
Do Light Logics allow a unified view of Stratification and Boundedness?

M. Gaboardi, L. Roversi, L. Vercelli
Università di Torino

July 22th 2009

Linear Logic and Light Logics

MELL

Light Logics

Targets

Stratification and Boundedness

LLL and **SLL**

LLL: Stratification

SLL: Boundedness

Bounded **LLL**

Stratified **SLL**

A hierarchy that may exist

The First Step

Embedding **MELL** in **ML**³

Conclusions

Linear Logic and
Light Logics

Stratification and
Boundedness

The First Step

Conclusions

Linear Logic and Light Logics

Linear Logic and
Light Logics

MELL

Light Logics

Targets

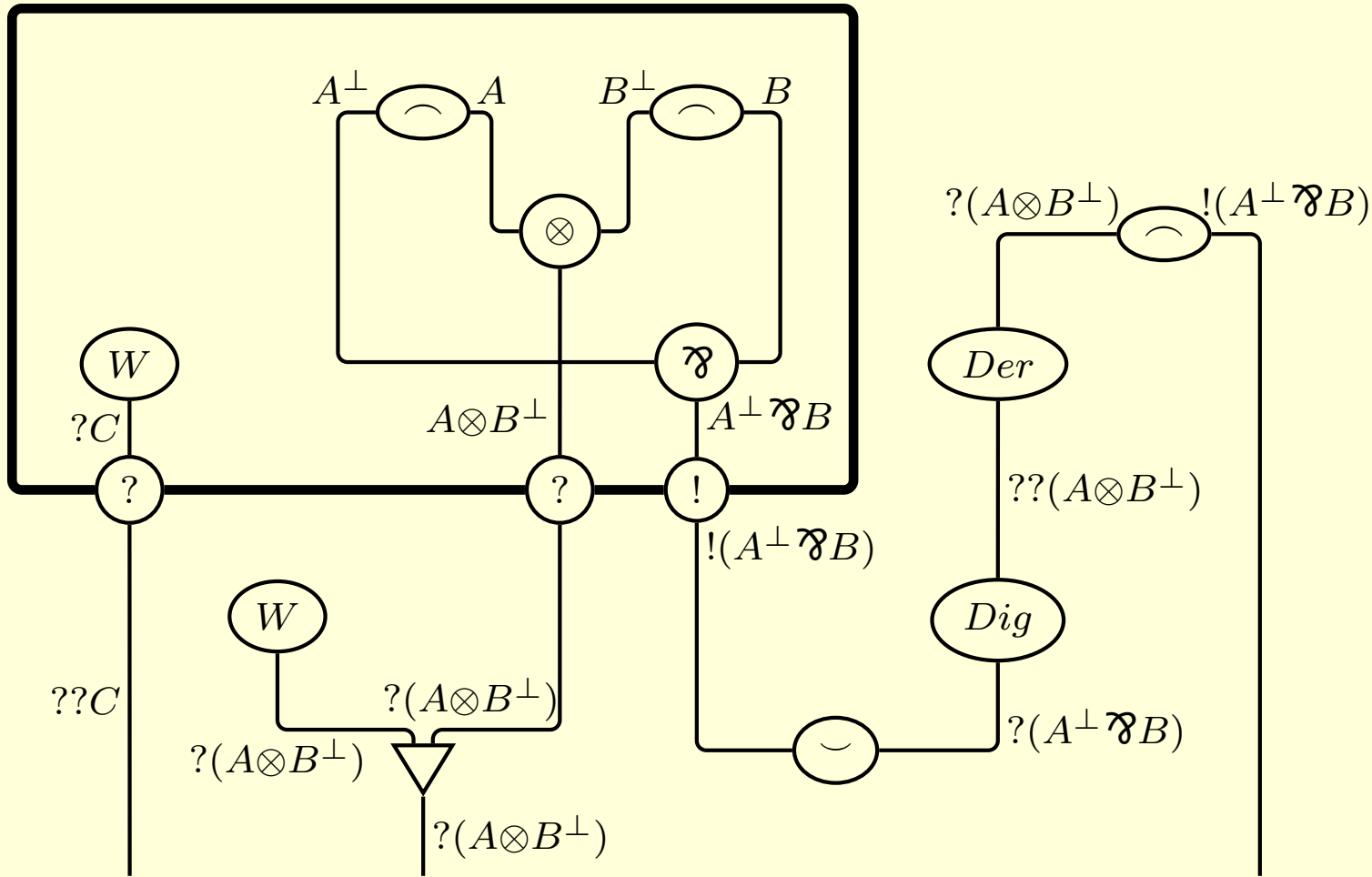
Stratification and
Boundedness

The First Step

Conclusions

Multiplicative Exponential Linear Logic

An example of proof net:



Linear Logic and
Light Logics

MELL

Light Logics
Targets

Stratification and
Boundedness

The First Step

Conclusions

Light Logics Characterize Complexity Classes

Programs	\leftrightarrow	Proofs
Types of Data	\leftrightarrow	Formulas
Execution of Programs	\leftrightarrow	Cut-elimination
Values	\leftrightarrow	Cut-free proofs
Classes of Programs	\leftrightarrow	Classes of Proofs (=Light Logics)

Linear Logic and
Light Logics

MELL

Light Logics

Targets

Stratification and
Boundedness

The First Step

Conclusions

