M2 LMFI

Proofs and programs: advanced topics

Linear Logic and Quantitative Semantics

Teachers:

Claudia Faggian **CNRS (IRIF)**
faggian@irif.fr
https://www.irif.fr/~faggian/

Gabriele Vanoni

INSTITUT DE RECHERCHE EN INFORMATIQUE FONDAMENTALE

1

Organization

- Lectures:
  Wednesday 14h00-16h00
  Friday 14h00-16h00

- **Grading:**
  - **weekly homework** projects

2

Plan

A foundational study of functional programming languages,

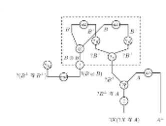
- building on:
  - *proof theory* (Types, Curry-Howard isomorphism) and
  - the theory of *lambda-calculus,*
- adopting *the **dynamic and quantitative view brought by Linear Logic.***
  - Focus first part: a **quantitative view in Operational Semantics**
  - Focus second prat: a **quantitative view in Denotational Semantics**
  - Openings towards active research topics: Bayesian learning/ probabilistic programming, ….
- Courses from **LMFI** first term we build on:
  - *Proof Theory (cut-elimination, lambda calculus,* Curry-Howard *iso)*
- Connected to the **MPRI** course: Semantics of Programming Language (which builds on the models of Linear Logic)

3

Linear Logic [Girard87] breakthroughs

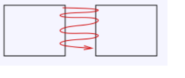
New insights into **proof theory** and
(via the **Curry-Howard correspondence** between proofs and programs)
into the **semantics of programming language**.

- **Proof Nets:** advanced formal system



- representation of proofs (λ-terms, functional programs) by graphs
- tool for the analysis of cut-elimination (= execution) as graph-rewriting process

- Dynamic view, capturing the **flow of computation**:



  - **Game Semantics**
  - **Geometry of Interaction**

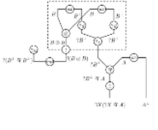
4

Linear Logic [Girard87] breakthroughs

New insights into **proof** …
(via the **Curry-Howard co**…
into the **semantics of pro**…

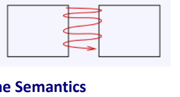
especially suitable for
- Cost analysis (runtime/memory space/ other resources)
- modelling probabilistic & quantum programming

- **Proof Nets:** advanced formal system



- representation of proofs (λ-terms, functional programs) by graphs
- tool for the analysis of cut-elimination (= execution) as graph-rewriting process

- Dynamic view, capturing the **flow of computation**:

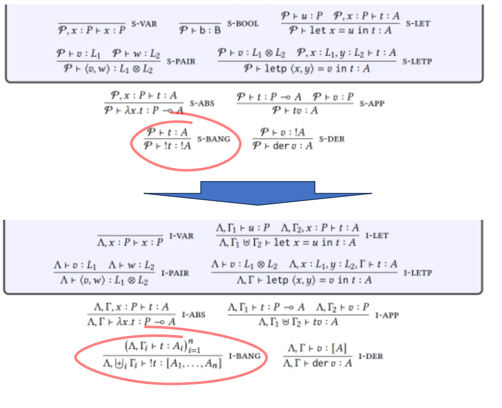


  - **Game Semantics**
  - **Geometry of Interaction**

- Account for resources
  - Quantitative Semantics
  - Quantitative Type Systems

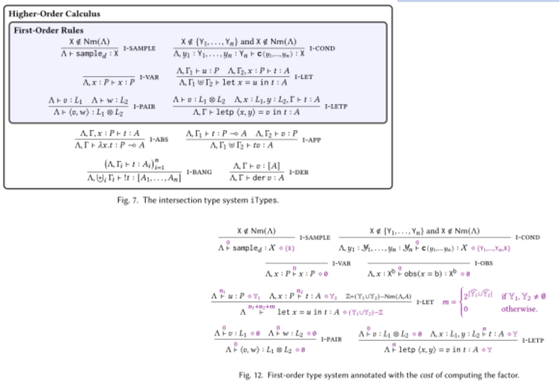
5

Resource awareness (Quantitative Types)



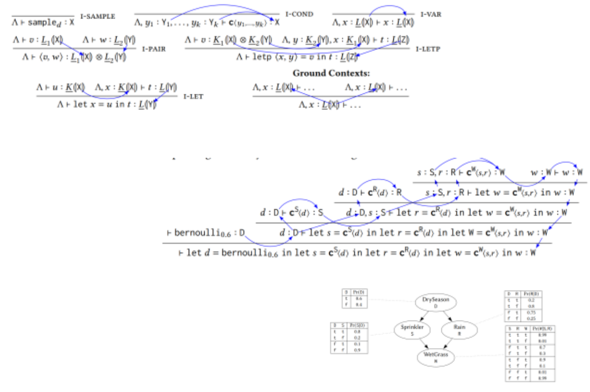
6

Higher-Order Bayesian Networks



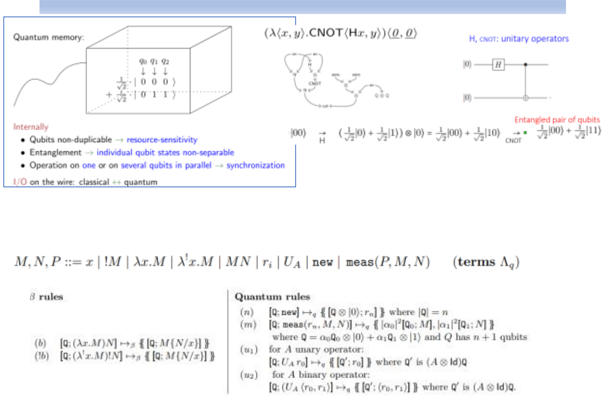
7

Higher-Order Bayesian Networks

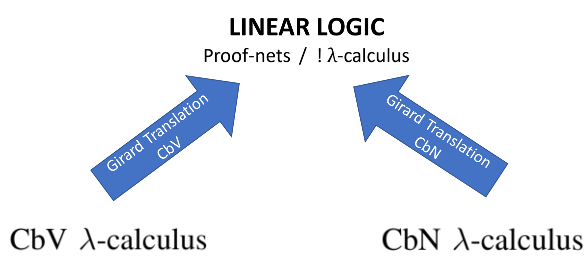


8

Quantum lambda calculus



9



10

Plan / topics for Part 1 HANDS-ON

- Theoretical tools to study the operational properties of a system:
  - Rewrite Theory (rewriting=abstract form of program execution)
- Linear Logic and Proof-Nets.
- Bridging between lambda-calculus and functional programming:
  - Call-by-Value and Call-by Name, weak and lazy calculi.
- Beyond pure functional:
  - Probabilistic programming and Bayesian Inference:
    Probabilistic lambda calculi, Bayesian proof-nets

  (Internships possible on operational aspects of probabilistic and quantum computation)

11

Resources

- **Webpage** https://www.irif.fr/~faggian/LMFI2025
- **Lecture Notes** (by A. Middeldorp, O. Laurent, L. Ong)

12

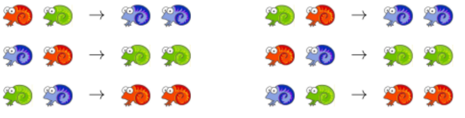
Operational semantics
of formal calculi and programming languages

Rewriting theory

- **Rewriting = abstract form of program execution**
- Paradigmatic example: **λ-calculus**
  (functional programming language, in its essence)

13

A colony of chameleons includes 20 red, 18 blue, and 16 green individuals. Whenever two chameleons of different color meet, each changes to the third color. Some time passes during which no chameleons are born or die nor do any enter or leave the colony. Is it possible that at the end of this period, all 54 chameleons are the same color?



