Exponential as Substitutions and the Cost of Cut Elimination in Linear Logic

Beniamino Accattoli

Inria & École Polytechnique

January 28, 2022

LL, Resources, & Programs

LL is a resource-conscious logic.

Curry-Howard:
β-reduction decomposes through LL cut elimination.

Even the untyped λ-calculus can be represented.

Using a recursive formula $\mathtt{m} = !\mathtt{m} \multimap \mathtt{m}$.

LL, Resources, & Programs

LL is a resource-conscious logic.

Curry-Howard:
β-reduction decomposes through LL cut elimination.

Even the untyped λ-calculus can be represented.
Using a recursive formula $\mathtt{m} = !\mathtt{m} \multimap \mathtt{m}$.

LL, Resources, & Programs

LL is a resource-conscious logic.

Curry-Howard:
β-reduction decomposes through LL cut elimination.

Even the untyped λ-calculus can be represented.

Using a recursive formula $\mathtt{m} = !\mathtt{m} \multimap \mathtt{m}$.

LL, Resources, & Programs

LL is a resource-conscious logic.

Curry-Howard:
β-reduction decomposes through LL cut elimination.

Even the untyped λ-calculus can be represented.

Using a recursive formula $\mathtt{m} = !\mathtt{m} \multimap \mathtt{m}$.

LL, Resources, & Programs

LL is a resource-conscious logic.

Curry-Howard:
β-reduction decomposes through LL cut elimination.

Even the untyped λ-calculus can be represented.

Using a recursive formula $\mathtt{m} = !\mathtt{m} \multimap \mathtt{m}$.

Complexity of Evaluation as Cut-Elimination?

Given a λ-term t represented as a LL proof π_t.

π_t normalizes in n steps of cut-elimination.

What can we say about the complexity of t? In general, nothing.

Complexity of Evaluation as Cut-Elimination?

Given a λ-term t represented as a LL proof π_t.

π_t normalizes in n steps of cut-elimination.

What can we say about the complexity of t? In general, nothing.

Complexity of Evaluation as Cut-Elimination?

Given a λ-term t represented as a LL proof π_t.

π_t normalizes in n steps of cut-elimination.

What can we say about the complexity of t? In general, nothing.

Complexity of Evaluation as Cut-Elimination?

Given a λ-term t represented as a LL proof π_t.

π_t normalizes in n steps of cut-elimination.

What can we say about the complexity of t? In general, nothing.

Hole in the Literature

Let π be a proof of MELL.

At present, no relationships between:

The # of steps of cut elimination of π,

The complexity of eliminating cuts from π.

Hole in the Literature

Let π be a proof of MELL.

At present, no relationships between:

The # of steps of cut elimination of π,

The complexity of eliminating cuts from π.

Hole in the Literature

Let π be a proof of MELL.

At present, no relationships between:

The # of steps of cut elimination of π,

The complexity of eliminating cuts from π.

Hole in the Literature

Let π be a proof of MELL.

At present, no relationships between:

- The # of steps of cut elimination of π,
- The complexity of eliminating cuts from π.

Time Cost Models

Open problem:

Finding a time cost model for LL

Time Cost Models

Open problem:

Finding a time cost model for LL

Polynomial Cost Models

Ideal (first) solution:

A cut elimination strategy $\rightarrow_x$

of which the # of steps is a polynomial cost model.

That is:
The cost of implementing $\pi \rightarrow_x^n \pi'$ is polynomial in $|\pi|$ and n.

CAREFUL: n can be whatever, it need not being polynomial in $|\pi|$.

Polynomial Cost Models

Ideal (first) solution:

A cut elimination strategy $\rightarrow_x$

of which the # of steps is a polynomial cost model.

That is:
The cost of implementing $\pi \rightarrow_x^n \pi'$ is polynomial in $|\pi|$ and n.

CAREFUL: n can be whatever, it need not being polynomial in $|\pi|$.

Polynomial Cost Models

Ideal (first) solution:

A cut elimination strategy $\rightarrow_x$

of which the # of steps is a polynomial cost model.

That is:
The cost of implementing $\pi \rightarrow_x^n \pi'$ is polynomial in $|\pi|$ and n.

CAREFUL: n can be whatever, it need not being polynomial in $|\pi|$.

Polynomial Cost Models

Ideal (first) solution:

A cut elimination strategy $\rightarrow_x$

of which the # of steps is a polynomial cost model.

That is:

The cost of implementing $\pi \rightarrow_x^n \pi'$ is polynomial in $|\pi|$ and n.

CAREFUL: n can be whatever, it need not being polynomial in $|\pi|$.

Polynomial Cost Models

Ideal (first) solution:

A cut elimination strategy $\rightarrow_x$

of which the # of steps is a polynomial cost model.

That is:
The cost of implementing $\pi \rightarrow_x^n \pi'$ is polynomial in $|\pi|$ and n.

CAREFUL: n can be whatever, it need not being polynomial in $|\pi|$.

Polynomial Cost Models

Ideal (first) solution:

A cut elimination strategy $\rightarrow_x$

of which the # of steps is a polynomial cost model.

That is:
The cost of implementing $\pi \rightarrow_x^n \pi'$ is polynomial in $|\pi|$ and n.

CAREFUL: n can be whatever, it need not being polynomial in $|\pi|$.

Polynomial Cost Models

Ideal (first) solution:

A cut elimination strategy $\rightarrow_x$

of which the # of steps is a polynomial cost model.

That is:
The cost of implementing $\pi \rightarrow_x^n \pi'$ is polynomial in $|\pi|$ and n.

CAREFUL: n can be whatever, it need not being polynomial in $|\pi|$.

Polynomial Cost Models

Ideal (first) solution:

A cut elimination strategy $\rightarrow_x$

of which the # of steps is a polynomial cost model.

That is:
The cost of implementing $\pi \rightarrow_x^n \pi'$ is polynomial in $|\pi|$ and n.

CAREFUL: n can be whatever, it need not being polynomial in $|\pi|$.

Time Cost Models

Last 10 years:

Good understanding of time cost models for the λ-calculus.

Good news: now of space ones too.

Enabled by a perfect dissection of meta-level substitution.

Namely, Accattoli and Kesner's Linear Substitution Calculus (LSC).

The dissection:

Ideas from LL, but not literally, and does not apply (yet) to LL.

Time Cost Models

Last 10 years:
Good understanding of time cost models for the λ-calculus.

Good news: now of space ones too.

Enabled by a perfect dissection of meta-level substitution.

Namely, Accattoli and Kesner's Linear Substitution Calculus (LSC).

The dissection:
Ideas from LL, but not literally, and does not apply (yet) to LL.

Time Cost Models

Last 10 years:
Good understanding of time cost models for the λ-calculus.

Good news: now of space ones too.

Enabled by a perfect dissection of meta-level substitution.

Namely, Accattoli and Kesner's Linear Substitution Calculus (LSC).

The dissection:
Ideas from LL, but not literally, and does not apply (yet) to LL.

Time Cost Models

Last 10 years:
Good understanding of time cost models for the λ-calculus.

Good news: now of space ones too.

Enabled by a perfect dissection of meta-level substitution.

Namely, Accattoli and Kesner's Linear Substitution Calculus (LSC).

The dissection:
Ideas from LL, but not literally, and does not apply (yet) to LL.

Time Cost Models

Last 10 years:
Good understanding of time cost models for the λ-calculus.

Good news: now of space ones too.