Theorem 1 (Soundness). *There exists a family of polynomials $\{p_d(x) \mid d \in \mathbb{N}\}$ such that every proof net Π of **LLL** is reduced in $p_{d(\Pi)}(|\Pi|)$ steps.
The same statement holds for **SLL**.*

Theorem 2 (FPTIME Completeness). *Every Turing Machine computing a function in polynomial time can be represented in **LLL**.*

Theorem 3 (PTIME Completeness). *Every Turing Machine solving a decision problem in polynomial time can be represented in **SLL**.*

Linear Logic and
Light Logics

MELL

Light Logics

Targets

Stratification and
Boundedness

The First Step

Conclusions

- Our long-term target: a polytime system **SOLI** (soft-light logic) that subsumes **LLL** and **SLL**.
- Today's target: a comparison between **LLL** and **SLL**.

[Linear Logic and
Light Logics](#)

[MELL](#)

[Light Logics](#)

[Targets](#)

[Stratification and
Boundedness](#)

[The First Step](#)

[Conclusions](#)

Stratification and Boundedness

Linear Logic and
Light Logics

**Stratification and
Boundedness**

LLL and **SLL**

LLL: Stratification

SLL: Boundedness

Bounded **LLL**

Stratified **SLL**

A hierarchy that
may exist

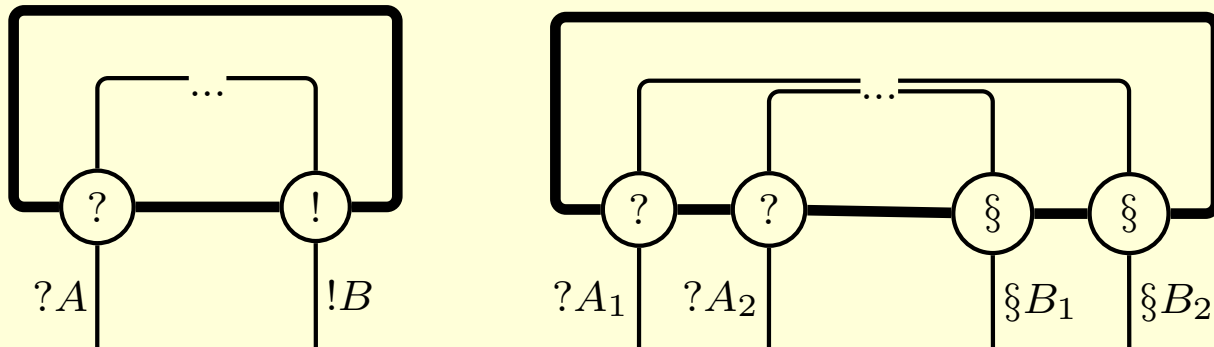
The First Step

Conclusions

Three restrictions on **MELL**:

- No Dereliction
- No Digging
- Boxes have at most one ?-port.

