λ calculus: Lecture 4

Madhavan Mukund

Chennai Mathematical Institute

madhavan@cmi.ac.in

PLC, 17 March 2017

One step reduction

- Can have other reduction rules like β
- Observe that $\lambda x.(Mx)$ and M are equivalent with respect to β-reduction
- New reduction rule η

$$\lambda x.(Mx) \to_\eta M$$

- Given basic rules β, η, ..., we are allowed to use them "in any context"
- Define a one step reduction relation $\to$ inductively

$$\frac{M \to_x M'}{M \to M} \quad x \in \{\beta, \eta, \ldots\} \qquad \frac{M \to M'}{\lambda x.M \to \lambda x.M'} \qquad \frac{M \to M'}{MN \to M'N} \qquad \frac{N \to N'}{MN \to MN'}$$

Normal forms

- Computation — a maximal sequence of reduction steps
- "Values" are expressions that cannot be further reduced: normal forms
- Allow reduction in any context $\Rightarrow$ multiple expressions may qualify for reduction in one step

Natural questions

- Does every term reduce to a normal form?
- Can a term reduce to more than one normal form, depending on order reduction strategy?
- If a term has a normal form, can we always find it?

Normal forms . . .

Does every term reduce to a normal form?

- Consider $(\lambda x.xx)(\lambda x.xx)$
- $(\lambda x.xx)(\lambda x.xx) \rightarrow_\beta (\lambda x.xx)(\lambda x.xx)$
  - Reduction never terminates
- Call this term Ω

Normal forms . . .

Can a term reduce to more than one normal form, depending on order reduction strategy?

- Consider $\langle \mathit{False} \rangle \Omega = (\lambda yz.z)((\lambda x.xx)(\lambda x.xx))$
- Outermost reduction:
  $(\lambda yz.z)((\lambda x.xx)(\lambda x.xx)) \rightarrow \lambda z.z$
- Innermost reduction:
  $(\lambda yz.z)((\lambda x.xx)(\lambda x.xx)) \rightarrow (\lambda yz.z)((\lambda x.xx)(\lambda x.xx)) \rightarrow \cdots$
- Choice of reduction strategies may determine whether a normal form is reached . . .
- . . . but the question is, can more than one normal form be reached?

Normal forms . . .

If a term has a normal form, can we always find it?

- We have seen how to encode recursive functions in λ-calculus
- Given a recursive function f and an argument n, we cannot determine, in general, if computation of $f(n)$ terminates
- Computing $f(n)$ is equivalent to asking if $\langle f \rangle \langle n \rangle$ achieves a normal form

Normal forms . . .

Can a term reduce to more than one normal form, depending on order reduction strategy?

- Define an equivalence relation $\leftrightarrow$ on λ-terms

$$M \leftrightarrow N \text{ iff } \exists P.\ P \rightarrow^* M, P \rightarrow^* N$$

$M \leftrightarrow N$ if both M and N can be obtained by reduction from a common "ancestor" P

- $\leftrightarrow$ is the symmetric transitive closure of $\rightarrow^*$

$$\frac{M \rightarrow^* N}{M \leftrightarrow N} \quad \frac{M \leftrightarrow N}{N \leftrightarrow M} \quad \frac{M \leftrightarrow N, N \leftrightarrow P}{M \leftrightarrow P}$$

- In general, for any reflexive, transitive relation R, can define the symmetric, transitive closure $\overset{R}{\leftrightarrow}$

Church-Rosser Theorem

Diamond property or Church-Rosser property

- Let R be any reflexive, transitive relation (such as $\rightarrow^*$)
- R has the diamond property if, whenever $X\ R\ Y$ and $X\ R\ Z$ there is W such that $Y\ R\ W$ and $Z\ R\ W$

Theorem [Church-Rosser]

Let R be Church-Rosser. Then $M \overset{R}{\leftrightarrow} N$ implies there exists Z, $M\ R\ Z$ and $N\ R\ Z$

Proof By induction on the definition of $\overset{R}{\leftrightarrow}$

Church-Rosser Theorem

Corollary [Church-Rosser]

Let R be a reduction relation that is Church-Rosser. Then a term can have at most one normal form with respect to R

Proof By picture

Church-Rosser Theorem

Is $\rightarrow^*$ Church-Rosser?

Consider $(\lambda x.xx)((\lambda x.x)(\lambda x.x))$

Two possible reductions

- $(\lambda x.xx)((\lambda x.x)(\lambda x.x)) \rightarrow$
  $((\lambda x.x)(\lambda x.x))((\lambda x.x)(\lambda x.x))$ (Outermost)
- $(\lambda x.xx)((\lambda x.x)(\lambda x.x)) \rightarrow ((\lambda x.xx)(\lambda x.x)$ (Innermost)

From second option, in one step we get

$$(\lambda x.xx)(\lambda x.x) \rightarrow ((\lambda x.x)(\lambda x.x))$$

Can reach this term from the first option as well, but it requires two steps!

Church-Rosser Theorem

Solution: Define a new notion of one step reduction $\twoheadrightarrow$ such that

- This new reduction is Church-Rosser.
- Its reflexive, transitive closure is equal to $\rightarrow^*$.