Enabled by a perfect dissection of meta-level substitution.

Namely, Accattoli and Kesner's Linear Substitution Calculus (LSC).

The dissection:
Ideas from LL, but not literally, and does not apply (yet) to LL.

Time Cost Models

Last 10 years:
Good understanding of time cost models for the λ-calculus.

Good news: now of space ones too.

Enabled by a perfect dissection of meta-level substitution.

Namely, Accattoli and Kesner's Linear Substitution Calculus (LSC).

The dissection:
Ideas from LL, but not literally, and does not apply (yet) to LL.

Time Cost Models

Last 10 years:
Good understanding of time cost models for the λ-calculus.

Good news: now of space ones too.

Enabled by a perfect dissection of meta-level substitution.

Namely, Accattoli and Kesner's Linear Substitution Calculus (LSC).

The dissection:
Ideas from LL, but not literally, and does not apply (yet) to LL.

Outline

Explicit Substitutions Basics

Decompose β:

$$(\lambda x.t)s \;\to_{\beta}\; t\{x \leftarrow s\}$$

via a new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\to_{\mathrm{B}}\; t[x \leftarrow s] \;\to_{\mathrm{S}}\; t\{x \leftarrow s\}$$

Explicit Substitutions Basics

Decompose β:

$$(\lambda x.t)s \;\; \rightarrow_{\beta} \;\; t\{x \leftarrow s\}$$

via a new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\; \rightarrow_{\mathrm{B}} \;\; t[x \leftarrow s] \;\; \rightarrow_{\mathrm{S}} \;\; t\{x \leftarrow s\}$$

Explicit Substitutions Basics

Decompose β:

$$(\lambda x.t)s \;\; \rightarrow_{\beta} \;\; t\{x \leftarrow s\}$$

via a new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\; \rightarrow_{\mathrm{B}} \;\; t[x \leftarrow s] \;\; \rightarrow_{\mathrm{S}} \;\; t\{x \leftarrow s\}$$

Explicit Substitutions Basics

Decompose β:

$$(\lambda x.t)s \;\; \rightarrow_{\beta} \;\; t\{x \leftarrow s\}$$

via a new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\; \rightarrow_{\mathrm{B}} \;\; t[x \leftarrow s] \;\; \rightarrow_{\mathrm{S}} \;\; t\{x \leftarrow s\}$$

Explicit Substitutions Basics

A new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\; \rightarrow_{\mathtt{B}} \;\; t[x \leftarrow s] \;\; \rightarrow_{\mathtt{S}} \;\; t\{x \leftarrow s\}$$

Next, decompose $\rightarrow_{\mathtt{S}}$ in a set of micro-step rules.

There are many (many!) possible sets of such rules.

Explicit Substitutions Basics

A new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\; \rightarrow_{\mathtt{B}} \;\; t[x \leftarrow s] \;\; \rightarrow_{\mathtt{S}} \;\; t\{x \leftarrow s\}$$

Next, decompose $\rightarrow_{\mathtt{S}}$ in a set of micro-step rules.

There are many (many!) possible sets of such rules.

Explicit Substitutions Basics

A new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\; \rightarrow_{\mathtt{B}} \;\; t[x \leftarrow s] \;\; \rightarrow_{\mathtt{S}} \;\; t\{x \leftarrow s\}$$

Next, decompose $\rightarrow_{\mathtt{S}}$ in a set of micro-step rules.

There are many (many!) possible sets of such rules.

Explicit Substitutions Basics

A new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\; \rightarrow_{\mathtt{B}} \;\; t[x \leftarrow s] \;\; \rightarrow_{\mathtt{S}} \;\; t\{x \leftarrow s\}$$

Key point for later:
Rules for $\rightarrow_{\mathtt{S}}$ are always strongly normalizing.

Somewhat surprising:
Counting only $\rightarrow_{\mathtt{B}}$ steps gives a polynomial cost model.

Explicit Substitutions Basics

A new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\to_{\mathtt{B}}\; t[x \leftarrow s] \;\to_{\mathtt{S}}\; t\{x \leftarrow s\}$$

Key point for later:
Rules for $\to_{\mathtt{S}}$ are always strongly normalizing.

Somewhat surprising:
Counting only $\to_{\mathtt{B}}$ steps gives a polynomial cost model.

Explicit Substitutions Basics

A new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\; \rightarrow_{\mathtt{B}} \;\; t[x \leftarrow s] \;\; \rightarrow_{\mathtt{S}} \;\; t\{x \leftarrow s\}$$

Key point for later:
Rules for $\rightarrow_{\mathtt{S}}$ are always strongly normalizing.

Somewhat surprising:
Counting only $\rightarrow_{\mathtt{B}}$ steps gives a polynomial cost model.

Explicit Substitutions Basics

A new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\; \rightarrow_{\mathtt{B}} \;\; t[x \leftarrow s] \;\; \rightarrow_{\mathtt{S}} \;\; t\{x \leftarrow s\}$$

Key point for later:
Rules for $\rightarrow_{\mathtt{S}}$ are always strongly normalizing.

Somewhat surprising:
Counting only $\rightarrow_{\mathtt{B}}$ steps gives a polynomial cost model.

Explicit Substitutions Basics

A new constructor $t[x \leftarrow s]$ and two steps:

$$(\lambda x.t)s \;\; \rightarrow_{\mathtt{B}} \;\; t[x \leftarrow s] \;\; \rightarrow_{\mathtt{S}} \;\; t\{x \leftarrow s\}$$

Key point for later:
Rules for $\rightarrow_{\mathtt{S}}$ are always strongly normalizing.

Somewhat surprising:
Counting only $\rightarrow_{\mathtt{B}}$ steps gives a polynomial cost model.

Explicit Substitutions Basics

Historically, the first λ-calculus with ES had rewriting problems.

Melliès'95

Similar idea in LL: exponentials decompose substitutions.

Di Cosmo and Kenser borrowed from LL to improve ES ('97).

Explicit Substitutions Basics

Historically, the first λ-calculus with ES had rewriting problems.

Melliès'95

Similar idea in LL: exponentials decompose substitutions.

Di Cosmo and Kenser borrowed from LL to improve ES ('97).

Explicit Substitutions Basics

Historically, the first λ-calculus with ES had rewriting problems.

Melliès'95

Similar idea in LL: exponentials decompose substitutions.

Di Cosmo and Kenser borrowed from LL to improve ES ('97).

Explicit Substitutions Basics

Historically, the first λ-calculus with ES had rewriting problems.
Melliès'95

Similar idea in LL: exponentials decompose substitutions.

Di Cosmo and Kenser borrowed from LL to improve ES ('97).

Explicit Substitutions Basics

ES went a long (not always glorious) way.

In '12, the arguably canonical λ-calculus with ES was found.

Namely, the linear substitution calculus (LSC).

Accattoli and Kesner, borrowing from proof nets and Milner.

Explicit Substitutions Basics

ES went a long (not always glorious) way.

In '12, the arguably canonical λ-calculus with ES was found.

Namely, the linear substitution calculus (LSC).

Accattoli and Kesner, borrowing from proof nets and Milner.

Explicit Substitutions Basics

ES went a long (not always glorious) way.

In '12, the arguably canonical λ-calculus with ES was found.

Namely, the linear substitution calculus (LSC).

Accattoli and Kesner, borrowing from proof nets and Milner.

Explicit Substitutions Basics

ES went a long (not always glorious) way.