14

Math formalizations...

Example (Group Theory)

signature	e (constant)	$^-$ (unary, postfix)	$\cdot$ (binary, infix)	
equations	$e \cdot x \approx x$	$x^- \cdot x \approx e$	$(x \cdot y) \cdot z \approx x \cdot (y \cdot z)$	$\mathcal{E}$
theorems	$e^- \approx_{\mathcal{E}} e$	$(x \cdot y)^- \approx_{\mathcal{E}} y^- \cdot x^-$		
rewrite rules	$e \cdot x \to x$ $x^- \cdot x \to e$ $(x \cdot y) \cdot z \to x \cdot (y \cdot z)$ $e^- \to e$ $x^- \cdot (x \cdot y) \to y$	$x \cdot e \to x$ $x \cdot x^- \to e$ $x^{--} \to x$ $(x \cdot y)^- \to y^- \cdot x^-$ $x \cdot (x^- \cdot y) \to y$		$\mathcal{R}$

① $s \approx t$ is valid in $\mathcal{E}$ ($s \approx_{\mathcal{E}} t$) if and only if s and t have same $\mathcal{R}$-normal form

② $\mathcal{R}$ admits no infinite computations

① & ② $\Longrightarrow$ $\mathcal{E}$ has decidable validity problem

15

Modelling computation

Example (Lambda Calculus)

signature	λ (binds variables) $\cdot$ (application, binary, infix)
terms	$M ::= x \mid (\lambda x.\, M) \mid (M \cdot M)$
α conversion	$\lambda x.\, x \cdot y =_\alpha \lambda z.\, z \cdot y$
β reduction	$(\lambda x.\, M) \cdot N \to_\beta M[x := N]$ replace free occurrences of x in M by N (and avoid variable capturing)
rewriting	$(\lambda x.\, x \cdot x) \cdot (\lambda x.\, x \cdot x) \to (\lambda x.\, x \cdot x) \cdot (\lambda x.\, x \cdot x)$
inventor	Alonzo Church (1932)

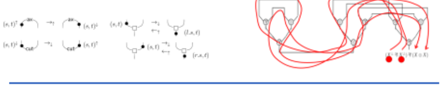
both Combinatory Logic and Lambda Calculus are Turing-complete

17

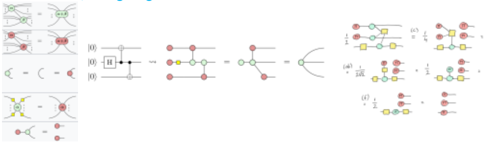
Graph Rewriting

LL proof-nets

Geometry of Interaction



ZX- calculus, string diagrams..



18

Rewriting

- Rewrite Theory provides a powerful set of tools to study computational and operational properties of a system : normalization, termination , confluence, uniqueness of normal forms
- tools to study and compare strategies:
  - Is there a strategy guaranteed to lead to normal form, if any (*normalizing strat.*)?
- Abstract Rewrite Systems (ARS) capture the common substratum of rewrite theory (independently from the particular structure of terms) - can be uses in the study of any calculus or programming language.

19

Abstract Rewriting: motivations

concrete rewrite formalisms / concrete operational semantics:

- λ-calculus
- *Quantum/ probabilistic/ non-deterministic/………..* λ-calculus
- Proof-nets / graph rewriting
- Sequent calculus and cut-elimination
- string rewriting
- term rewriting

abstract rewriting

- **independent from structure** of objects that are rewritten
- **uniform** presentation of properties and proofs

20

Why a theory of rewriting matters?

- **Rewriting = abstract form of program execution**

Rewriting theory provides a *sound framework* for reasoning about
- **programs transformations**, such as compiler optimizations or parallel implementations,
- **program equivalence**.

21

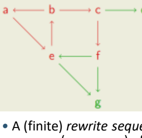
Abstract Rewriting

Basic language

22

ARS

Definition 1.1.1. An *abstract rewrite system* (ARS for short) is a pair $\mathcal{A} = \langle A, \rightarrow \rangle$ consisting of a set A and a binary relation $\rightarrow$ on A. Instead of $(a, b) \in \rightarrow$ we write $a \rightarrow b$ and we say that $a \rightarrow b$ is a *rewrite step*.



ARS $\mathcal{A} = \langle A, \rightarrow \rangle$
- $A = \{ a, b, c, d, e, f, g \}$
- $\rightarrow = \left\{ \begin{array}{l} (a,e), (b,a), (b,c), (c,d), (c,f) \\ (e,b), (e,g), (f,e), (f,g) \end{array} \right\}$

- A (finite) *rewrite sequence* is a non-empty sequence (a_0, … a_n) of elements in *A* such that $a_i \rightarrow a_{\{i+1\}}$

We write $a_0 \rightarrow^n a_n$ or simply $a_0 \rightarrow^* a_n$

- rewrite sequence
  - finite $a \rightarrow e \rightarrow b \rightarrow c \rightarrow f$
  - empty a
  - infinite $a \rightarrow e \rightarrow b \rightarrow a \rightarrow e \rightarrow b \rightarrow \cdots$

23

- $\leftarrow$ inverse of $\rightarrow$
- $\rightarrow^*$ transitive and reflexive closure of $\rightarrow$
- ${}^*\leftarrow$ inverse of $\rightarrow^*$

$s \leftrightarrow_{\mathcal{R}} t$ iff $s \rightarrow_{\mathcal{R}} t$ or $t \rightarrow_{\mathcal{R}} s$

$s \leftrightarrow^*_{\mathcal{R}} t$ iff $s = s_0 \leftrightarrow_{\mathcal{R}} s_1 \leftrightarrow_{\mathcal{R}} \ldots \leftrightarrow_{\mathcal{R}} s_n = t$ for $n \geq 0$

- $\leftrightarrow$ symmetric closure of $\rightarrow$
- $\leftrightarrow^*$ conversion (equivalence relation generated by $\rightarrow$) **
- $\rightarrow^+$ transitive closure of $\rightarrow$
- $\rightarrow^=$ reflexive closure of $\rightarrow$

$\cdot$ is relation composition: $R \cdot S = \{ (a, c) \mid a\ R\ b \text{ and } b\ S\ c \}$

$\downarrow = \rightarrow^* \cdot {}^*\leftarrow$

24

Composition

- If $\rightarrow_1, \rightarrow_2$ are binary relations on A then $\rightarrow_1 \cdot \rightarrow_2$ denotes their composition, *i.e.* $t \rightarrow_1 \cdot \rightarrow_2 s$ iff there exists $u \in A$ such that $t \rightarrow_1 u \rightarrow_2 s$.
- We write $(A, \{\rightarrow_1, \rightarrow_2\})$ to denote the ARS $(A, \rightarrow)$ where $\rightarrow = \rightarrow_1 \cup \rightarrow_2$.