Define $\twoheadrightarrow$ as follows.

$$M \twoheadrightarrow M \qquad \frac{M \twoheadrightarrow M'}{\lambda x.M \twoheadrightarrow \lambda x.M'}$$

$$\frac{M \twoheadrightarrow M', N \twoheadrightarrow N'}{MN \twoheadrightarrow M'N'} \qquad \frac{M \twoheadrightarrow M', N \twoheadrightarrow N'}{(\lambda x.M)N \twoheadrightarrow M'\{x \leftarrow N'\}}$$

- $\twoheadrightarrow$ combines nonoverlapping $\rightarrow$ reductions into one parallel step

Normal forms and reduction strategies

- Outermost reduction is also called lazy
  - Arguments to a function are evaluated only when needed
- Normal order reduction — outermost, leftmost
  - Among all possible top level reductions, choose the leftmost
- Lemma: If a normal form exists, normal order reduction is guaranteed to find it
  - Normal form is unique, by Church-Rosser property
- However, normal order reduction is "inefficient"
  - If an argument is duplicated, it must be re-evaluated each time it occurs
- 'Graph reduction": maintain pointers to shared subexpressions, avoid duplicated work
  - Used in Haskell implementations
  - Normal order graph reduction is close to optimal

Recursive definitions

Suppose $F = \lambda x_1 x_2 \ldots x_n E$, where where E contains an occurrence of F

- Choose a new variable f
- Convert E to E^* replacing every F in E by ff
  - If E is of the form $\cdots F \cdots F \cdots$ then E^* is $\cdots (ff) \cdots (ff) \cdots$.

Now write

$$\begin{aligned} G &= \lambda f x_1 x_2 \ldots x_n . E^* \\ &= \lambda f x_1 x_2 \ldots x_n . \cdots (ff) \cdots (ff) \cdots \end{aligned}$$

Then

$$GG = \lambda x_1 x_2 \ldots x_n . \cdots (GG) \cdots (GG) \cdots$$

- GG satisfies the equation defining F
- Write $F = GG$, where $G = \lambda f x_1 x_2 \ldots x_n . E^*$.

Fixed point combinator

- Consider recursive definition $F = \lambda x.x(Fx)$
- Can use the GG trick to get a λ-expression of F

$$\begin{aligned} F &= GG, \text{where } G = \lambda fx.x(ffx) \\ &= \lambda fx.x(\lambda fx.x(ffx)\lambda fx.x(ffx)x) \end{aligned}$$

- Note that $FX = X(FX)$ for any term X
- Fixed point : Given Z, M such that $ZM = M$.
- F Z is a fixed point for Z
- Due to Turing — Θ

Another fixed point combinator

- Let $Y = \lambda h.(\lambda x.h(xx))(\lambda x.h(xx))$
- Consider YH for any term H
- YH

  $\equiv (\lambda h.(\lambda x.h(xx))(\lambda x.h(xx)))H$

  $\rightsquigarrow (\lambda x.H(xx))(\lambda x.H(xx))$

  $\rightsquigarrow H(\lambda x.H(xx))(\lambda x.H(xx))$

  $\equiv H(YH)$
- Due to Haskell Curry

Terms without normal forms

Are all terms without normal forms equally "meaningless"?

Can we define an equivalence $\approx$ on λ-terms such that:

- $(\lambda x M)N \approx M\{x \leftarrow N\}$—that is, $\approx$ the equivalence induced by the β reduction.
- If M and N do not have normal forms, then $M \equiv N$.
- Functions that are equated by $\approx$ yield equivalent results for the same arguments. That is, if $M \approx N$ then for all R, $MR \approx NR$.

Terms without normal forms

Consider the function F defined by

$$Fxb = \text{ if } b \text{ then } x \text{ else } (Fxb)$$

If we unravel FF, we get

$$F = GG, \text{ where } G = \lambda fxb.(\text{ if } b \text{ then } x \text{ else } (ffxb))$$

Consider $FX\langle true\rangle$ and $FX\langle false\rangle$

- $FX\langle true\rangle \to \text{if } \langle T\rangle \text{ then } X \text{ else } (FX\langle true\rangle) \to X$.
- $FX\langle false\rangle \to \text{if } \langle F\rangle \text{ then } X \text{ else } (FX\langle false\rangle) \to FX\langle false\rangle$.

Terms without normal forms

$$
\begin{aligned}
FZ &\to (\lambda xb.(\text{if } b \text{ then } x \text{ else } (Fxb)))Z \\
&\to \lambda b.(\text{if } b \text{ then } Z \text{ else } (FZb)) \\
&\to \lambda b.(\text{if } b \text{ then } Z \text{ else } G), \text{where } G = \text{if } b \text{ then } Z \text{ else } (fZB) \\
&\to \lambda b.(\text{if } b \text{ then } Z \text{ else } (\text{if } b \text{ then } Z \text{ else } G)) \\
&\to \dots
\end{aligned}
$$

- FZ does not terminate for any $Z \Rightarrow FX \approx FY$ for all X, Y
- $FX \approx FY$ implies $FXM \approx FYM$ for all M
- $FX\langle true\rangle \approx FY\langle true\rangle$
- $FZ\langle true\rangle \to Z$ for all Z, so $X \approx Y$ for all X and Y!