In '12, the arguably canonical λ-calculus with ES was found.

Namely, the linear substitution calculus (LSC).

Accattoli and Kesner, borrowing from proof nets and Milner.

Linear Substitution Calculus (LSC)

Terms and contexts:

$$t, s, u \quad := \quad x \quad | \quad \lambda x.t \quad | \quad ts \quad | \quad t[x \leftarrow s]$$

$$C \quad := \quad \langle \cdot \rangle \quad | \quad Ct \quad | \quad tC \quad | \quad \lambda x.C \quad | \quad C[x \leftarrow t] \quad | \quad t[x \leftarrow C]$$

Lists of substitutions:

$$L \quad := \quad \langle \cdot \rangle \quad | \quad L[x \leftarrow t]$$

Rewriting rules (closed by contexts $C\langle \cdot \rangle$):

distant **B**eta	$L\langle \lambda x.t \rangle s$	$\to_{\mathtt{dB}}$	$L\langle t[x \leftarrow s] \rangle$	
linear **s**ubstitution	$C\langle x \rangle [x \leftarrow s]$	$\to_{\mathtt{ms}}$	$C\langle s \rangle [x \leftarrow s]$	
garbage **c**ollection	$t[x \leftarrow s]$	$\to_{\mathtt{gc}}$	t	if $x \notin \mathtt{fv}(t)$

Outline

The Sub-Term Property

Strategies of the LSC naturally have the sub-term property:

> all sub-terms duplicated or erased
>
> along an evaluation sequence from t
>
> are sub-terms of t.

This is the key property for the study of cost models.

The Sub-Term Property

Strategies of the LSC naturally have the sub-term property:

> all sub-terms duplicated or erased
>
> along an evaluation sequence from t
>
> are sub-terms of t.

This is the key property for the study of cost models.

The Sub-Term Property

Strategies of the LSC naturally have the sub-term property:

all sub-terms duplicated or erased

along an evaluation sequence from t

are sub-terms of t.

This is the key property for the study of cost models.

The Sub-Term Property

Strategies of the LSC naturally have the sub-term property:

> all sub-terms duplicated or erased
>
> along an evaluation sequence from t
>
> are sub-terms of t.

This is the key property for the study of cost models.

The Sub-Term Property

Strategies of the LSC naturally have the sub-term property:

> all sub-terms duplicated or erased
>
> along an evaluation sequence from t
>
> are sub-terms of t.

This is the key property for the study of cost models.

Sub-Term Property and Polynomial Cost Models

Suppose that $\to_x$ has the sub-term property and is micro-step.

Then $t \to_x^n s$ is implementable in time polynomial in $|t|$ and n.

Usually in bi-linear time.

Therefore:

sub-term property + micro-steps $\Rightarrow$ polynomial cost model.

Sub-Term Property and Polynomial Cost Models

Suppose that $\to_x$ has the sub-term property and is micro-step.

Then $t \to_x^n s$ is implementable in time polynomial in $|t|$ and n.

Usually in bi-linear time.

Therefore:

sub-term property + micro-steps $\Rightarrow$ polynomial cost model.

Sub-Term Property and Polynomial Cost Models

Suppose that $\to_x$ has the sub-term property and is micro-step.

Then $t \to_x^n s$ is implementable in time polynomial in $|t|$ and n.

Usually in bi-linear time.

Therefore:

sub-term property + micro-steps $\Rightarrow$ polynomial cost model.

Sub-Term Property and Polynomial Cost Models

Suppose that $\to_x$ has the sub-term property and is micro-step.

Then $t \to_x^n s$ is implementable in time polynomial in $|t|$ and n.

Usually in bi-linear time.

Therefore:

sub-term property + micro-steps $\Rightarrow$ polynomial cost model.

Sub-Term Property and Polynomial Cost Models

Suppose that $\rightarrow_x$ has the sub-term property and is micro-step.

Then $t \rightarrow_x^n s$ is implementable in time polynomial in $|t|$ and n.

Usually in bi-linear time.

Therefore:

sub-term property + micro-steps $\Rightarrow$ polynomial cost model.

λ-Calculus *vs* Sub-Term Property

In the λ-calculus, no strategy has the sub-term property.

Consider $\tau_t := \lambda y.ytt$ and $\mathtt{I} := \lambda x.x$. Then:

$$\begin{array}{rll} s := & (\lambda x.x(\lambda z.\tau_z)\tau_x)\mathtt{I} & \to_\beta \\ & \mathtt{I}(\lambda z.\tau_z)\tau_{\mathtt{I}} & \to_\beta \\ & (\lambda z.\tau_z)\tau_{\mathtt{I}} & \to_\beta \quad \tau_{\tau_{\mathtt{I}}} \end{array}$$

There never is a choice, each term has only one redex.

The third step duplicates $\tau_{\mathtt{I}}$ which is not a sub-term of s.

λ-Calculus *vs* Sub-Term Property

In the λ-calculus, no strategy has the sub-term property.

Consider $\tau_t := \lambda y.ytt$ and $\mathtt{I} := \lambda x.x$. Then:

$$\begin{array}{rll} s := & (\lambda x.x(\lambda z.\tau_z)\tau_x)\mathtt{I} & \to_\beta \\ & \mathtt{I}(\lambda z.\tau_z)\tau_{\mathtt{I}} & \to_\beta \\ & (\lambda z.\tau_z)\tau_{\mathtt{I}} & \to_\beta \quad \tau_{\tau_{\mathtt{I}}} \end{array}$$

There never is a choice, each term has only one redex.

The third step duplicates $\tau_{\mathtt{I}}$ which is not a sub-term of s.

λ-Calculus *vs* Sub-Term Property

In the λ-calculus, no strategy has the sub-term property.

Consider $\tau_t := \lambda y.ytt$ and $\mathtt{I} := \lambda x.x$. Then:

$$\begin{array}{rlll} s := & (\lambda x.x(\lambda z.\tau_z)\tau_x)\mathtt{I} & \to_\beta & \\ & \mathtt{I}(\lambda z.\tau_z)\tau_{\mathtt{I}} & \to_\beta & \\ & (\lambda z.\tau_z)\tau_{\mathtt{I}} & \to_\beta & \tau_{\tau_{\mathtt{I}}} \end{array}$$

There never is a choice, each term has only one redex.

The third step duplicates $\tau_{\mathtt{I}}$ which is not a sub-term of s.

λ-Calculus *vs* Sub-Term Property

In the λ-calculus, no strategy has the sub-term property.

Consider $\tau_t := \lambda y.ytt$ and $\mathtt{I} := \lambda x.x$. Then:

$$\begin{array}{rlll} s := & (\lambda x.x(\lambda z.\tau_z)\tau_x)\mathtt{I} & \to_\beta & \\ & \mathtt{I}(\lambda z.\tau_z)\tau_{\mathtt{I}} & \to_\beta & \\ & (\lambda z.\tau_z)\tau_{\mathtt{I}} & \to_\beta & \tau_{\tau_{\mathtt{I}}} \end{array}$$

There never is a choice, each term has only one redex.

The third step duplicates $\tau_{\mathtt{I}}$ which is not a sub-term of s.

λ-Calculus *vs* Sub-Term Property

In the λ-calculus, no strategy has the sub-term property.