25

Closure

The transitive-reflexive closure of a relation is a closure operator, *i.e.* satisfies

$$\to \subseteq \to^*, \quad (\to^*)^* = \to^*, \quad \to_1 \subseteq \to_2 \text{ implies } \to_1^* \subseteq \to_2^*$$

As a consequence

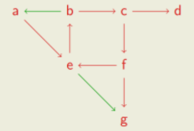
$$(\to_1 \cup \to_2)^* = (\to_1^* \cup \to_2^*)^*.$$

26

Terminology

- if $x \to^* y$ then x rewrites to y and y is reduct of x
- if $x \to^* z \ {}^*\!\leftarrow y$ then z is common reduct of x and y
- if $x \leftrightarrow^* y$ then x and y are convertible

Example

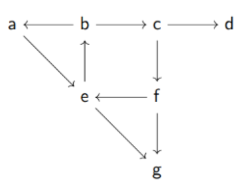


- a $\to^*$ f
- e $\downarrow$ f f $\downarrow$ d not g $\downarrow$ d
- g $\leftrightarrow^*$ d

27

Normal forms model **results**

Definition 1.1.11. Let $\mathcal{A} = \langle A, \to \rangle$ be an ARS. An element $a \in A$ is *reducible* if there exists an element $b \in A$ with $a \to b$. A *normal form* is an element that is not reducible. The set of normal forms of $\mathcal{A}$ is denoted by $\mathsf{NF}(\mathcal{A})$ or $\mathsf{NF}(\to)$ when A can be inferred from the context. An element $a \in A$ *has a normal form* if $a \to^* b$ for some normal form b. In that case we write $a \to^! b$.



Element **a** has normal forms ?
How many normal forms has this ARS?

ARS $\mathcal{A} = \langle A, \to \rangle$

- d is normal form
- $\mathsf{NF}(\mathcal{A}) = \{$ d, g $\}$
- b $\to^!$ g

28

Operational properties of interest

- Termination and Confluence

Existence and uniqueness of normal forms

- How to Compute

reduction strategies with good properties:

- standardization,
- normalization

29

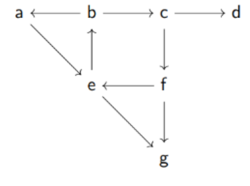
- SN strong normalization termination
  - no infinite rewrite sequences
- WN (weak) normalization
  - every element has at least one normal form
  - $\forall a \, \exists b \quad a \to^! b$
- UN unique normal forms
  - no element has more than one normal form
  - $\forall a, b, c \quad$ if $a \to^! b$ and $a \to^! c$ then $b = c$

30

Termination

Definition 1.2.1. Let $\mathcal{A} = \langle A, \to \rangle$ be an ARS. An element $a \in A$ is called *terminating* or *strongly normalizing* (SN) if there are no infinite rewrite sequences starting at a. The ARS $\mathcal{A}$ is terminating or strongly normalizing if all its elements are terminating. An element $a \in A$ has *unique normal forms* (UN) if it does not have different normal forms ($\forall\, b, c \in A$ if $a \to^! b$ and $a \to^! c$ then $b = c$). The ARS $\mathcal{A}$ has unique normal forms if all its elements have unique normal forms.

An element *a* is *weakly normalizing* (WN) (or simply *normalizing)* if it has a normal form.



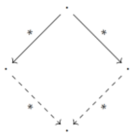
a is WN? SN?
c is WN? SN?
a or c has UN ?

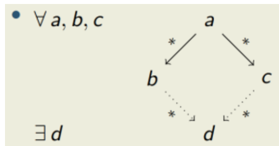
The nf are convertible?

31

Confluence

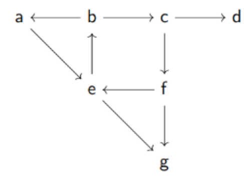
Definition 1.2.3. Let $\mathcal{A} = \langle A, \to \rangle$ be an ARS. An element $a \in A$ is *confluent* if for all elements $b, c \in A$ with $b \; {}^*\!\!\leftarrow a \to^* c$ we have $b \downarrow c$. The ARS $\mathcal{A}$ is confluent if all its elements are confluent.





Every confluent ARS *has unique normal forms.*

32



1. a is confluent?
2. f is confluent?

Bonus Point

3. Can you add a single arrow so that the resulting ARS has **unique normal forms without being confluent ?**

33

Given

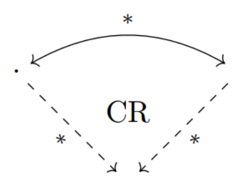
$$\mathcal{R} = \begin{cases} f(x,x) & \to & c \\ a & \to & b \\ f(x,b) & \to & d \end{cases}$$

f(a,a) has normal form?
Can you produce two different nf?

we can compute from the same term $f(a, a)$ two different normal-forms c and d
different meaning for same term!
(also: different meaning for equivalent terms)

34

Same meaning for *equivalent* terms

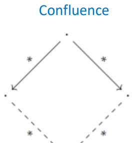


35

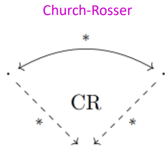
Confluence & CR

Definition 1.2.3. Let $\mathcal{A} = \langle A, \to \rangle$ be an ARS. An element $a \in A$ is *confluent* if for all elements $b, c \in A$ with $b \; {}^*\!\!\leftarrow a \to^* c$ we have $b \downarrow c$. The ARS $\mathcal{A}$ is confluent if all its elements are confluent.

Confluence



Church-Rosser



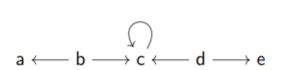
An ARS $\mathcal{A} = \langle A, \to \rangle$ *is confluent if and only if* $\leftrightarrow^* \subseteq \downarrow$.

36

Definition 1.2.10. An ARS $\mathcal{A} = \langle A, \to \rangle$ has *unique normal forms with respect to conversion* (UNC) if different normal forms are not convertible ($\forall a, b \in \mathsf{NF}(\mathcal{A})$ if $a \leftrightarrow^* b$ then $a = b$).

in an ARS with the property UNC every equivalence class of convertible elements contains at most one normal form.

Q: are UN and UNC equivalent?



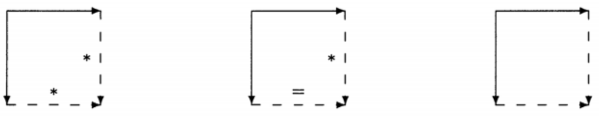
37

Global vs Local

Confluence

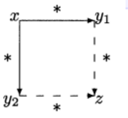
A property of term *t* is *local* if it is quantified over only *one-step reductions* from *t*; it is *global* if it is quantified over all *rewrite sequences* from *t*.

Locally confluent (WCR) **Strongly confluent** **Diamond**



confluence Let $\mathcal{A} = \langle A, \rightarrow \rangle$ be an ARS. An element $a \in A$ is *confluent* if for all elements $b, c \in A$ with $b \; {}^*\!\!\leftarrow a \rightarrow^* c$ we have $b \downarrow c$. The ARS $\mathcal{A}$ is confluent if all its elements are confluent.

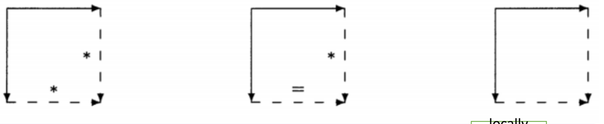
Global property:



Confluence

A property of term *t* is *local* if it is quantified over only *one-step reductions* from *t*; it is *global* if it is quantified over all *rewrite sequences* from *t*.

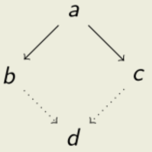
Locally confluent (WCR) **Strongly confluent** **Diamond**



Local confluence Let $\mathcal{A} = \langle A, \rightarrow \rangle$ be an ARS. An element $a \in A$ is locally confluent for all elements $b, c \in A$ with b←a→c we have $b \downarrow c$. The ARS $\mathcal{A}$ is confluent if all its elements are confluent.

