions

Assume that we have methods for representing members of the sets A and B as closed χ expressions.

A partial function $f \in A \rightharpoonup B$ is χ-*computable* if there is a closed expression e such that:

- $\forall a \in A$.
  if $f\ a$ is defined then $e \ulcorner a \urcorner \Downarrow \ulcorner f\ a \urcorner$.
- $\forall a \in A$, $v \in Exp$.
  if $e \ulcorner a \urcorner \Downarrow v$ then $f\ a$ is defined and $v = \ulcorner f\ a \urcorner$.

χ-computable functions

A special case:

A (total) function $f \in A \to B$ is χ-computable if there is a closed expression e such that:

- $\forall a \in A.\ e \ulcorner a \urcorner \Downarrow \ulcorner f\ a \urcorner$.

An alternative characterisation

- Define $CExp = \{ p \in Exp \mid p \text{ is closed} \}$.
- The semantics as a partial function:

$$\begin{aligned} \llbracket \_ \rrbracket &\in CExp \rightharpoonup CExp \\ \llbracket p \rrbracket &= v \ \text{ if } p \Downarrow v \end{aligned}$$

- $f \in A \rightharpoonup B$ is χ-computable iff

$$\exists\, e \in CExp.\ \forall\, a \in A.\ \llbracket e \ulcorner a \urcorner \rrbracket = \ulcorner f\ a \urcorner.$$

Quiz

What would go "wrong" if we decided to represent closed χ expressions in the following way?

A closed χ expression is represented by $\mathsf{True}()$ if it terminates, and by $\mathsf{False}()$ otherwise.

Representation

- The choice of representation is important.
- In this course (unless otherwise noted or inapplicable): The “standard” representation.

Examples

- Addition of natural numbers is χ-computable:

$$
\begin{aligned}
&add \in \mathbb{N} \times \mathbb{N} \to \mathbb{N} \\
&add\ (m, n) = m + n
\end{aligned}
$$

- The intensional halting problem is not χ-computable:

$$
\begin{aligned}
&halts \in CExp \to Bool \\
&halts\ p = \textbf{if}\ p\ \textsf{terminates}\ \textbf{then}\ \textsf{true}\ \textbf{else}\ \textsf{false}
\end{aligned}
$$

- The semantics $[\![\_]\!]$ is computable.

Self-interpreter

Self-interpreter

Goal: Define $eval \in CExp$ satisfying:

- $\forall e, v \in CExp$,
  if $e \Downarrow v$ then $eval \ulcorner e \urcorner \Downarrow \ulcorner v \urcorner$.
- $\forall e, v' \in CExp$,
  if $eval \ulcorner e \urcorner \Downarrow v'$ then there is some v such that $e \Downarrow v$ and $v' = \ulcorner v \urcorner$.

Or: $\forall e \in CExp.\ [\![eval \ulcorner e \urcorner]\!] = \ulcorner [\![e]\!] \urcorner$.

Self-interpreter

$$\textbf{rec}\ eval = \lambda\, e.\ \textbf{case}\ e\ \textbf{of}$$
$$\{ \ldots$$
$$\}$$

Self-interpreter

$$\frac{}{\textsf{lambda}\ x\ \ e \Downarrow \textsf{lambda}\ x\ \ e}$$

$$\textsf{Lambda}(x, e) \to \textsf{Lambda}(x, e)$$

Self-interpreter

$$\frac{e_1 \Downarrow \mathsf{lambda}\ x\ e \qquad e_2 \Downarrow v_2 \qquad e\,[x \leftarrow v_2] \Downarrow v}{\mathsf{apply}\ e_1\ e_2 \Downarrow v}$$

$\mathsf{Apply}(e_1, e_2) \rightarrow \textbf{case}\ \mathit{eval}\ e_1\ \textbf{of}$
$\quad \{\, \mathsf{Lambda}(x, e) \rightarrow \mathit{eval}\ (\mathit{subst}\ x\ (\mathit{eval}\ e_2)\ e)$
$\quad \}$

Exercise: Define subst.

Self-interpreter

$$\frac{e\ [x \leftarrow \mathsf{rec}\ x\ \ e] \Downarrow v}{\mathsf{rec}\ x\ \ e \Downarrow v}$$

$$\mathsf{Rec}(x, e) \rightarrow eval\ (subst\ x\ \mathsf{Rec}(x, e)\ \ e)$$

Self-interpreter

$$\frac{es \Downarrow^{\star} vs}{\mathsf{const}\ c\ es \Downarrow \mathsf{const}\ c\ vs}$$

$\mathsf{Const}(c, es) \rightarrow \mathsf{Const}(c, map\ eval\ es)$

Exercise: Define map.