Consider $\tau_t := \lambda y.ytt$ and $\mathtt{I} := \lambda x.x$. Then:

$$\begin{array}{rlll} s := & (\lambda x.x(\lambda z.\tau_z)\tau_x)\mathtt{I} & \to_\beta & \\ & \mathtt{I}(\lambda z.\tau_z)\tau_{\mathtt{I}} & \to_\beta & \\ & (\lambda z.\tau_z)\tau_{\mathtt{I}} & \to_\beta & \tau_{\tau_{\mathtt{I}}} \end{array}$$

There never is a choice, each term has only one redex.

The third step duplicates $\tau_{\mathtt{I}}$ which is not a sub-term of s.

No Sub-Term Property and Size Explosion

Lack of sub-term property leads to size explosion.

$$s_n \to_\beta^n u_n \qquad \text{with } |s_n| = \mathcal{O}(n) \text{ and } |u_n| = \Omega(2^n)$$

The number of steps does not seem to be a polynomial cost model.

No Sub-Term Property and Size Explosion

Lack of sub-term property leads to size explosion.

$$s_n \to_\beta^n u_n \qquad \text{with } |s_n| = \mathcal{O}(n) \text{ and } |u_n| = \Omega(2^n)$$

The number of steps does not seem to be a polynomial cost model.

No Sub-Term Property and Size Explosion

Lack of sub-term property leads to size explosion.

$$s_n \ \rightarrow_\beta^n \ u_n \qquad \text{with } |s_n| = \mathcal{O}(n) \text{ and } |u_n| = \Omega(2^n)$$

The number of steps does not seem to be a polynomial cost model.

No Sub-Term Property and Size Explosion

Moral:

sub-term property $\Rightarrow$ polynomial cost model.

no sub-term property $\Rightarrow$ size explosion.

No Sub-Term Property and Size Explosion

Moral:

sub-term property $\Rightarrow$ polynomial cost model.

no sub-term property $\Rightarrow$ size explosion.

No Sub-Term Property and Size Explosion

Moral:

sub-term property $\Rightarrow$ polynomial cost model.

no sub-term property $\Rightarrow$ size explosion.

Sub-Term Property

The sub-term property is mandatory for polynomial cost models.

$$\frac{\text{Sub-term}}{\text{Polyn. Cost Models}} = \frac{\text{Cut elimination}}{\text{Sequent Calculus}} = \frac{\text{Sub-formula}}{\text{Proof search}}$$

Mandatory also for logarithmic space.

Sub-Term Property

The sub-term property is mandatory for polynomial cost models.

$$\frac{\text{Sub-term}}{\text{Polyn. Cost Models}} = \frac{\text{Cut elimination}}{\text{Sequent Calculus}} = \frac{\text{Sub-formula}}{\text{Proof search}}$$

Mandatory also for logarithmic space.

Sub-Term Property

The sub-term property is mandatory for polynomial cost models.

$$\frac{\text{Sub-term}}{\text{Polyn. Cost Models}} = \frac{\text{Cut elimination}}{\text{Sequent Calculus}} = \frac{\text{Sub-formula}}{\text{Proof search}}$$

Mandatory also for logarithmic space.

Outline

Linear Logic

Linear logic models micro-step well.

Unfortunately:
At present, no strategies with the sub-term property.

LL strategies of reference in the literature:

λ	LL
Head	Level 0
Leftmost	Least level

Linear Logic

Linear logic models micro-step well.

Unfortunately:
At present, no strategies with the sub-term property.

LL strategies of reference in the literature:

λ	LL
Head	Level 0
Leftmost	Least level

Linear Logic

Linear logic models micro-step well.

Unfortunately:
At present, no strategies with the sub-term property.

LL strategies of reference in the literature:

λ	LL
Head	Level 0
Leftmost	Least level

Linear Logic

Linear logic models micro-step well.

Unfortunately:
At present, no strategies with the sub-term property.

LL strategies of reference in the literature:

λ	LL
Head	Level 0
Leftmost	Least level

Linear Logic

Linear logic models micro-step well.

Unfortunately:
At present, no strategies with the sub-term property.

LL strategies of reference in the literature:

λ	LL
Head	Level 0
Leftmost	Least level

Linear Logic Strategies

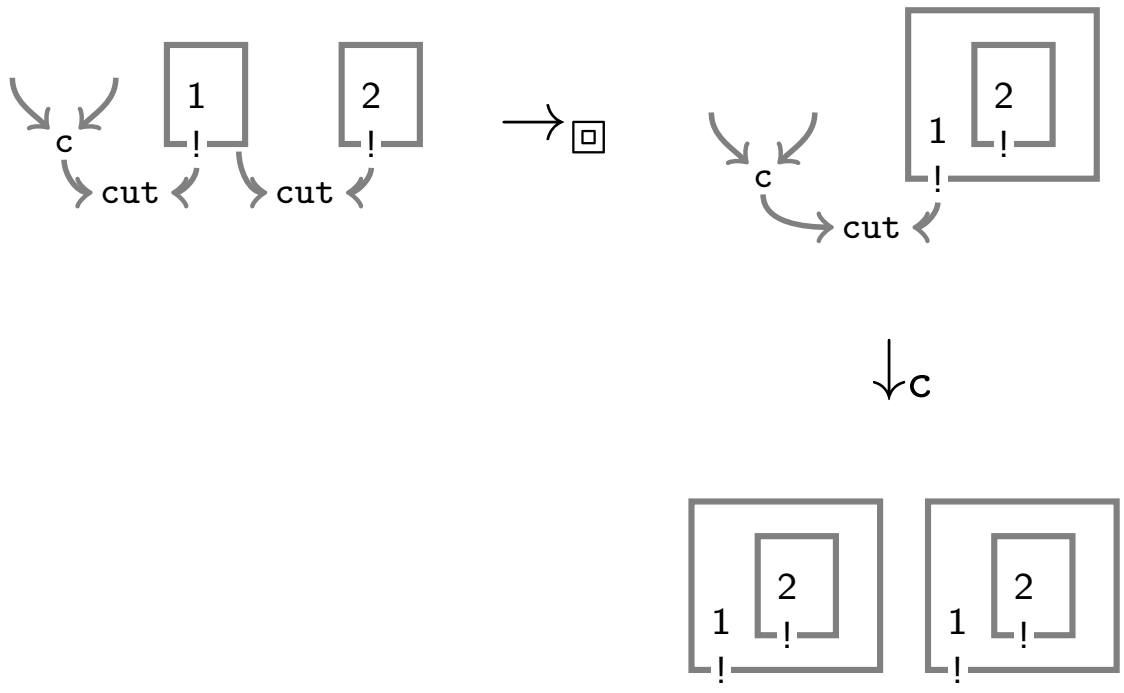
Reduction at level 0:

No sub-term property:

The second step duplicates a box not in the initial proof net.