An ARS $\mathcal{A} = \langle A, \rightarrow \rangle$ has the *diamond property* ($\diamond$) if $\leftarrow \cdot \rightarrow \; \subseteq \; \rightarrow \cdot \leftarrow$

- diamond property $\diamond$
  - $\leftarrow \cdot \rightarrow \; \subseteq \; \rightarrow \cdot \leftarrow$
  - $\forall a, b, c$



$\exists d$

- *every ARS with diamond property is confluent*

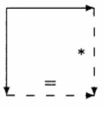
Proof by tiling

An ARS $\mathcal{A} = \langle A, \rightarrow \rangle$ is *strongly confluent* (SCR) if $\leftarrow \cdot \rightarrow \; \subseteq \; \rightarrow^{=} \cdot {}^*\!\!\leftarrow$, see Figure

a Show that every strongly confluent ARS is confluent.

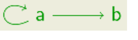
b Does the converse also hold?

c Show that an ARS $\mathcal{A} = \langle A, \rightarrow \rangle$ is confluent if and only if $\leftarrow^* \cdot \rightarrow \; \subseteq \; \rightarrow^* \cdot \leftarrow^*$

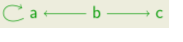


Which is true?

1. SN => WN
2. WN => SN



3. Confluence => UN
4. UN => Confluence



5. Confluence => Local confluence
6. Local confluence => Confluence

7. WN & UN => Confluence
8. WN & Local Conf. => Confluence
9. SN & Local Conf. => Confluence

WN vs SN

↻ a ⟶ b

(ii) WN $\not\Rightarrow$ SN

$$\mathcal{R} = \left\{ \begin{array}{lcl} f(a) & \to & c \\ f(x) & \to & f(a) \end{array} \right.$$

The system is weakly normalising but not strongly normalising:

Can you find an infinite reduction sequence from f(b)?

$$f(b) \to f(a) \to c$$

$$f(b) \to f(a) \to f(a) \ldots$$

44

1. SN => WN
2. WN => SN
3. Confluence => UN
4. UN => Confluence
5. Confluence => Local confluence
6. Local confluence => Confluence
7. WN & UN => Confluence
8. WN & Local Conf. => Confluence
9. SN & Local Conf. => Confluence

↻ a ⟶ b

↻ a ⟵ b ⟶ c

a ⟵ b ⇄ c ⟶ d

Newman's Lemma

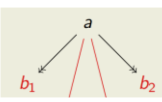
45

Lemma

WN & UN $\Longrightarrow$ CR

Proof

- WN $\Longrightarrow$ $\exists\, n_1, n_2 \colon\; b_1 \to^! n_1$ and $b_2 \to^! n_2$
- UN $\Longrightarrow$ $n_1 = n_2$ $\Longrightarrow$ $b_1 \downarrow b_2$



46

Newman Lemma

Newman's Lemma. *Every terminating and locally confluent ARS is confluent.*

By well-founded induction

47

Memo: Well-founded Induction

given

- property P of ARSs with $\mathsf{P}(\mathcal{A}) \iff \forall\, a \colon \mathsf{P}(a)$
- strongly normalizing ARS $\mathcal{A} = \langle A, \to \rangle$

to conclude

- $\mathsf{P}(\mathcal{A})$

it is sufficient to prove

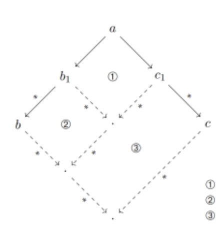
- if $\underbrace{\mathsf{P}(b) \text{ for every } b \text{ with } a \to b}_{\text{induction hypothesis}}$ then $\mathsf{P}(a)$

for arbitrary element a

48

Newman Lemma

Newman's Lemma. *Every terminating and locally confluent ARS is confluent.*



① WCR
② induction hypothesis ($a \to b_1 \implies b_1$ is CR)
③ induction hypothesis ($a \to c_1 \implies c_1$ is CR)

49

Newman Lemma

Bonus Exercise

Newman's Lemma. *Every terminating and locally confluent ARS is confluent.*

A second Proof. Let $\mathcal{A} = \langle A, \to \rangle$ *terminating and locally confluent*

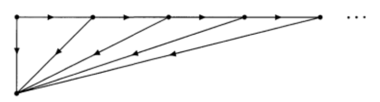
It suffices to show that every element has unique normal forms

- suppose $B = \{ a \in A \mid \neg \mathrm{UN}(a) \} \neq \varnothing$
- let $b \in B$ be minimal element (with respect to $\to$)
- $b \to^! n_1$ and $b \to^! n_2$ with $n_1 \neq n_2$

➢ **Conclude** by showing that it is impossible (**absurd**)

50

Recap Flash Ex



➢ **EX** Say which properties hold

1. Confluent
2. Locally confluent
3. Normalizing (weakly normalizing, WN)
4. Terminating (strongly normalizing, SN)

51

Recap basics

- An *abstract rewriting system (ARS)* is a pair $(\mathcal{A}, \to)$ consisting of a set $\mathcal{A}$ and a binary relation $\to$ on $\mathcal{A}$ whose pairs are written $t \to s$ and called *steps*.
- We denote $\to^*$ (resp. $\to^=$) the transitive-reflexive (resp. reflexive) closure of $\to$. We write $t \leftarrow u$ if $u \to t$.
- If $\to_1, \to_2$ are binary relations on $\mathcal{A}$ then $\to_1 \cdot \to_2$ denotes their composition, *i.e.* $t \to_1 \cdot \to_2 s$ if there exists $u \in \mathcal{A}$ such that $t \to_1 u \to_2 s$.
- We write $(\mathcal{A}, \{\to_1, \to_2\})$ to denote the *compound system* $(\mathcal{A}, \to)$ where $\to = \to_1 \cup \to_2$.

- A $\to$-*sequence* (or **reduction sequence**) from t is a (possibly infinite) sequence $t, t_1, t_2, \ldots$ such that $t_i \to t_{i+1}$.
  $t \to^* s$ indicates that there is a finite sequence from t to s.
  A $\to$-sequence from t is *maximal* if it is either infinite or ends in a $\to$-*nf*.

52

The heart of confluence is a diamond

Prop. DIAMOND implies CONFLUENCE

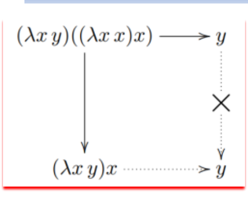
*Can rarely be used **directly**:*
Most relations of interest do not satisfy it

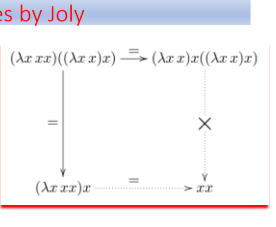
Lemma **(Characterize Confluence).** $\to$ *is confluent* if and only if *there exists a relation* $\Rrightarrow$ *such that*

a. $\Rrightarrow^* = \to^*$,
b. $\Rrightarrow$ *is diamond.*

53

You have already seen an example: in the notes by Joly





Definition *The* development relation *is the least reflexive relation* $\rhd$ *on* Λ *such that:*

- $t \rhd t' \implies \lambda x\, t \rhd \lambda x\, t'$
- $t \rhd t',\ u \rhd u' \implies tu \rhd t'u'$
- $t \rhd t',\ u \rhd u' \implies (\lambda x\, t)u \rhd t'[x := u']$.

Lemma 1 $\to\ \subseteq\ \rhd\ \subseteq\ \twoheadrightarrow$.

