COMP4630: $\lambda$-Calculus

4. Standardisation

Michael Norrish
Michael.Norrish@nicta.com.au

Canberra Research Lab., NICTA

Semester 2, 2015

NICTA
Last Time

Confluence
- The property that divergent evaluations can rejoin one another

Proof
- Diamond properties
- Uses parallel reduction (\(\Rightarrow\beta\)); and
- Many inductions

Consequences
- Soundness of \(\lambda\)
- (With an analogous proof) Soundness of \(\lambda\eta\)
- Incompleteness of \(\lambda\)
- (See end of lecture 2)
Today

Introduction

Head Reduction
   Weak Head Reduction

The Proof
   Failing Approaches
   The Right Approach
   Consequences

Conclusion
Objective

From last time, we know that a term $M$ has at most one normal form.

Unfortunately, we also know that not all evaluation strategies will lead to that normal form.

▶ This is **not** inconsistent with confluence.
▶ Why?
Evaluation Strategies

An evaluation strategy is basically a way of answering the question:

Where (i.e., in which sub-term) should I do my next reduction?

Or:

Where should I do the next bit of work?

Some languages (e.g., Java) do not allow for choices to be made at all. They specify a precise evaluation order. Why would they do that?
Evaluation Strategy #1: Applicative Order

Evaluate everything from the bottom up.

- I.e., in \((\lambda v. M) \ N\) work will start with \(M\), passing to \(N\) and performing the top-level \(\beta\)-reduction last

A function’s arguments (and the function itself) will be evaluated before the argument is passed to the function.

Also known as strict evaluation.

(Used in Fortran, C, Pascal, Ada, SML, Java . . .)

Causes \((\lambda x. y) \ \Omega\) to go into an infinite loop.
Evaluation Strategy #2: Normal Order

Evaluate top-down, left-to-right.

- With \((\lambda v. M) \; N\), start by performing the \(\beta\)-reduction, producing \(M[v := N]\)
- Find the top-most, left-most \(\beta\)-redex and reduce it.
- Keep going

This strategy is behind the lazy evaluation of languages like Haskell.

Normal order evaluation will always terminate with a normal form if a term has one. (Proof to come...
Evaluation Strategy Trade-offs

An evaluation strategy might

1. be guaranteed to find normal forms; or
2. aim to perform the least number of $\beta$-reductions

Naïvely,

- normal order reduction does 1;
- applicative order “sort of” achieves 2, but gives up on 1

(In fact, optimal reduction is very difficult to get right.)
Proving Normal Order Evaluation

Our focus is in showing that normal order evaluation is guaranteed to find normal forms.

► (That’s why it’s called normal order...)

Here are the rules:

\[(\lambda v. M) N \rightarrow_n M[v := N]\]

\[M \rightarrow_n M' \quad \text{(\(\lambda v. M\)) \rightarrow_n (\lambda v. M')}\]

\[M \rightarrow_n M' \quad M \text{ not an abstraction}\]

\[MN \rightarrow_n M'N\]

\[N \rightarrow_n N' \quad M \text{ not an abstraction}\]

\[M \text{ in } \beta\text{-nf}\]

\[MN \rightarrow_n MN'\]
Head Reduction

We can divide normal order reduction into two different sorts of reduction.
First, normal order reduction:

\[
(\lambda v. M) N \rightarrow_n M[v := N] \quad \frac{M \rightarrow_n M'}{\lambda v. M \rightarrow_n \lambda v. M'}
\]

\[
\frac{M \rightarrow_n M'}{M \text{ not an abstraction}}
\[
\frac{M \rightarrow_n M' \quad M \text{ not an abstraction}}{M N \rightarrow_n M' N}
\]

\[
\frac{N \rightarrow_n N'}{M \text{ not an abstraction}}
\[
\frac{M \text{ in } \beta\text{-nf}}{M N \rightarrow_n M N'}
\]
Head Reduction

We can divide normal order reduction into two different sorts of reduction.

Then, **head reduction**:

\[
\begin{align*}
(\lambda v. M) \, N & \rightarrow_h M[v := N] \\
\lambda v. M & \rightarrow_h (\lambda v. M')
\end{align*}
\]

\[
\begin{align*}
M & \rightarrow_h M' \quad M \text{ not an abstraction} \\
M \, N & \rightarrow_h M' \, N
\end{align*}
\]

\[
\begin{align*}
N & \rightarrow_h N' \quad M \text{ not an abstraction} \\
M & \text{ in } \beta\text{-nf} \\
M \, N & \not\rightarrow_h M \, N'
\end{align*}
\]

When head reducing, you never reduce to the right of an application.
Hence, Head Normal Forms

Head Normal Form is a reasonable stopping place.