Linear Logic Strategies

Reduction at level 0:

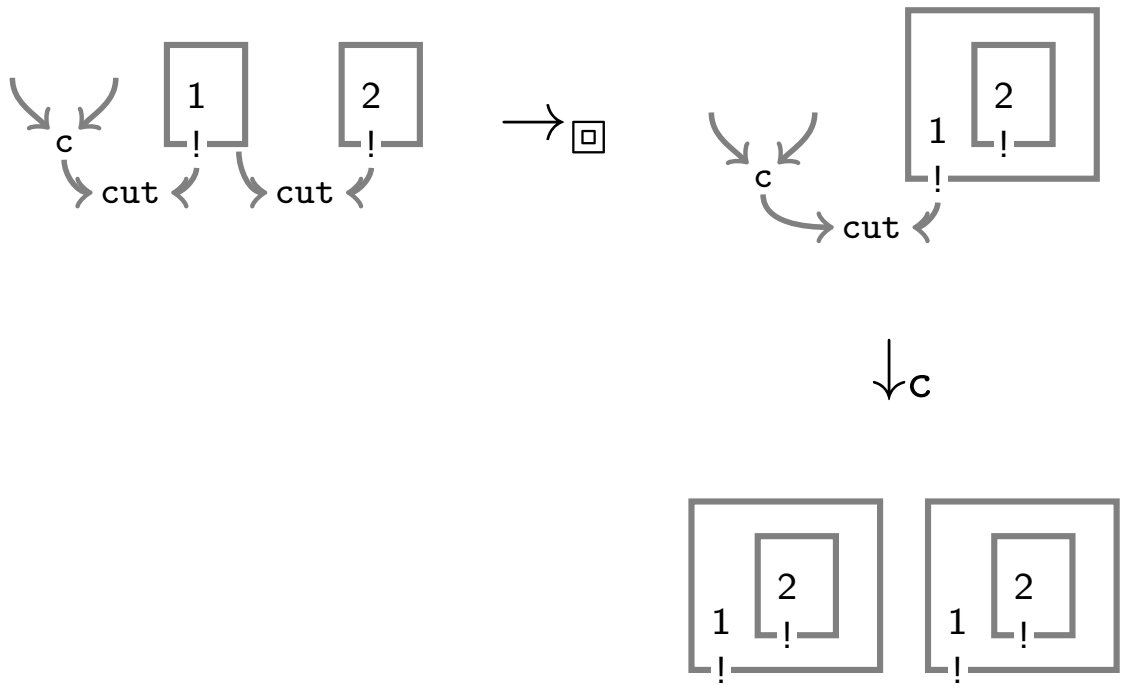


No sub-term property:

The second step duplicates a box not in the initial proof net.

Linear Logic Strategies

Reduction at level 0:

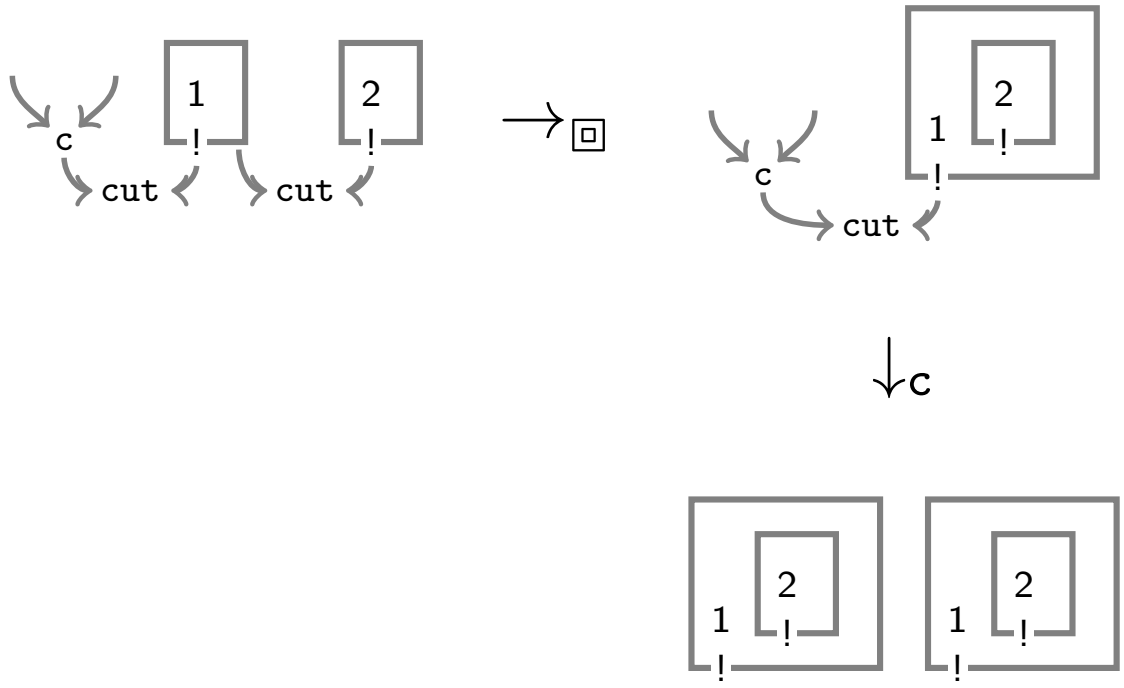


No sub-term property:

The second step duplicates a box not in the initial proof net.

Linear Logic Strategies

Reduction at level 0:



No sub-term property:

The second step duplicates a box not in the initial proof net.

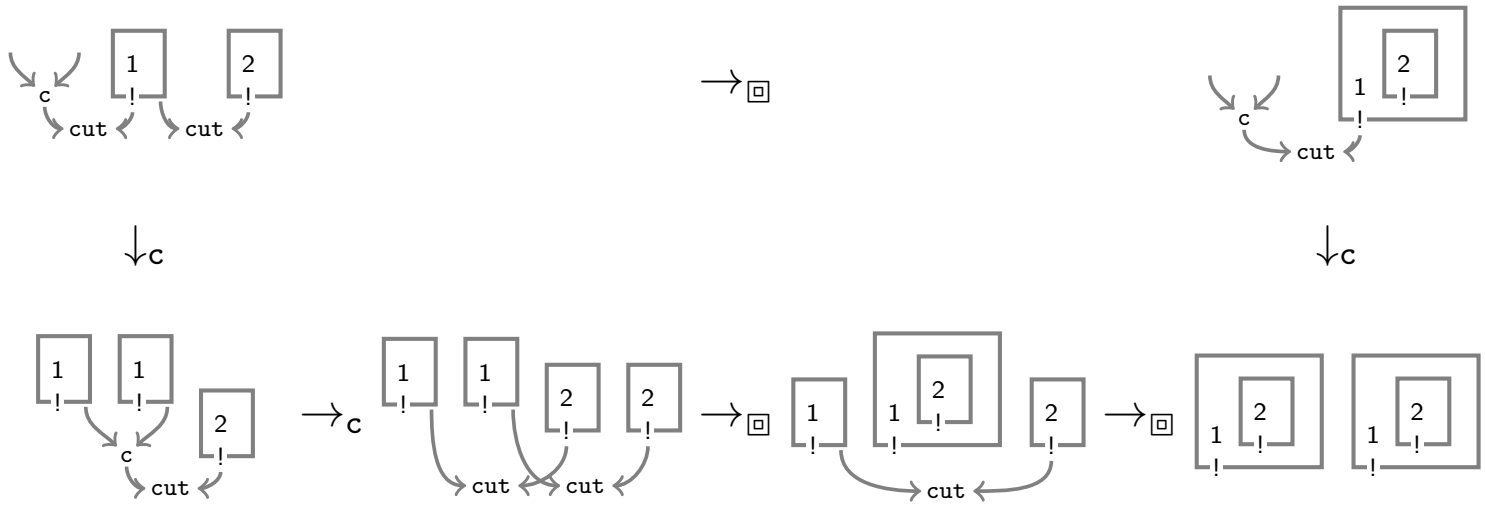
Linear Logic Strategies

Let's close the diagram:

Further problem:
Reduction at level 0 is not diamond.