54

You have already seen an example: in the notes by Joly

Lemma 3 (Characterize Confluence). $\to$ *is confluent* if and only if *there exists a relation* $\Rrightarrow$ *such that*

a. $\Rrightarrow^* = \to^*$,
b. $\Rrightarrow$ *is diamond.*

Definition *The* development relation *is the least reflexive relation* $\rhd$ *on* Λ *such that:*

- $t \rhd t' \implies \lambda x\, t \rhd \lambda x\, t'$
- $t \rhd t',\ u \rhd u' \implies tu \rhd t'u'$
- $t \rhd t',\ u \rhd u' \implies (\lambda x\, t)u \rhd t'[x := u']$.

Lemma 1 $\to\ \subseteq\ \rhd\ \subseteq\ \twoheadrightarrow$.

55

Closure

$\twoheadrightarrow_R$ is the reflexive, transitive closure of $\to_R$:
(1) $M \to_R N \Rightarrow M \twoheadrightarrow_R N$,
(2) $M \twoheadrightarrow_R M$,
(3) $M \twoheadrightarrow_R N,\ N \twoheadrightarrow_R L \Rightarrow M \twoheadrightarrow_R L$.

The transitive-reflexive closure of a relation is a closure operator, *i.e.* satisfies

$$\to \subseteq \to^*, \qquad (\to^*)^* = \to^*, \qquad \to_1 \subseteq \to_2 \text{ implies } \to_1^* \subseteq \to_2^*$$

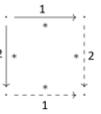
As a consequence

$$(\to_1 \cup \to_2)^* = (\to_1^* \cup \to_2^*)^*.$$

56

Commutation

Commutation. Two relations $\to_1$ and $\to_2$ on A *commute* if $\leftarrow_1{}^* \cdot \to_2{}^* \subseteq \to_2{}^* \cdot \leftarrow_1{}^*$.



Confluence. A relation $\to$ on A is confluent if it commutes with itself.

57

Proving confluence modularly

Lemma (Hindley-Rosen)

*If two relations $\to_1$ and $\to_2$ are **confluent** and **commute with each other**, then*

$\to_1 \cup \to_2$ is confluent.

58

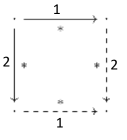
An effective usable technique

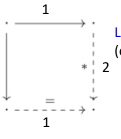
Lemma (Hindley-Rosen)

*If two relations $\to_1$ and $\to_2$ are **confluent** and **commute with each other**, then*

$\to_1 \cup \to_2$ is confluent.

Global condition
(all sequences)





Local condition
(one-step test)

Lemma (Hindley's local test)

Strong commutation $\leftarrow_1 \cdot \to_2 \subseteq \to_2{}^* \cdot \leftarrow_1{}^=$ implies commutation.

59

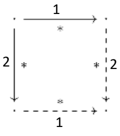
an effective usable technique

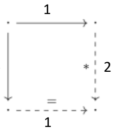
Lemma (Hindley-Rosen)

*If two relations $\to_1$ and $\to_2$ are **confluent** and **commute with each other**, then*

$\to_1 \cup \to_2$ is confluent.

Global condition
(all sequences)



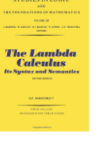


Local condition
(one-step test)

$$\leftarrow_1 \cdot \to_2 \subseteq \to_2{}^* \cdot \leftarrow_1{}^= \qquad \textbf{(Strong Commutation)}$$

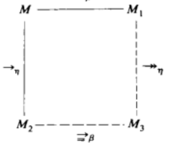
▶ **Lemma** (Local test). *Strong commutation implies commutation.*

60



3.3.8. Lemma. $\twoheadrightarrow_\beta$ *commutes with* $\twoheadrightarrow_\eta$.

Proof. By lemma 3.3.6 it suffices to show



3.3.9. Theorem (*Church–Rosser theorem for βη-reduction*).
(i) *The notion of reduction* **βη** *is* CR.

61

Operational properties of interest

- Termination and Confluence

Existence and uniqueness of normal forms

- How to Compute

reduction strategies with good properties:

- standardization,
- normalization

Strategies

Normalization

▶ **Def.** $(\mathcal{A},\to)$ *is strongly (weakly, uniformly) normalizing if each* $t\in\mathcal{A}$ *is, where the three normalization notions are as follows.*

- t *is* strongly $\to$-normalizing*: every maximal* $\to$*-sequence from* t *ends in a normal form.*
- t *is* weakly $\to$-normalizing*: there exist a* $\to$*-sequence from* t *which ends in a normal form.*
- t *is* uniformly $\to$-normalizing*:* t *weakly* $\to$*-normalizing implies* t *strongly* $\to$*-normalizing.*

If terms are not strongly normalizing, how do we compute a normal form, or even test if any exists? This is the problem tackled by *normalization*. By repeatedly performing *only specific steps* $\underset{e}{\to}$, we are guaranteed that a normal form will eventually be computed, if any exists.

Normalizing strategis

▶ **Def.** $(\mathcal{A},\to)$ *is strongly (weakly, uniformly) normalizing if each* $t\in\mathcal{A}$ *is, where the three normalization notions are as follows.*

- t *is* strongly $\to$-normalizing*: every maximal* $\to$*-sequence from* t *ends in a normal form.*
- t *is* weakly $\to$-normalizing*: there exist a* $\to$*-sequence from* t *which ends in a normal form.*
- t *is* uniformly $\to$-normalizing*:* t *weakly* $\to$*-normalizing implies* t *strongly* $\to$*-normalizing.*

▶ **Def.**

- $\underset{e}{\to}$ *is a* ***strategy for*** $\to$ *if* $\underset{e}{\to}\subseteq\to$*, and it has the same normal forms as* $\to$*.*
- *It is a* ***normalizing strategy*** *for* $\to$ *if whenever* $t\in\mathcal{A}$ *has* $\to$*-normal form, then every maximal* $\underset{e}{\to}$*-sequence from* t *ends in normal form.*

Factorization

(*aka weak Standardization*)

another commutation!

Factorization

(aka Semi-Standardization, Postponement, or often simply Standardization)

- most basic property about *how to compute*

$$t \to_\beta^* u \quad\Rightarrow\quad t \underset{h}{\to}^* \cdot \underset{\neg h}{\to}^* u \qquad \text{head factorization}$$

A **key building-block** in proofs of more sophisticated *how-to-compute* properties:

- allows immediate proofs of **normalization** (a reduction strategy reaches a normal form, whenever one exists)
- simplest way to prove **standardization**, by using Mitschke's argument *(left-to-right standardization = iterate head factorization)*

Factorization
(aka Semi-Standardization, Postponement, or often simply Standardization)

Melliès 97:
the **meaning of *factorization*** is that the *essential* part of a computation can always be separated from its junk.

Assume computations consists of
- steps $\underset{e}{\to}$ which are in some sense *essential*, and
- steps $\underset{i}{\to}$ which are not.