Rules again:

\[
\frac{(\lambda v. M) \ N}{\text{h}} \rightarrow M[v := N]
\]

\[
\frac{M \rightarrow \ M'}{\text{h}}
\]

\[
\frac{M \text{ not an abstraction}}{\text{h}}
\]

\[
\frac{M \ N}{\text{h}} \rightarrow M' \ N
\]

Examples:

- \( v \) is in hnf
- \( (\lambda v. v) \) is in hnf
- \( (\lambda w. v (\lambda u. M) \ N) \) is in hnf
- \( (\lambda w z. v ((\lambda u. M) \ N)) \) is in hnf
Head Normal Forms, Generally

If term $M$ is in hnf, then it will look like:

$$(\lambda \vec{v}. u \, M_1 \cdots M_n)$$

The vector $\vec{v}$ may be empty, $u$ may be free or bound, and the number of extra arguments, $n$, may be 0.

Once a term is in hnf, its top-level structure can’t change.
After Head Reductions

Once a term is in hnf, its top-level structure can’t change.

Any further reductions (of any sort) inside

$$(\lambda \vec{v}. \ u \ M_1 \cdots M_n)$$

will be reductions within an $M_i$.

Each argument will evolve independently, and the number of arguments can’t change.

These are internal reductions $\xrightarrow{i}$.

- If the term is $(\lambda \vec{v}. (\lambda u. M) \ N_1 \ N_2 \cdots)$ (not in hnf) and $M$ reduces, then that is an internal reduction too
- All reductions are either head or internal
Normal Order Reduction Splits in Two

The last rule of normal order reduction (which we deleted to get head reduction):

\[
\frac{N \rightharpoonup_n N' \quad M \text{ not an abstraction} \quad M \text{ in } \beta\text{-nf}}{MN \rightharpoonup_n MN'}
\]

If \( M \rightharpoonup_n^* N \), and the reductions aren’t all head, then there must be a first head normal form \( P \), such that

\[
M \xrightarrow{h}^* P \xrightarrow{i}^* N
\]

We want to show the same sort of split for arbitrary \( \beta \)-reduction (\( \rightharpoonup^*_\beta \))
Interlude: Weak Head Reduction

Thanks to:

\[
\begin{align*}
M & \xrightarrow{h} M' \\
(\lambda v. M) & \xrightarrow{h} (\lambda v. M')
\end{align*}
\]

head reduction proceeds inside function bodies.

If you take this rule out, you get weak head reduction.

Weak head normal forms are

- head normal forms, or
- abstractions.

Weak head reduction is used in implementations of functional programming languages.
Basic Strategy

We want to know that, if by some path:

$$M \xrightarrow{\beta}^* N$$

with $N$ a normal form, then normal order reduction will take $M$ to $N$ too.

Will do this by showing a more general result.

That for any $N$, if $M \xrightarrow{\beta}^* N$, then there exists a $P$ such that

$$M \xrightarrow{h}^* P \xrightarrow{i}^* N$$
We Want to Commute Steps

If we had

\[ \text{M} \ni \text{N} \ni \text{P} \]

we'd like to know that there was a \( \text{N}' \) we could get to via head reduction, and from which we could make internal reductions to get to \( \text{P} \) (maybe with multiple steps?).

Maybe this would allow head and internal steps to be "bubble-sorted" so that all head steps came first.
We Want to Commute Steps

If we had

\[ M \]  
\[ N \]  
\[ N' \]  
\[ P \]

we’d like to know that there was a \( N' \) that we could get to via head reduction, and from which we could make internal reductions to get to \( P \).
We Want to Commute Steps

If we had

\[
\begin{align*}
N & \quad M \\
\downarrow h & \quad \downarrow h' \\
N' & \quad P \\
\end{align*}
\]

we’d like to know that there was a \( N' \) that we could get to via head reduction, and from which we could make internal reductions to get to \( P \) (maybe with multiple steps?).
We Want to Commute Steps

If we had

we’d like to know that there was a \( N' \) that we could get to via head reduction, and from which we could make internal reductions to get to \( P \) (maybe with multiple steps?).

Maybe this would allow head and internal steps to be “bubble-sorted” so that all head steps came first.
But Direct Commuting is Hard

Commuting does require multiple steps:

\[(\lambda x. f x x) \ ((\lambda y. y z) \ u)\]

This example requires multiple (2) internal reductions.
But Direct Commuting is Hard

Commuting does require multiple steps:

\[
(\lambda u. (\lambda v. v \ u \ z) \ f) \ N
\]

\[
(\lambda u. f \ u \ z) \ N
\]  \[\text{h} \]
\[
f \ N \ z
\]

\[
(\lambda v. v \ N \ z) \ f
\]  \[\text{h} \]
\[
f \ N \ z
\]

This example requires multiple head reductions (and no internals).
Commuting with Multiple Steps Isn’t Good Enough

This is a theorem:

\[ M \xrightarrow{i} N \xrightarrow{h} P \]

\[ \implies \exists N'. M \xrightarrow{h,*} N' \xrightarrow{i,*} P \]

But it’s not good enough.