To gain again some expressiveness, one more modality \S .



[Linear Logic and Light Logics](#)

[Stratification and Boundedness](#)

LLL and SLL

LLL: Stratification

SLL: Boundedness

Bounded **LLL**

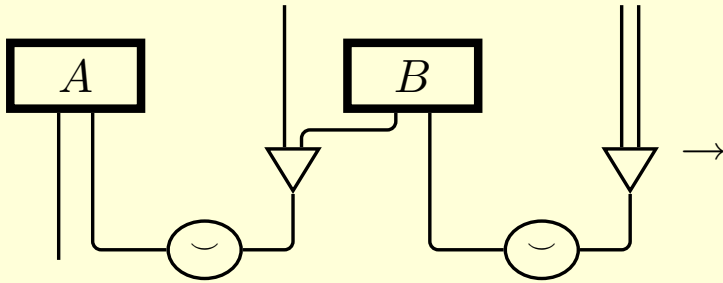
Stratified **SLL**

A hierarchy that may exist

[The First Step](#)

[Conclusions](#)

A Typical Behaviour of LLL



Linear Logic and
Light Logics

Stratification and
Boundedness

LLL and SLL

LLL: Stratification

SLL: Boundedness

Bounded **LLL**

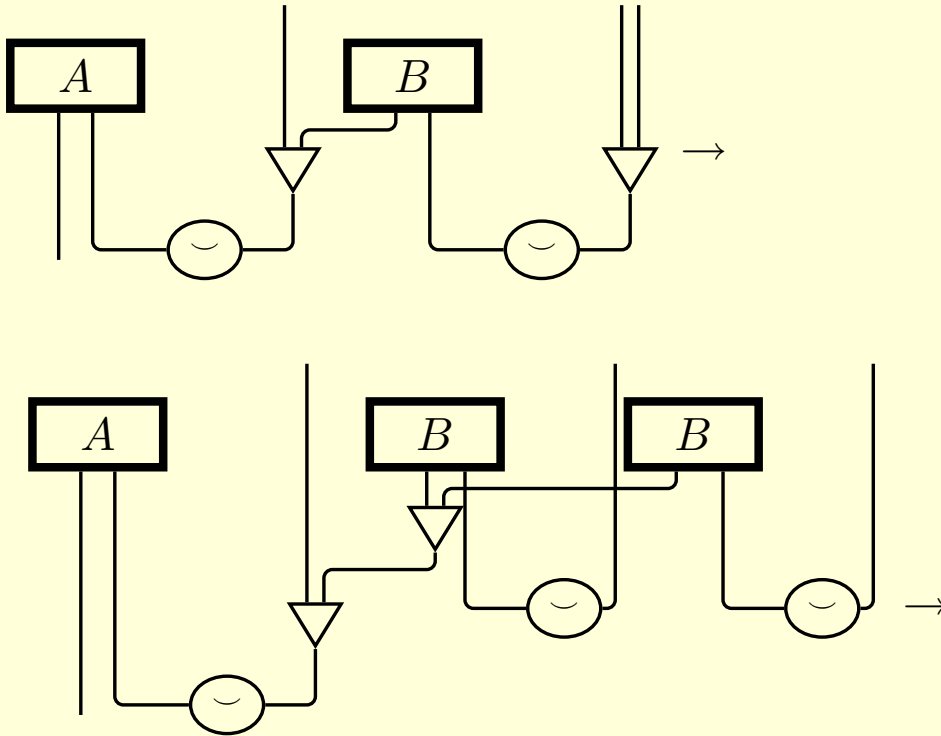
Stratified **SLL**

A hierarchy that
may exist

The First Step

Conclusions

A Typical Behaviour of LLL



Linear Logic and
Light Logics

Stratification and
Boundedness

LLL and SLL

LLL: Stratification

SLL: Boundedness

Bounded **LLL**

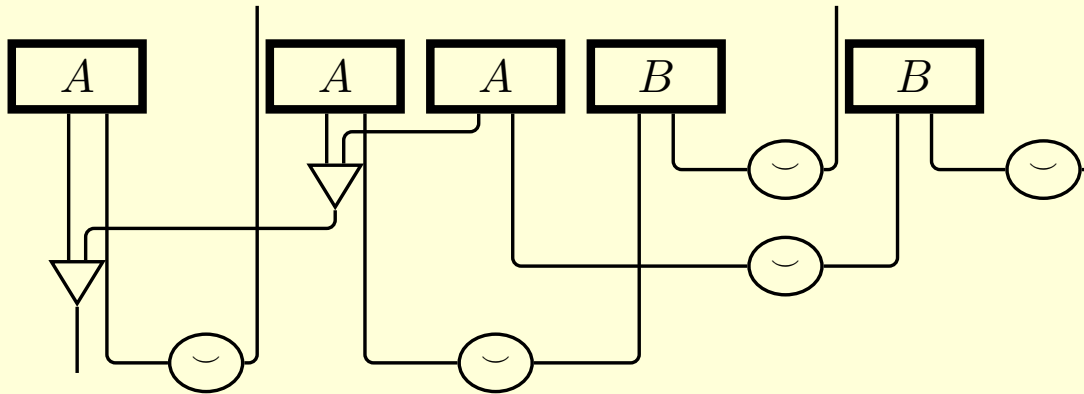
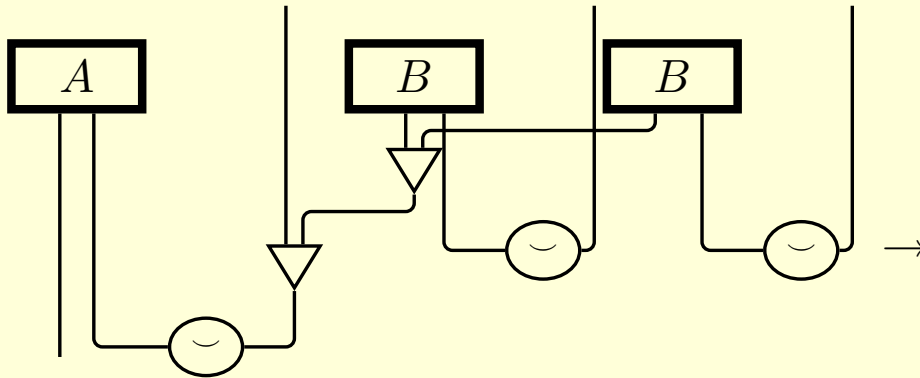
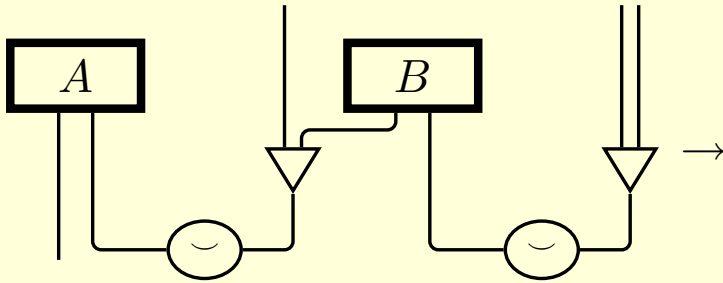
Stratified **SLL**