Linear Logic Strategies

Let's close the diagram:

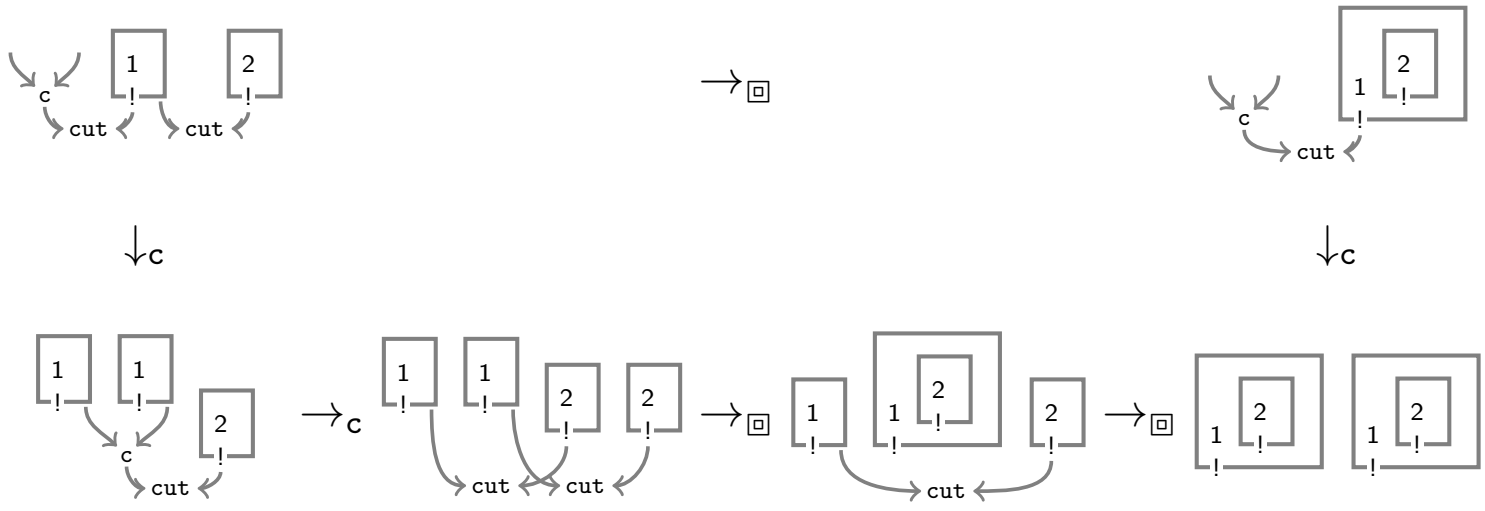


Further problem:

Reduction at level 0 is not diamond.

Linear Logic Strategies

Let's close the diagram:

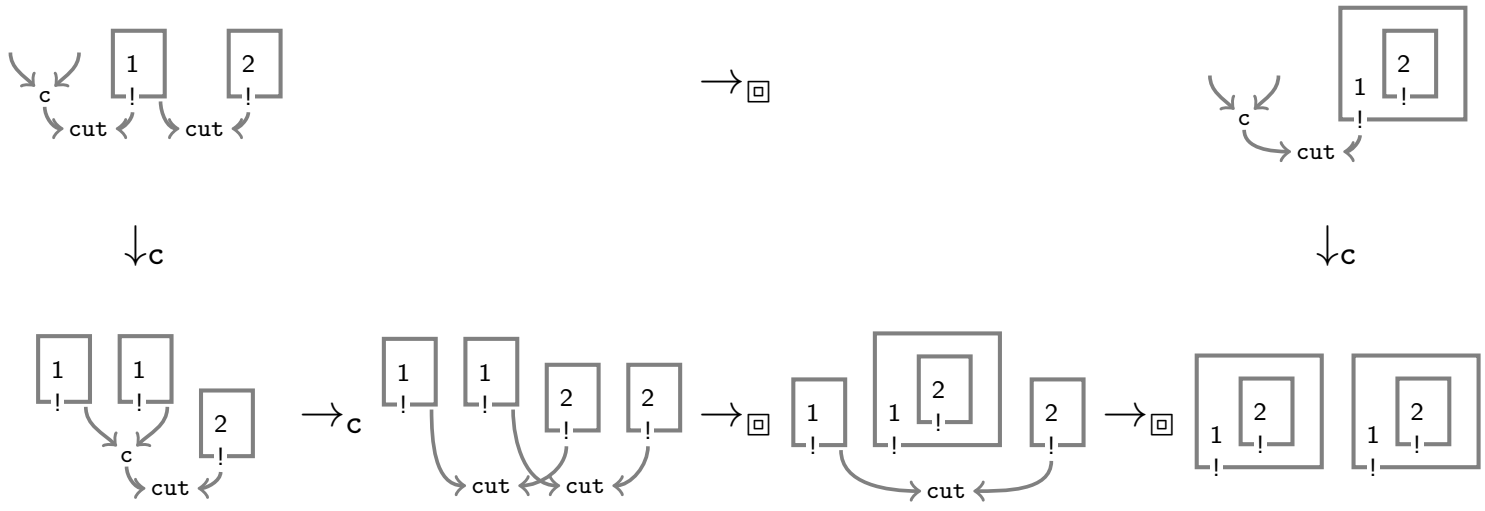


Further problem:

Reduction at level 0 is not diamond.

Linear Logic Strategies

Let's close the diagram:

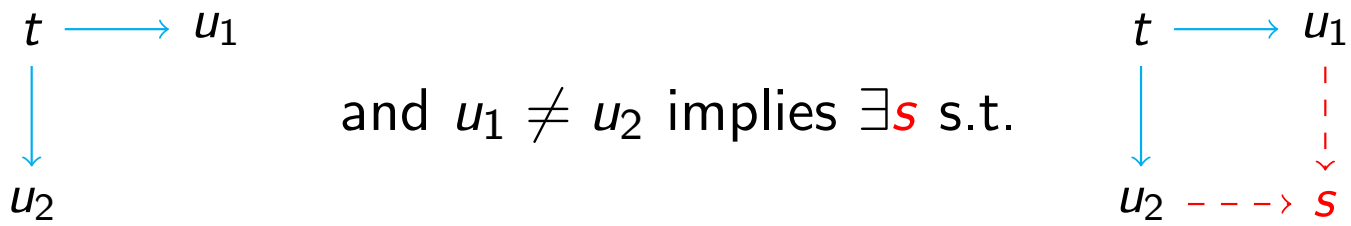


Further problem:

Reduction at level 0 is not diamond.

Diamond

A cost model strategy is expected to be deterministic or diamond:

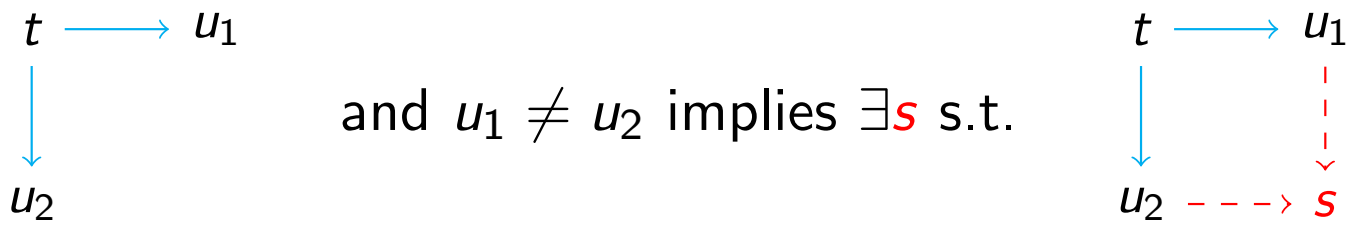


Roughly, diamond = choices do not impact evaluation lengths.