Self-interpreter

$$\frac{e \Downarrow \mathsf{const}\ c\ vs \qquad Lookup\ c\ bs\ xs\ e' \qquad e'\ [xs \leftarrow vs] \mapsto e'' \qquad e'' \Downarrow v}{\mathsf{case}\ e\ bs \Downarrow v}$$

$$
\begin{array}{l}
\mathsf{Case}(e, bs) \to \mathbf{case}\ eval\ e\ \mathbf{of} \\
\quad \{\, \mathsf{Const}(c, vs) \to \mathbf{case}\ lookup\ c\ bs\ \mathbf{of} \\
\qquad \{\, \mathsf{Pair}(xs, e') \to eval\ (substs\ xs\ vs\ e') \\
\qquad \} \\
\quad \}
\end{array}
$$

Exercise: Define $lookup$ and $substs$.

Self-interpreter

$$
\begin{array}{l}
\textbf{rec}\ \mathit{eval} = \lambda\, e.\ \textbf{case}\ e\ \textbf{of} \\
\quad \{\, \mathsf{Lambda}(x, e) \to \mathsf{Lambda}(x, e) \\
\quad ;\ \mathsf{Apply}(e_1, e_2)\ \to \textbf{case}\ \mathit{eval}\ e_1\ \textbf{of} \\
\qquad\quad \{\, \mathsf{Lambda}(x, e) \to \mathit{eval}\ (\mathit{subst}\ x\ (\mathit{eval}\ e_2)\ e) \} \\
\quad ;\ \mathsf{Rec}(x, e) \qquad \to \mathit{eval}\ (\mathit{subst}\ x\ \mathsf{Rec}(x, e)\ e) \\
\quad ;\ \mathsf{Const}(c, es) \to \mathsf{Const}(c, \mathit{map}\ \mathit{eval}\ es) \\
\quad ;\ \mathsf{Case}(e, bs)\ \to \textbf{case}\ \mathit{eval}\ e\ \textbf{of} \\
\qquad\quad \{\, \mathsf{Const}(c, vs) \to \textbf{case}\ \mathit{lookup}\ c\ bs\ \textbf{of} \\
\qquad\qquad\quad \{\, \mathsf{Pair}(xs, e') \to \mathit{eval}\ (\mathit{substs}\ xs\ vs\ e') \} \\
\qquad\quad \} \\
\quad \}
\end{array}
$$

Note: subst, map, lookup and substs are meta-variables that stand for (closed) expressions.

Quiz

Is the following partial function χ-computable?

$$
\begin{aligned}
&halts \in CExp \rightharpoonup Bool \\
&halts\ p = \\
&\quad \textbf{if}\ p\ \text{terminates}\ \textbf{then}\ \text{true}\ \textbf{else}\ \text{undefined}
\end{aligned}
$$

X-decidable

A function $f \in A \to Bool$ is χ-*decidable* if it is χ-computable. If not, then it is χ-*undecidable*.

X-semi-decidable

A function $f \in A \to Bool$ is χ*-semi-decidable* if there is a closed expression e such that, for all $a \in A$:

- If $f\ a =$ true then $e \ulcorner a \urcorner \Downarrow \ulcorner \text{true} \urcorner$.
- If $f\ a =$ false then $e \ulcorner a \urcorner$ does not terminate.

The halting problem is semi-decidable

The halting problem:

$$halts \in CExp \to Bool$$
$$halts\ p = \textbf{if}\ p\ \textsf{terminates}\ \textbf{then}\ \textsf{true}\ \textbf{else}\ \textsf{false}$$

A program witnessing the semi-decidability:

$$\lambda\, p.\ (\lambda\, \_.\ \textsf{True}())\ (eval\ p)$$

Reductions

Reductions (one variant)

A χ-reduction of $f \in A \rightharpoonup B$ to $g \in C \rightharpoonup D$ consists of a proof showing that, if g is χ-computable, then f is χ-computable.

Reductions

- If f is reducible to g, and f is not computable, then g is not computable.
- Last week we proved that the halting problem is undecidable by reducing another problem to it.