Our examples show us that we can have

\[ i h \rightarrow h i^2 \]

\[ i h \rightarrow h^2 \]
The Bubble-Sort That Never Ends

Our examples show us that we can have

\[ ih \rightarrow hi^2 \]
\[ ih \rightarrow h^2 \]

Start with a reduction sequence \( ihihi \):

\[ ihihi \rightarrow hi^3hi \]
\[ \rightarrow hi^2h^2i \]
\[ \rightarrow hihi^2hi \]
\[ \rightarrow \ldots \]

This is not progress: we still have two head reductions that haven’t been “sorted” to the start of the sequence.
A Better Lemma

Though commuting internal and head reductions can result in multiple internal reductions, the latter are all parallel (write $\rightarrow^i$).

So, prove instead:

If a internal parallel reduction is followed by a head reduction,
A Better Lemma

Though commuting internal and head reductions can result in multiple internal reductions, the latter are all parallel (write \( \Rightarrow^i \)).

So, prove instead:

If a internal parallel reduction is followed by a head reduction, there is an alternative route where head reductions come first, and there is one internal parallel reduction afterwards.
Proof in More Detail

Have

\[ M \xrightarrow{i} P \xrightarrow{h} N \]

As \( P \) head-reduces it is \( (\lambda \vec{v}. (\lambda u. P_0) P_1 P_2 \cdots P_n) \), with \( n \geq 1 \)

And \( M \) is of form \( (\lambda \vec{v}. (\lambda u. M_0) M_1 M_2 \cdots M_n) \), with \( M_i \xrightarrow{\beta} P_i \)
Proof in More Detail

Have

\[ M \overset{i}{\Rightarrow} P \overset{h}{\Rightarrow} N \]

As \( P \) head-reduces it is \((\lambda \vec{v}. (\lambda u. P_0) P_1 P_2 \cdots P_n)\), with \( n \geq 1 \)

And \( M \) is of form \((\lambda \vec{v}. (\lambda u. M_0) M_1 M_2 \cdots M_n)\), with \( M_i \overset{\beta}{\Rightarrow} P_i \)

So

\[ M \overset{h}{\Rightarrow} (\lambda \vec{v}. M_0[u := M_1] M_2 \cdots M_n) \]
Proof in More Detail

Have

\[ M \xrightarrow{i} P \xrightarrow{h} N \]

As \( P \) head-reduces it is \( (\lambda \bar{v}. (\lambda u. P_0) P_1 P_2 \cdots P_n) \), with \( n \geq 1 \)

And \( M \) is of form \( (\lambda \bar{v}. (\lambda u. M_0) M_1 M_2 \cdots M_n) \), with \( M_i \xrightarrow{\beta} P_i \)

So

\[ M \xrightarrow{h} (\lambda \bar{v}. M_0[u := M_1] M_2 \cdots M_n) \]

\[ \xrightarrow{\beta} N \]

Last transition is \( \xrightarrow{\beta} \), not \( \xrightarrow{i} \) because \( M_0 \)'s reduction may be at top level, making it head.

A little more work is still required

(decomposing \( \xrightarrow{\beta} \) into head and internal parts).
The Last Big Lemma

If $M \xrightarrow{\beta} N$, then there are $M_i$ such that

$$M \xrightarrow{h} M_1 \xrightarrow{h} M_2 \cdots \xrightarrow{h} M_n \xrightarrow{i} N$$

and each

$$M_i \xrightarrow{\beta} N$$

This gives:
Putting the Lemmas Together

Proving: $M \xrightarrow{\beta}^* N \implies \exists P. M \xrightarrow{h}^* P \xrightarrow{i}^* N$

Have $M \xrightarrow{\beta}^* N$, and so also $M \xrightarrow{\beta}^* N$.

(Base case of zero steps trivial.)
Putting the Lemmas Together

Proving: \( M \xrightarrow{\beta}^* N \implies \exists P. \ M \xrightarrow{h}^* P \xrightarrow{i}^* N \)

Have \( M \xrightarrow{\beta}^* N \), and so also \( M \xrightarrow{\beta}^* N \).