Otherwise, the number of steps is not a meaningful measure.

Diamond

A cost model strategy is expected to be deterministic or diamond:

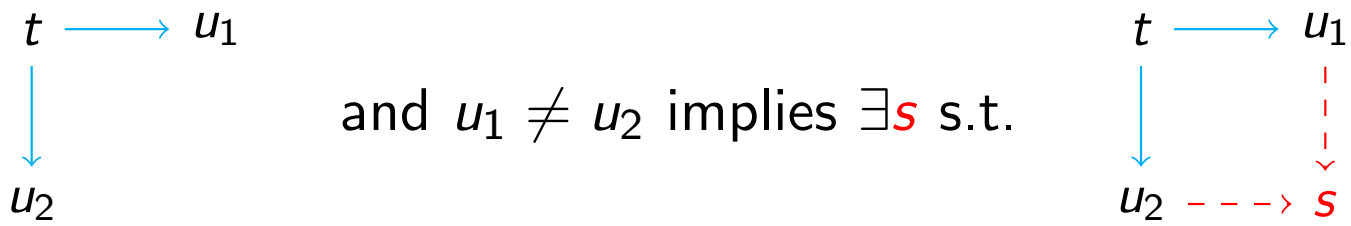


Roughly, diamond = choices do not impact evaluation lengths.

Otherwise, the number of steps is not a meaningful measure.

Diamond

A cost model strategy is expected to be deterministic or diamond:



Roughly, diamond = choices do not impact evaluation lengths.

Otherwise, the number of steps is not a meaningful measure.

Linear Head Reduction

Another recurrent strategy in the literature is linear head reduction.

By Danos & Regnier and Mascari & Pedicini.

Reduce only cuts at level 0, but not those involving box borders.

Linear Head Reduction

Another recurrent strategy in the literature is linear head reduction.

By Danos & Regnier and Mascari & Pedicini.

Reduce only cuts at level 0, but not those involving box borders.

Linear Head Reduction

Another recurrent strategy in the literature is linear head reduction.

By Danos & Regnier and Mascari & Pedicini.

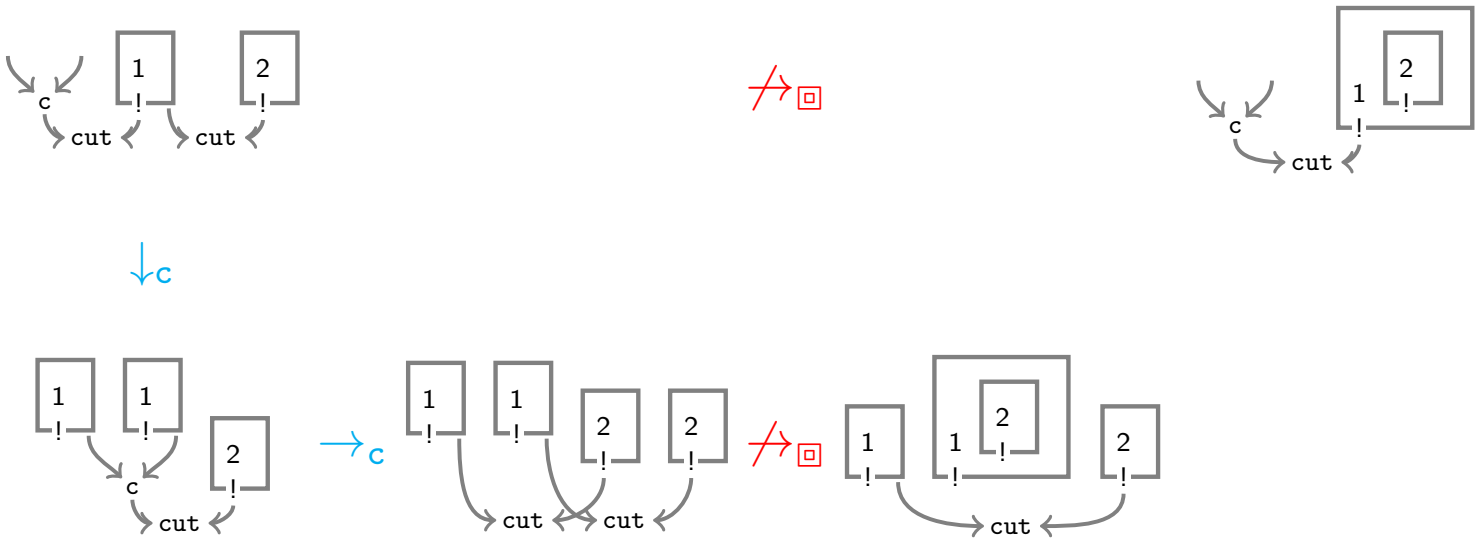
Reduce only cuts at level 0, but not those involving box borders.

Linear Head Reduction

The previous example with linear head reduction

Linear Head Reduction

The previous example with linear head reduction



Linear Head Reduction

Deep connections with games, GoI, intersection types, π-calculus,...

The linear head strategy is diamond and has the sub-term property.

Trivially, because it never touches any box.

Linear Head Reduction

Deep connections with games, GoI, intersection types, π-calculus,...

The linear head strategy is diamond and has the sub-term property.

Trivially, because it never touches any box.

Linear Head Reduction

Deep connections with games, GoI, intersection types, π-calculus,...

The linear head strategy is diamond and has the sub-term property.

Trivially, because it never touches any box.

Linear Head Reduction

Linear head is not a strategy for LL.

It is defined only for (proofs representing) λ-terms.

Even then, linear head does not compute cut-free proofs.

We need a good generalization of the linear head strategy.

Far from obvious!

Linear Head Reduction

Linear head is not a strategy for LL.

It is defined only for (proofs representing) λ-terms.

Even then, linear head does not compute cut-free proofs.

We need a good generalization of the linear head strategy.

Far from obvious!

Linear Head Reduction

Linear head is not a strategy for LL.

It is defined only for (proofs representing) λ-terms.

Even then, linear head does not compute cut-free proofs.

We need a good generalization of the linear head strategy.

Far from obvious!

Linear Head Reduction

Linear head is not a strategy for LL.

It is defined only for (proofs representing) λ-terms.

Even then, linear head does not compute cut-free proofs.

We need a good generalization of the linear head strategy.

Far from obvious!

Linear Head Reduction

Linear head is not a strategy for LL.

It is defined only for (proofs representing) λ-terms.

Even then, linear head does not compute cut-free proofs.

We need a good generalization of the linear head strategy.

Far from obvious!

Linear Head, the LSC, and LL

The LSC models linear head reduction elegantly.

Two possible routes:

1. Generalize LHR using proof nets.
2. Generalize the LSC to linear logic.

We follow 2.

Linear Head, the LSC, and LL

The LSC models linear head reduction elegantly.

Two possible routes:

1. Generalize LHR using proof nets.
2. Generalize the LSC to linear logic.

We follow 2.

Linear Head, the LSC, and LL

The LSC models linear head reduction elegantly.