More (un)decidable problems

Semantic equality

- Are two closed χ expressions semantically equal?

$$
\begin{array}{l}
equal \in CExp \times CExp \to Bool \\
equal\ (e_1, e_2) = \\
\quad \mathbf{if}\ [\![e_1]\!] = [\![e_2]\!]\ \mathbf{then}\ \mathsf{true}\ \mathbf{else}\ \mathsf{false}
\end{array}
$$

- The halting problem reduces to this one:

$$
halts = \lambda\, p.\ not\ (equal\ \mathsf{Pair}(p, \ulcorner \mathbf{rec}\ x = x \urcorner))
$$

Pointwise equality

- Pointwise equality:

$$
\begin{array}{l}
\mathit{pointwise\text{-}equal} \in \mathit{CExp} \times \mathit{CExp} \to \mathit{Bool} \\
\mathit{pointwise\text{-}equal}\ (e_1, e_2) = \\
\quad \mathbf{if}\ \forall\ e \in \mathit{CExp}.\ [\![e_1\ e]\!] = [\![e_2\ e]\!] \\
\quad \mathbf{then}\ \mathsf{true}\ \mathbf{else}\ \mathsf{false}
\end{array}
$$

- The previous problem reduces to this one:

$$
\begin{array}{l}
\mathit{equal} = \lambda\, p.\ \mathbf{case}\ p\ \mathbf{of} \\
\quad \{\mathsf{Pair}(e_1, e_2) \to \\
\qquad \mathit{pointwise\text{-}equal} \\
\qquad\quad \mathsf{Pair}(\mathsf{Lambda}(\ulcorner x \urcorner, e_1), \\
\qquad\qquad \mathsf{Lambda}(\ulcorner x \urcorner, e_2)) \\
\quad \}
\end{array}
$$

Termination in n steps

- Termination in n steps:

$$\begin{aligned}&\mathit{terminates\text{-}in} \in \mathit{CExp} \times \mathbb{N} \to \mathit{Bool}\\&\mathit{terminates\text{-}in}\ (e, n) =\\&\quad \mathbf{if}\ \exists\, p \in e \Downarrow v.\ \mid p \mid \leq n\ \mathbf{then}\ \mathsf{true}\ \mathbf{else}\ \mathsf{false}\end{aligned}$$

  $|p|$: The number of rules in the derivation tree.
- Decidable: We can define a variant of the self-interpreter that tries to evaluate e but stops if more than n rules are needed.

Representation

- How do we represent a χ-computable function?
- By the representation of one of the closed expressions witnessing the computability of the function.

Quiz

Is the following problem χ-decidable for $A = Bool$? What if $A = \mathbb{N}$?

Let $Fun = \{f \in A \to Bool \mid f \text{ is } \chi\text{-computable}\}$.

$$
\begin{aligned}
&\mathit{pointwise\text{-}equal}' \in Fun \times Fun \to Bool \\
&\mathit{pointwise\text{-}equal}'\ (f, g) = \\
&\quad \mathbf{if}\ \forall\ a \in A.\ f\ a = g\ a\ \mathbf{then}\ \mathsf{true}\ \mathbf{else}\ \mathsf{false}
\end{aligned}
$$

Hint: Use *eval* or *terminates-in*.

Pointwise equality of computable functions in $Bool \to Bool$

- The function $pointwise\text{-}equal'$ is decidable.
- Implementation:

$$
\begin{aligned}
&pointwise\text{-}equal' = \lambda\, p.\ \mathbf{case}\ p\ \mathbf{of} \\
&\quad \{\, \mathsf{Pair}(f, g) \to \\
&\qquad and\ (equal_{Bool}\ (eval\ \mathsf{Apply}(f, \mathsf{True}()))\\
&\qquad\qquad\qquad\qquad (eval\ \mathsf{Apply}(g, \mathsf{True}())))\\
&\qquad\qquad (equal_{Bool}\ (eval\ \mathsf{Apply}(f, \mathsf{False}()))\\
&\qquad\qquad\qquad\qquad (eval\ \mathsf{Apply}(g, \mathsf{False}())))\\
&\quad \}
\end{aligned}
$$

Pointwise equality of computable functions in $\mathbb{N} \rightarrow Bool$

- The function $pointwise\text{-}equal'$ is undecidable.
- The halting problem reduces to it:

$$\begin{aligned} &halts = \lambda\, p.\ not\ (pointwise\text{-}equal' \\ &\quad \ulcorner \lambda\, n.\ terminates\text{-}in\ \mathsf{Pair}(\llcorner\ code\ p\ \lrcorner, n)\ \urcorner \\ &\quad \ulcorner \lambda\, \_.\ \mathsf{False}()\ \urcorner) \end{aligned}$$

Quiz

Is the following function χ-computable?

$$optimise \in CExp \to CExp$$
$$optimise\ e =$$

some optimally small expression with
the same semantics as e

Size: The number of constructors in the abstract syntax (Exp, Br, $List$, not Var or $Const$).