So, assume \( M \xrightarrow{\beta} M' \xrightarrow{\beta}^* N \).
Putting the Lemmas Together

Proving: $M \xrightarrow{\beta}^* N \implies \exists P. M \xrightarrow{h}^* P \xrightarrow{i}^* N$

Have $M \xrightarrow{\beta}^* N$, and so also $M \xrightarrow{\beta}^* N$.

So, assume $M \xrightarrow{\beta} M' \xrightarrow{\beta}^* N$.

By Last Big Lemma, also have $P_1$ s.t. $M \xrightarrow{h}^* P_1 \xrightarrow{i} M'$
Putting the Lemmas Together

Proving: \( M \xrightarrow{\beta}^* N \implies \exists P. M \xrightarrow{h}^* P \xrightarrow{i}^* N \)

Have \( M \xrightarrow{\beta}^* N \), and so also \( M \implies \beta N \).

So, assume \( M \implies \beta M' \implies \beta N \).

By Last Big Lemma, also have \( P_1 \) s.t. \( M \xrightarrow{h}^* P_1 \xrightarrow{i} M' \)

By inductive hypothesis, have \( P_2 \) s.t. \( M' \xrightarrow{h}^* P_2 \xrightarrow{i} N \)

I.e.,

\[
M \xrightarrow{h}^* P_1 \xrightarrow{i} M' \xrightarrow{h}^* P_2 \xrightarrow{i} N
\]

The Proof

The Right Approach
Putting the Lemmas Together

Proving: \( M \xrightarrow{\beta}^* N \implies \exists P. M \xrightarrow{h}^* P \xrightarrow{i}^* N \)

Have \( M \xrightarrow{\beta}^* N \), and so also \( M \xrightarrow{\beta}^* N \).

So, assume \( M \xrightarrow{\beta} M' \xrightarrow{\beta}^* N \).

By Last Big Lemma, also have \( P_1 \) s.t. \( M \xrightarrow{h}^* P_1 \xrightarrow{i} M' \)

By inductive hypothesis, have \( P_2 \) s.t. \( M' \xrightarrow{h}^* P_2 \xrightarrow{i}^* N \)

I.e.,
\[
M \xrightarrow{h}^* P_1 \xrightarrow{i} M' \xrightarrow{h}^* P_2 \xrightarrow{i}^* N
\]

Now, we can “bubble” head reductions after \( M' \) up over the \( \xrightarrow{i} \), using the Better Lemma.
Consequences: Standardisation

Have shown: $M \xrightarrow{\beta}^* N \implies \exists P. M \xrightarrow{h}^* P \xrightarrow{i}^* N$

It’s obviously possible to order the internal reductions so that they occur left-to-right.

- By induction.
  The internal terms within $N$ are all smaller than $N$ itself, so the internal reductions within each $N_i$ can themselves be sorted appropriately.

Gives **Standardisation**: 

If $M \xrightarrow{\beta}^* N$ is possible, then $N$ can be reached from $M$ in a “standard” way (doing reductions in left to right order)
Recall that normal order evaluation is a “standard” evaluation strategy that does all possible reductions.
Recall that normal order evaluation is a “standard” evaluation strategy that does all possible reductions.

If $M$ can reduce to $N$, a $\beta$-normal form, then there is a standard reduction that does the same.
Recall that normal order evaluation is a “standard” evaluation strategy that does all possible reductions.

If $M$ can reduce to $N$, a $\beta$-normal form, then there is a standard reduction that does the same.

If a standard reduction terminates in a $\beta$-normal form, it has done all possible reductions.
Recall that normal order evaluation is a “standard” evaluation strategy that does all possible reductions.

If $M$ can reduce to $N$, a $\beta$-normal form, then there is a standard reduction that does the same.

If a standard reduction terminates in a $\beta$-normal form, it has done all possible reductions.

And so that standard reduction was a normal order reduction.
Recall that normal order evaluation is a “standard” evaluation strategy that does all possible reductions.

If M can reduce to N, a $\beta$-normal form, then there is a standard reduction that does the same.

If a standard reduction terminates in a $\beta$-normal form, it has done all possible reductions.

And so that standard reduction was a normal order reduction.

So, normal order evaluation finds normal forms if they exist.
Summary

An involved proof.

**Lesson #1:** The $\lambda$-calculus is a plausible programming language
- there is an algorithm for turning $\lambda$-terms into values
- (when those terms have values at all)

**Lesson #2:** Evaluation Orders are a Design Question
- Do you want to guarantee as much termination as possible?
  - Use normal order
- Or, do you want more speed, and unnecessary non-terminations?
  - Use applicative order (like C, Java etc)

Next time: adding plausibility.
Numbers, Pairs and Lists for the $\lambda$-calculus.