Two possible routes:

1. Generalize LHR using proof nets.
2. Generalize the LSC to linear logic.

We follow 2.

Linear Head, the LSC, and LL

The LSC models linear head reduction elegantly.

Two possible routes:

1. Generalize LHR using proof nets.
2. Generalize the LSC to linear logic.

We follow 2.

Linear Head, the LSC, and LL

The LSC models linear head reduction elegantly.

Two possible routes:

1. Generalize LHR using proof nets.
2. Generalize the LSC to linear logic.

We follow 2.

Linear Head, the LSC, and LL

Reasons:

1. Graphs are not handy for complex proofs.
2. LSC rewriting behaves way better than proof nets rewriting.
3. Fresh look at LL, importing ideas from sibling field.

Linear Head, the LSC, and LL

Reasons:

1. Graphs are not handy for complex proofs.
2. LSC rewriting behaves way better than proof nets rewriting.
3. Fresh look at LL, importing ideas from sibling field.

Linear Head, the LSC, and LL

Reasons:

1. Graphs are not handy for complex proofs.
2. LSC rewriting behaves way better than proof nets rewriting.
3. Fresh look at LL, importing ideas from sibling field.

Linear Head, the LSC, and LL

Reasons:

1. Graphs are not handy for complex proofs.
2. LSC rewriting behaves way better than proof nets rewriting.
3. Fresh look at LL, importing ideas from sibling field.

Linear Head, the LSC, and LL

Reasons:

1. Graphs are not handy for complex proofs.
2. LSC rewriting behaves way better than proof nets rewriting.
3. Fresh look at LL, importing ideas from sibling field.

Outline

Termination of Substitutions

Key property of explicit substitutions:

Substitution rules alone always terminate.

Issue with Classical MELL: untyped exponentials do not terminate.

Not even weakly.

Termination of Substitutions

Key property of explicit substitutions:

Substitution rules alone always terminate.

Issue with Classical MELL: untyped exponentials do not terminate.

Not even weakly.

Termination of Substitutions

Key property of explicit substitutions:

> Substitution rules alone always terminate.

Issue with Classical MELL: untyped exponentials do not terminate.

Not even weakly.

Termination of Substitutions

Key property of explicit substitutions:

Substitution rules alone always terminate.

Issue with Classical MELL: untyped exponentials do not terminate.

Not even weakly.

Classical Exponentials are not Explicit Substitutions

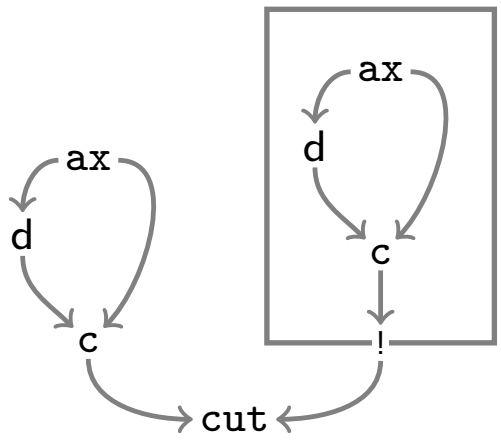
The following proof net reduces to itself in 3 exponential steps:

It is an exponentials-only variant of $\delta\delta$.

Logical insight: involutive negation is dangerous!

Classical Exponentials are not Explicit Substitutions

The following proof net reduces to itself in 3 exponential steps:

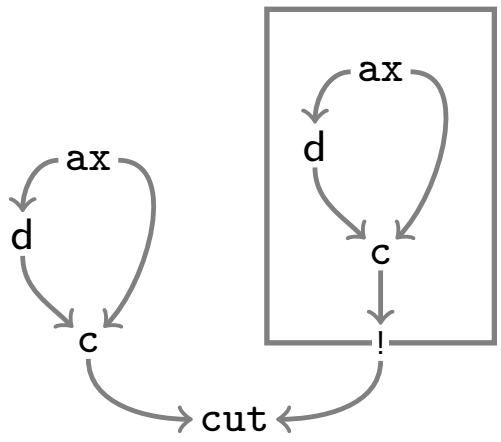


It is an exponentials-only variant of $\delta\delta$.

Logical insight: involutive negation is dangerous!

Classical Exponentials are not Explicit Substitutions

The following proof net reduces to itself in 3 exponential steps:

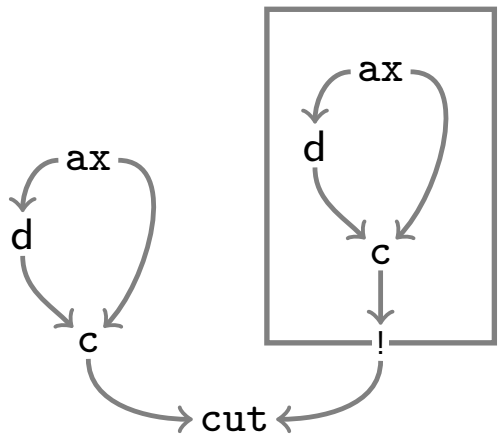


It is an exponentials-only variant of $\delta\delta$.

Logical insight: involutive negation is dangerous!

Classical Exponentials are not Explicit Substitutions

The following proof net reduces to itself in 3 exponential steps:



It is an exponentials-only variant of $\delta\delta$.

Logical insight: involutive negation is dangerous!

Consequence 2: Intuitionism

Intuitionistic LL: untyped exponentials terminate.

Then we place ourselves in ILL.

Leaving additives aside, we actually study IMELL.

Straightforward additives via slices.

Consequence 2: Intuitionism

Intuitionistic LL: untyped exponentials terminate.

Then we place ourselves in ILL.

Leaving additives aside, we actually study IMELL.

Straightforward additives via slices.

Consequence 2: Intuitionism

Intuitionistic LL: untyped exponentials terminate.

Then we place ourselves in ILL.

Leaving additives aside, we actually study IMELL.

Straightforward additives via slices.

Consequence 2: Intuitionism

Intuitionistic LL: untyped exponentials terminate.

Then we place ourselves in ILL.

Leaving additives aside, we actually study IMELL.

Straightforward additives via slices.

Exponential Substitution Calculus

Solution for IMELL.

ESC = an untyped LSC-like calculus with IMELL as typing system.

It has a strategy generalizing LHR and reaching cut-free proofs.

With the sub-term property $\Rightarrow$ polynomial cost model for IMELL.

Exponential Substitution Calculus

Solution for IMELL.

ESC = an untyped LSC-like calculus with IMELL as typing system.

It has a strategy generalizing LHR and reaching cut-free proofs.

With the sub-term property $\Rightarrow$ polynomial cost model for IMELL.

Exponential Substitution Calculus

Solution for IMELL.

ESC = an untyped LSC-like calculus with IMELL as typing system.

It has a strategy generalizing LHR and reaching cut-free proofs.

With the sub-term property $\Rightarrow$ polynomial cost model for IMELL.

Exponential Substitution Calculus

Solution for IMELL.

ESC = an untyped LSC-like calculus with IMELL as typing system.

It has a strategy generalizing LHR and reaching cut-free proofs.

With the sub-term property $\Rightarrow$ polynomial cost model for IMELL.

Exponential Substitution Calculus

Solution for IMELL.

ESC = an untyped LSC-like calculus with IMELL as typing system.

It has a strategy generalizing LHR and reaching cut-free proofs.

With the sub-term property $\Rightarrow$ polynomial cost model for IMELL.

THANKS!