A hierarchy that
may exist

The First Step

Conclusions

A Typical Behaviour of LLL



Linear Logic and
Light Logics

Stratification and
Boundedness

LLL and SLL

LLL: Stratification

SLL: Boundedness

Bounded LLL

Stratified SLL

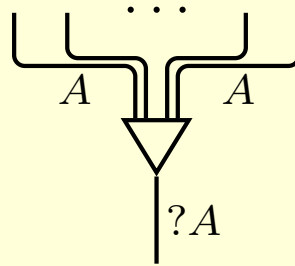
A hierarchy that
may exist

The First Step

Conclusions

Two restrictions on **MELL**:

- No Digging
- Multiplexor instead of Contraction / Dereliction.



[Linear Logic and Light Logics](#)

[Stratification and Boundedness](#)

LLL and SLL

LLL: Stratification

SLL: Boundedness

Bounded **LLL**

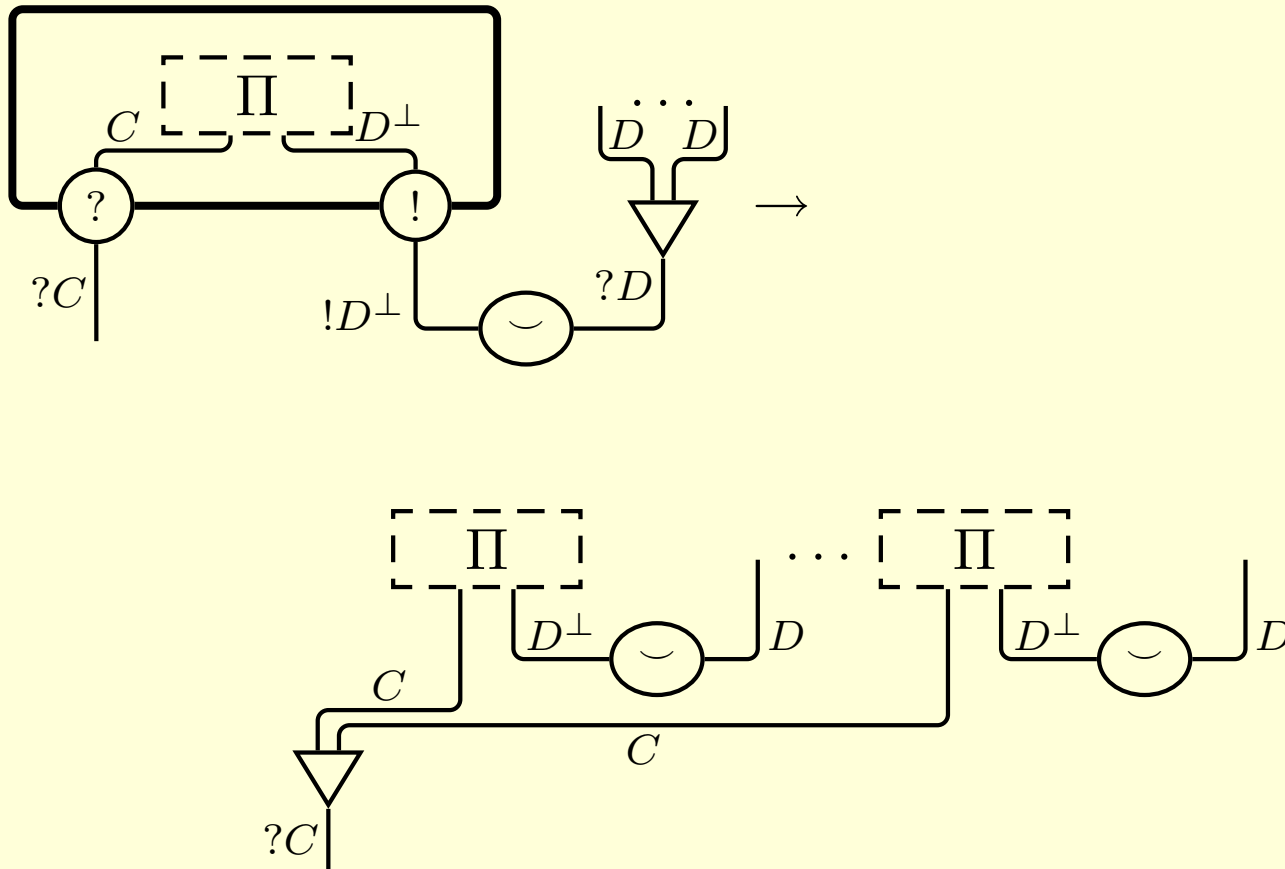
Stratified **SLL**

A hierarchy that may exist

[The First Step](#)

[Conclusions](#)

A Typical Behaviour of SLL



Linear Logic and
Light Logics

Stratification and
Boundedness

LLL and SLL

LLL: Stratification

SLL: Boundedness

Bounded LLL

Stratified SLL

A hierarchy that
may exist

The First Step

Conclusions

- **LLL**: without digging/dereliction, the nodes do never change their depth during reduction.
- When duplicating a box, the two new boxes have the same depth of the first one.
- This is **stratification**

Linear Logic and
Light Logics

Stratification and
Boundedness

LLL and **SLL**

LLL: Stratification

SLL: Boundedness

Bounded **LLL**

Stratified **SLL**

A hierarchy that
may exist

The First Step

Conclusions

- **LLL**: without digging/dereliction, the nodes do never change their depth during reduction.
- When duplicating a box, the two new boxes have the same depth of the first one.
- This is [stratification](#)

- **SLL**: without digging, the nodes do never increase their depth during reduction.
- When n -uplicating a box, the box disappear; the n new copies have depth less than the first one.
- So, **SLL** is [not stratified](#).