Factorization says that every rewrite sequence can be reorganized/factorized as a sequence of essential steps followed by inessential ones.

$$t \to^* u \quad \Rightarrow \quad t \underset{e}{\to}^* \cdot \underset{i}{\to}^* u \qquad \text{e-factorization}$$

70

Factorization. Let $\mathcal{A} = (A, \{\underset{e}{\to}, \underset{i}{\to}\})$ be an ARS.
- The relation $\to = \underset{e}{\to} \cup \underset{i}{\to}$ satisfies **e-factorization**, written $\mathtt{Fact}(\underset{e}{\to}, \underset{i}{\to})$, if

$$\mathtt{Fact}(\underset{e}{\to}, \underset{i}{\to}): \quad (\underset{e}{\to} \cup \underset{i}{\to})^* \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^* \qquad \textbf{(Factorization)}$$

- The relation $\underset{i}{\to}$ **postpones** after $\underset{e}{\to}$, written $\mathtt{PP}(\underset{e}{\to}, \underset{i}{\to})$, if

$$\mathtt{PP}(\underset{e}{\to}, \underset{i}{\to}): \quad \underset{i}{\to}^* \cdot \underset{e}{\to}^* \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^*. \qquad \textbf{(Postponement)}$$

▶ **Lemma.** *For any two relations $\underset{e}{\to}, \underset{i}{\to}$ the following are equivalent:*
1. $\underset{i}{\to}^* \cdot \underset{e}{\to} \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^*$
2. $\underset{i}{\to} \cdot \underset{e}{\to}^* \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^*$)
3. Postponement: $\underset{i}{\to}^* \cdot \underset{e}{\to}^* \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^*$
4. Factorization: $(\underset{e}{\to} \cup \underset{i}{\to})^* \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^*$

71

Local test ?

We say that $\underset{i}{\to}$ **strongly postpones** after $\underset{e}{\to}$, if

$$\mathtt{SP}(\underset{e}{\to}, \underset{i}{\to}): \quad \underset{i}{\to} \cdot \underset{e}{\to} \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^= \qquad \textbf{(Strong Postponement)}$$

▶ **Lemma** (Local test for postponement [26]). *Strong postponement implies postponement:*

$$\mathtt{SP}(\underset{e}{\to}, \underset{i}{\to}) \textit{ implies } \mathtt{PP}(\underset{e}{\to}, \underset{i}{\to}), \textit{ and so } \mathtt{Fact}(\underset{e}{\to}, \underset{i}{\to}).$$

72

Does SP hold for λ-calculus?

▶ **Ex** (λ-calculus and strong postponement). *β reduction is decomposed in head reduction $\underset{h}{\to}_\beta$ and its dual $\underset{\neg h}{\to}_\beta$*

$$\to_\beta = \underset{h}{\to}_\beta \cup \underset{\neg h}{\to}_\beta$$

Consider:

$$(\lambda x.xxx)(Iz) \underset{\neg h}{\to}_\beta (\lambda x.xxx)z \underset{h}{\to}_\beta zzz.$$

73

Does SP hold for λ-calculus?

▶ **Ex** (λ-calculus and strong postponement). *β reduction is decomposed in head reduction $\underset{h}{\to}_\beta$ and its dual $\underset{\neg h}{\to}_\beta$*

$$\to_\beta = \underset{h}{\to}_\beta \cup \underset{\neg h}{\to}_\beta$$

Consider:

$$(\lambda x.xxx)(Iz) \underset{\neg h}{\to}_\beta (\lambda x.xxx)z \underset{h}{\to}_\beta zzz.$$

$$(\lambda x.xxx)(Iz) \underset{h}{\to}_\beta (Iz)(Iz)(Iz) \underset{h}{\to}_\beta z(Iz)(Iz) \underset{\neg h}{\to}_\beta zz(Iz) \underset{\neg h}{\to}_\beta zzz$$

74

The heart of confluence is a diamond

TRICK

Prop. DIAMOND implies CONFLUENCE

*Can rarely be used **directly**:*
Most relations of interest do not satisfy it

Lemma **(Characterize Confluence).** $\to$ *is confluent* if and only if *there exists a relation $\circ\!\!\!\to$ such that*

a. $\circ\!\!\!\to^* = \to^*$,
b. $\circ\!\!\!\to$ *is diamond.*

75

postponement trick

▶ **Property 2** (Criterion). *Given* $\to \,=\, \underset{e}{\to} \cup \underset{i}{\to}$, *e-factorization holds*

$$\to^* \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^*$$

iff exists $\underset{i}{\oplus\!\!\!\rightarrow}$

- $\underset{i}{\oplus\!\!\!\rightarrow}^* = \underset{i}{\to}^*$ *(same closure)*
- $\underset{i}{\oplus\!\!\!\rightarrow} \cdot \underset{e}{\to} \subseteq \underset{e}{\to}^* \cdot \underset{i}{\oplus\!\!\!\rightarrow}^=$ (strong postponement)

Hence: $\underset{i}{\to}^* \cdot \underset{e}{\to}^* \subseteq \underset{e}{\to}^* \cdot \underset{i}{\to}^*.$ **(Postponement)**

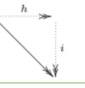
76

Concretely: CbN and Head Factorization

MEMO from last semester

You have already seen this! T. Joly, page 119

We now want to prove that if $t \twoheadrightarrow t'$ then there is u such that $t \twoheadrightarrow_h u \twoheadrightarrow_i t'$:



- $\underset{i}{\oplus\!\!\!\rightarrow}^* = \underset{i}{\to}^*$ *(same closure)*
- $\underset{i}{\oplus\!\!\!\rightarrow} \cdot \underset{e}{\to} \subseteq \underset{e}{\to}^* \cdot \underset{i}{\oplus\!\!\!\rightarrow}^=$ (strong postponement)

77

Concretely: CbN and Head Factorization

MEMO from last semester

- $\underset{i}{\oplus\!\!\!\rightarrow}^* = \underset{i}{\to}^*$ *(same closure)*
- $\underset{i}{\oplus\!\!\!\rightarrow} \cdot \underset{e}{\to} \subseteq \underset{e}{\to}^* \cdot \underset{i}{\oplus\!\!\!\rightarrow}^=$ (strong postponement)

$\rhd_i$ *is the smallest reflexive relation on* Λ *such that:*

- $t \rhd_i t' \implies \lambda x\, t \rhd_i \lambda x\, t'$
- $t \rhd_i t',\ u \rhd u' \implies tu \rhd_i t'u'$
- $t \rhd t',\ u \rhd u' \implies (\lambda x\, t)u \rhd_i (\lambda x\, t')u'$.

$\to_i \subseteq \rhd_i \subseteq \to_i^*$

The development relation *is the least reflexive relation* $\rhd$ *on* Λ *s*

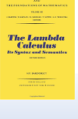
- $t \rhd t' \implies \lambda x\, t \rhd \lambda x\, t'$
- $t \rhd t',\ u \rhd u' \implies tu \rhd t'u'$
- $t \rhd t',\ u \rhd u' \implies (\lambda x\, t)u \rhd t'[x:=u']$.

1. Merge: $t \rhd_i \cdot \to_h u$ then $t \rhd u$
2. Split: If $t \rhd u$ then $t \to_h^* \cdot \rhd_i u$

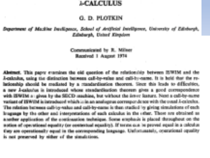
78

Examples of uses for factorization

79



Call-by-Name and Call-by-Value λ-calculus


CALL-BY-NAME, CALL-BY-VALUE AND THE λ-CALCULUS

G. D. PLOTKIN

Department of Machine Intelligence, School of Artificial Intelligence, University of Edinburgh, Edinburgh, United Kingdom

Communicated by R. Milner
Received 1 August 1974

Abstract. This paper examines the old question of the relationship between ISWIM and the λ-calculus, using the distinction between call-by-value and call-by-name. It is held that the relationship should be mediated by a standardisation theorem. Since this leads to difficulties, a new λ-calculus is introduced whose standardisation theorem gives a good correspondence with ISWIM as given by the SECD machine, but without the letrec feature. Next a call-by-name variant of ISWIM is introduced which is in an analogous correspondence with the usual λ-calculus. The relation between call-by-value and call-by-name is then studied by giving simulations of each language by the other and interpretations of each calculus in the other. These are obtained as another application of the continuation technique. Some emphasis is placed throughout on the notion of operational equality (or contextual equality). If terms can be proved equal in a calculus they are operationally equal in the corresponding language. Unfortunately, operational equality is not preserved by either of the simulations.