Full employment theorem for compiler writers

- An optimally small non-terminating expression is equal to $\mathbf{rec}\ x = x$ (for some x).
- The halting problem reduces to this one:

$$
\begin{aligned}
&\mathit{halts} = \lambda p.\ \mathbf{case}\ \mathit{optimise}\ p\ \mathbf{of} \\
&\quad \{\mathsf{Rec}(\_, e) \to \mathbf{case}\ e\ \mathbf{of} \\
&\qquad \{\mathsf{Var}(\_) \quad\ \to \mathsf{True}() \\
&\qquad ;\ \mathsf{Rec}(\_, \_) \to \mathsf{False}() \\
&\qquad ;\ \ldots \\
&\qquad \} \\
&\quad ;\ \ldots \\
&\quad \}
\end{aligned}
$$

Computable real numbers

- Computable reals can be defined in many ways.
- One example, using signed digits:

$$\begin{aligned} &\mathit{Interval} = \\ &\quad \{f \in \mathbb{N} \to \{-1, 0, 1\} \mid f \text{ is } \chi\text{-computable}\} \end{aligned}$$

$$\begin{aligned} &[\![\_]\!] \in \mathit{Interval} \to [-1, 1] \\ &[\![f]\!] = \textstyle\sum_{i=0}^{\infty} f\ i \cdot 2^{-i-1} \end{aligned}$$

- Why signed digits? Try computing the first digit of $0.00000\ldots + 0.11111\ldots$ (in binary notation).

Is a computable real number equal to zero?

- Is a computable real number equal to zero?

$$\begin{aligned}&\mathit{is\text{-}zero} \in \mathit{Interval} \to \mathit{Bool}\\&\mathit{is\text{-}zero}\ x = \mathbf{if}\ [\![x]\!] = 0\ \mathbf{then}\ \mathsf{true}\ \mathbf{else}\ \mathsf{false}\end{aligned}$$

- The halting problem reduces to this one:

$$\begin{aligned}&\mathit{halts} = \lambda\, p.\ \mathit{not}\ (\mathit{is\text{-}zero} \ulcorner \lambda\, n.\\&\quad \mathbf{case}\ \mathit{terminates\text{-}in}\ \mathsf{Pair}(\llcorner\ \mathit{code}\ p\ \lrcorner, n)\ \mathbf{of}\\&\qquad \{\ \mathsf{True}()\ \to \mathsf{One}()\\&\qquad ;\ \mathsf{False}() \to \mathsf{Zero}()\\&\qquad \}\urcorner)\end{aligned}$$

Undecidable problems

- A list on Wikipedia.
- A list on MathOverflow.

Rice's theorem

Rice's theorem

Assume that $P \in CExp \rightarrow Bool$ satisfies the following properties:

- P is non-trivial:
  There are expressions e_{true}, $e_{\text{false}} \in CExp$ satisfying $P\ e_{\text{true}} = \text{true}$ and $P\ e_{\text{false}} = \text{false}$.
- P respects pointwise semantic equality:

$$\begin{array}{l} \forall\ e_1, e_2 \in CExp. \\ \quad \text{if}\ \ \forall\ e \in CExp.\ [\![e_1\ e]\!] = [\![e_2\ e]\!]\ \text{then} \\ \qquad P\ e_1 = P\ e_2 \end{array}$$

Then P is χ-undecidable.

Rice's theorem

The halting problem reduces to P:

$$
\begin{array}{l}
halts = \lambda\, e.\ \mathbf{case}\ P\ \ulcorner \lambda\, \_.\ \mathbf{rec}\ x = x\ \urcorner\ \mathbf{of} \\
\quad \{\,\mathsf{False}() \rightarrow \\
\qquad P\ \ulcorner \lambda\, x.\ (\lambda\, \_.\ e_{\mathsf{true}}\ x)\ (eval\ \llcorner\ code\ \ e\ \lrcorner)\ \urcorner \\
\quad ;\ \mathsf{True}()\ \rightarrow \\
\qquad not\ (P\ \ulcorner \lambda\, x.\ (\lambda\, \_.\ e_{\mathsf{false}}\ x)\ (eval\ \llcorner\ code\ \ e\ \lrcorner)\ \urcorner) \\
\quad \}
\end{array}
$$

Quiz

Which of the following problems are χ-decidable?

- Is $e \in CExp$ an implementation of the successor function for natural numbers?
- Is $e \in CExp$ syntactically equal to $\lambda\, n.\ \mathsf{Succ}(n)$?

Summary

- X-computability.
- A self-interpreter for χ.
- Reductions.
- More problems that are or are not computable.
- Rice's theorem.

Please give any kind of feedback on the course