80

Call-by-Name and Call-by-Value λ–calculus: terms

Terms and *values* are generated by the following grammars

$$
\begin{array}{llll}
V & ::= & x \mid \lambda x.M & (\textit{Values}, \mathcal{V}) \\
M & ::= & x \mid c \mid \lambda x.M \mid MM & (\textit{Terms}\,)
\end{array}
$$

where x ranges over a countable set of *variables*, and c over a disjoint (possibly empty) set $\mathcal{O}$ of constants.

- If the set of constants is empty, the calulus is *pure*, and the set of terms is denoted Λ.
- Otherwise, the calculus is called *applied*, and the set of terms is often indicated as $\Lambda_{\mathcal{O}}$.

Terms are identified up to renaming of bound variables, where λx is the only binder constructor. $P\{Q/x\}$ is the capture-avoiding substitution of Q for the free occurrences of x in P.

81

Contexts (with one hole $(\!|\ |\!)$) are generated as follows. $\mathbf{C}(\!|M|\!)$ stands for the term obtained from $\mathbf{C}$ by replacing the hole with the term M (possibly capturing free variables of M).

$$\mathbf{C} ::= (\!|\ |\!) \mid\mid M\mathbf{C} \mid \mathbf{C}M \mid \lambda x.\mathbf{C} \qquad (\textit{Contexts})$$

82

Reduction = contextual closure of a rule

Contexts (with one hole $(\!|\ |\!)$) are generated as follows. $\mathbf{C}(\!|M|\!)$ stands for the term obtained from $\mathbf{C}$ by replacing the hole with the term M (possibly capturing free variables of M).

$$\mathbf{C} ::= (\!|\ |\!) \mid\mid M\mathbf{C} \mid \mathbf{C}M \mid \lambda x.\mathbf{C} \qquad (\textit{Contexts})$$

- A **rule** ρ is a binary relation on $\Lambda_{\mathcal{O}}$, which we also denote $\mapsto_\rho$, writing $R \mapsto_\rho R'$. R is called a *ρ-redex*.
  The best known rule is β:

  $$(\lambda x.M)N \mapsto_\beta M\{N/x\}$$

- A **reduction step** $\to_\rho$ is the closure under context $\mathbf{C}$ of ρ.
  Explicitly, $T \to T'$ holds if $T = \mathbf{C}(\!|R|\!)$, $T' = \mathbf{C}(\!|R'|\!)$, and $R \mapsto_\rho R'$.

83

Call-by-Name vs Call-by-Value λ–calculus

- *The λ-calculus can be seen both as an equational theory on terms and as an abstract model of computation.*

- *With the functional paradigm point of view, the meaning of any λ-term is the value it evaluates to.*


CALL-BY-NAME, CALL-BY-VALUE AND THE λ-CALCULUS

G. D. PLOTKIN

Department of Machine Intelligence, School of Artificial Intelligence, University of Edinburgh, Edinburgh, United Kingdom

Communicated by R. Milner
Received 1 August 1974

Abstract. This paper examines the old question of the relationship between ISWIM and the λ-calculus, using the distinction between call-by-value and call-by-name. It is held that the relationship should be mediated by a standardisation theorem. Since this leads to difficulties, a new λ-calculus is introduced whose standardisation theorem gives a good correspondence with ISWIM as given by the SECD machine, but without the letrec feature. Next a call-by-name variant of ISWIM is introduced which is in an analogous correspondence with the usual λ-calculus. The relation between call-by-value and call-by-name is then studied by giving simulations of each language by the other and interpretations of each calculus in the other. These are obtained as another application of the continuation technique. Some emphasis is placed throughout on the notion of operational equality (or contextual equality). If terms can be proved equal in a calculus they are operationally equal in the corresponding language. Unfortunately, operational equality is not preserved by either of the simulations.

84

Call-by-Name and Call-by-Value λ–calculus

CbN and CbV Calculi.

- The (pure) **Call-by-Name** calculus $\Lambda^{\mathsf{cbn}} = (\Lambda, \to_\beta)$ is the set of terms equipped with the contextual closure of the β-rule.

  $$(\lambda x.M)N \mapsto_\beta M\{N/x\}$$

- The (pure) **Call-by-Value** calculus $\Lambda^{\mathsf{cbv}} = (\Lambda, \to_{\beta_v})$ is the same set equipped with the contextual closure of the β_v-rule.

  $$(\lambda x.M)V \mapsto_{\beta_v} M\{V/x\} \text{ where } V \in \mathcal{V}$$

85

CbN: Head Reduction

Head reduction in CbN

Head reduction is the closure of β under head context

$$\lambda x_1 \dots x_n.(\!|\ |\!) M_1 \dots M_k$$

Head normal forms (*hnf*), whose set is denoted by $\mathcal{H}$, are its normal forms.

- Given a rule ρ, we write $\underset{\mathsf{h}}{\to}_\rho$ for its closure under head context.
- A step $\to_\rho$ is non-head, written $\underset{\neg\mathsf{h}}{\to}_\rho$ if it is not head.

What about?

$$\mathbf{H} ::= (\!|\ |\!) \mid \lambda x.\mathbf{H} \mid \mathbf{H}M$$

86

CbN Head Factorization

Head Factorization

Head factorization allows for a characterization of the terms which have head normal form, that is M has *hnf* if and only if $\underset{\mathsf{h}}{\to}$-reduction from M terminates.

▶ **Theorem 2** (Head Factorization)**.**

- Head Factorization: $\quad \to_\beta^* \subseteq \underset{\mathsf{h}}{\to}_\beta^* \cdot \underset{\neg\mathsf{h}}{\to}_\beta^*$.
- Head Normalization: *M has hnf if and only if $M \underset{\mathsf{h}}{\to}_\beta^* S$ (for some $S \in \mathcal{H}$)* .

87

Call-by-Value

- According to the function paradigm of computation the goal of every computation is to determine its value
- Since functions are seen as values, it is natural to consider *weak evaluation*. In practical implementations, weak evaluation is more realistic than the full beta reduction

88

CbV: Weak Reduction

Weak reductions in CbV

The result of interest are **values** (*i.e.* functions).

In languages, in general the reduction is *weak*, that is, it does not reduce in the body of a function.

There are three main weak schemes: left, right and in arbitrary order.

Left contexts **L**, *right* contexts **R**, and (arbitrary order) *weak* contexts **W** are defined by

$$\mathbf{L} ::= (\!|\ |\!) \mid \mathbf{L}M \mid V\mathbf{L}$$
$$\mathbf{R} ::= (\!|\ |\!) \mid M\mathbf{R} \mid \mathbf{R}V$$
$$\mathbf{W} ::= (\!|\ |\!) \mid \mathbf{W}M \mid M\mathbf{W}$$

Given a rule $\mapsto$ on Λ, *weak reduction* $\underset{\mathsf{w}}{\rightarrow}$ is the closure of $\mapsto$ under context **W**.

A step $T \rightarrow S$ is non-weak, written $T \underset{\neg\mathsf{w}}{\rightarrow} S$ if it is not weak. Similarly for left ($\underset{\mathsf{l}}{\rightarrow}$ and $\underset{\neg\mathsf{l}}{\rightarrow}$), and right ($\underset{\mathsf{r}}{\rightarrow}$ and $\underset{\neg\mathsf{r}}{\rightarrow}$).

▶ **Fact 3** (Weak normal forms). *Given M a closed term, M is $\underset{\mathsf{s}}{\rightarrow}$-normal* iff *$M$ is a value.*

89

CbV: *Left* contexts **L**, *right* contexts **R**, and (arbitrary order) *weak* contexts **W** are defined by

$$\mathbf{L} ::= (\!|\ |\!) \mid \mathbf{L}M \mid V\mathbf{L}$$
$$\mathbf{R} ::= (\!|\ |\!) \mid M\mathbf{R} \mid \mathbf{R}V$$
$$\mathbf{W} ::= (\!|\ |\!) \mid \mathbf{W}M \mid M\mathbf{W}$$

The closure under **L** (resp. **W**,**R**) context is noted $\underset{\mathsf{l}}{\rightarrow}$ (resp $\underset{\mathsf{w}}{\rightarrow}$, $\underset{\mathsf{r}}{\rightarrow}$)

▶ **Fact 3** (Weak normal forms). *Given M a closed term, M is $\underset{\mathsf{s}}{\rightarrow}$-normal* iff *$M$ is a value.*

Question: which of the above reductions are deterministic?

▶ **Fact 6** (?). *Let M be a closed term.*

1. *$M \underset{\mathsf{l}}{\rightarrow}^* V$ iff $M \underset{\mathsf{r}}{\rightarrow}^* V$. True?*
2. *$M \underset{\mathsf{w}}{\rightarrow}^V$ iff $M \underset{\mathsf{l}}{\rightarrow}^*$. True?*
3. *Assume you proved $M \underset{\mathsf{l}}{\rightarrow}^k V$ (runtime is k). Does the sequence of $\underset{\mathsf{r}}{\rightarrow}$-steps also terminates? Can we say how long does it take?*
4. *With the same assumption as above, what about $\underset{\mathsf{w}}{\rightarrow}$?*

90

CbV Weak Factorization

Weak Factorization.

Let $\mathsf{s} \in \{\mathsf{w},\mathsf{l},\mathsf{r}\}$

- *weak factorization of* $\rightarrow_{\beta_v}$: $\quad \rightarrow_{\beta_v}{}^* \subseteq \underset{\mathsf{s}}{\rightarrow}_{\beta_v}{}^* \cdot \underset{\neg\mathsf{s}}{\rightarrow}_{\beta_v}{}^*$.
- *Convergence:* $T \rightarrow_{\beta_v} W (W \in \mathcal{V})$ if and only if $T \underset{\mathsf{s}}{\rightarrow}_{\beta_v}{}^* V \ (V \in \mathcal{V})$

▶ **Corollary 4.** *Given M a closed term, M has a β_v-reduction to a value, if and only if the $\underset{\mathsf{s}}{\rightarrow}_{\beta_v}$-reduction from M terminates.*

91

Basic properties of the contextual closure

92

Basic properties of contextual closure

If a step $T \rightarrow_\gamma T'$ is obtained by closure under *non-empty context* of a rule $\mapsto_\gamma$, then T and T' have *the same shape*, i.e. *both terms are an application (resp. an abstraction, a variable).*

▶ **Fact 5** (Shape preservation).

- *Assume $T = \mathbf{C}(\!|R|\!) \rightarrow \mathbf{C}(\!|R'|\!) = T'$ and that the context* **C** *is* non-empty. *Then T and T' have the same shape.*
- *Hence, for any internal step $M \underset{\neg\mathsf{s}}{\rightarrow} M'$ ($\mathsf{s} \in \{\mathsf{h}, \mathsf{w},\mathsf{l},\mathsf{r},\dots\}$) M and M have the same shape.*

The following is an easy to verify consequence.

▶ **Lemma 6** (Redexes preservation).

1. *CbN: Assume $T \underset{\neg\mathsf{h}}{\rightarrow}_\beta S$. T is a β-redex* iff *so is S.*
2. *CbV. Assume $T \underset{\neg\mathsf{w}}{\rightarrow}_{\beta_v} S$. T is a β_v-redex* iff *so is S.*

93

Internal steps preserve head and weak normal nf

Fixed a set of redexes $\mathcal{R}$, M is w-normal (resp. h-normal) if there is no redex $R \in \mathcal{R}$ such that $M = \mathbf{W}(\!|R|\!)$ (resp. $M = \mathbf{H}(\!|R|\!)$)

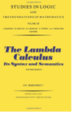
▸ **Lemma 7** (Surface normal forms). 1. *CbN. Let $\mathcal{R}$ be the set of β-redexes.*
Assume $M \underset{\neg h}{\to}_\beta M'$. *$M$ is h-normal ⇔ M' is h-normal.*
2. *CbV. Let $\mathcal{R}$ be the set of β_v-redexes.*
Assume $M \underset{\neg w}{\to}_{\beta_v} M'$. *$M$ is w-normal ⇔ M' is w-normal.*

Homework: point 2

Back to Factorization

Back to using it

Recap



- Classical key result (e.g. in Barendregt 84 book)

in Call-by-Name:

- Head Factorization: $\to_\beta^* \subseteq \underset{h}{\to}_\beta^* \cdot \underset{\neg h}{\to}_\beta^*$.
- Head Normalization: *M has* hnf *if and only if* $M \underset{h}{\to}_\beta^* S$ *(for some $S \in \mathcal{H}$)* .

- Classical key result [Plotkin 75]


CALL-BY-NAME, CALL-BY-VALUE AND THE λ-CALCULUS

G. D. PLOTKIN

in Call-by-Value:

Let s ∈ {w,l,r}

- *weak factorization of* $\to_{\beta_v}$*:* $\to_{\beta_v}{}^* \subseteq \underset{s}{\to}_{\beta_v}{}^* \cdot \underset{\neg s}{\to}_{\beta_v}{}^*$.
- *Convergence:* $T \to_{\beta_v} W (W \in \mathcal{V})$ if and only if $T \underset{s}{\to}_{\beta_v}{}^* V \ (V \in \mathcal{V})$

CbN Head Factorization

Head Factorization

Head factorization allows for a characterization of the terms which have head normal form, that is M has *hnf* if and only if $\underset{h}{\to}$-reduction from M terminates.

▸ **Theorem 2** (Head Factorization).
- Head Factorization: $\to_\beta^* \subseteq \underset{h}{\to}_\beta^* \cdot \underset{\neg h}{\to}_\beta^*$.
- Head Normalization: *M has* hnf *if and only if* $M \underset{h}{\to}_\beta^* S$ *(for some $S \in \mathcal{H}$)* .

CbV Weak Factorization

Weak Factorization.

Let s ∈ {w,l,r}

- *weak factorization of* $\to_{\beta_v}$*:* $\to_{\beta_v}{}^* \subseteq \underset{s}{\to}_{\beta_v}{}^* \cdot \underset{\neg s}{\to}_{\beta_v}{}^*$.
- *Convergence:* $T \to_{\beta_v} W (W \in \mathcal{V})$ if and only if $T \underset{s}{\to}_{\beta_v}{}^* V \ (V \in \mathcal{V})$

▸ **Corollary 4.** *Given M a closed term, M has a β_v-reduction to a value, if and only if the* $\underset{s}{\to}_{\beta_v}$*-reduction from M terminates.*

You designed a system You have Factorization Now what?

From Factorization to **Normalization** (or Standardization)
in a few easy steps [Mitschke 79]