[Linear Logic and Light Logics](#)

[Stratification and Boundedness](#)

[LLL and SLL](#)

[LLL: Stratification](#)

[SLL: Boundedness](#)

[Bounded LLL](#)

[Stratified SLL](#)

[A hierarchy that may exist](#)

[The First Step](#)

[Conclusions](#)

- **SLL**: we know *a priori* how many times a box will be duplicated.
- This constant is called **rank**.
- We call **boundedness** this property.

[Linear Logic and Light Logics](#)

[Stratification and Boundedness](#)

LLL and SLL

LLL: Stratification

SLL: Boundedness

Bounded **LLL**

Stratified **SLL**

A hierarchy that may exist

[The First Step](#)

[Conclusions](#)

- **SLL**: we know *a priori* how many times a box will be duplicated.
- This constant is called **rank**.
- We call **boundedness** this property.

- **LLL**: The *typical behaviour* shows that **LLL** does not enjoy boundedness.

[Linear Logic and Light Logics](#)

[Stratification and Boundedness](#)

LLL and SLL

LLL: Stratification

SLL: Boundedness

Bounded **LLL**

Stratified **SLL**

A hierarchy that may exist

[The First Step](#)

[Conclusions](#)

LLL as a bounded system?

- How many copies of the box b will be created?
- Context semantics defines paths over the proof nets
- This gives a bound: count the paths that start from b .
 $R(b)$ paths means at most $R(b)$ copies of b

Linear Logic and
Light Logics

Stratification and
Boundedness

LLL and SLL

LLL: Stratification

SLL: Boundedness

Bounded LLL

Stratified **SLL**

A hierarchy that
may exist

The First Step

Conclusions

SLL as a stratified system?

- **Idea:** **SLL** is not stratified as **LLL**, but it could be stratified in a weaker sense.

Linear Logic and
Light Logics

Stratification and
Boundedness

LLL and SLL

LLL: Stratification

SLL: Boundedness

Bounded **LLL**

Stratified SLL

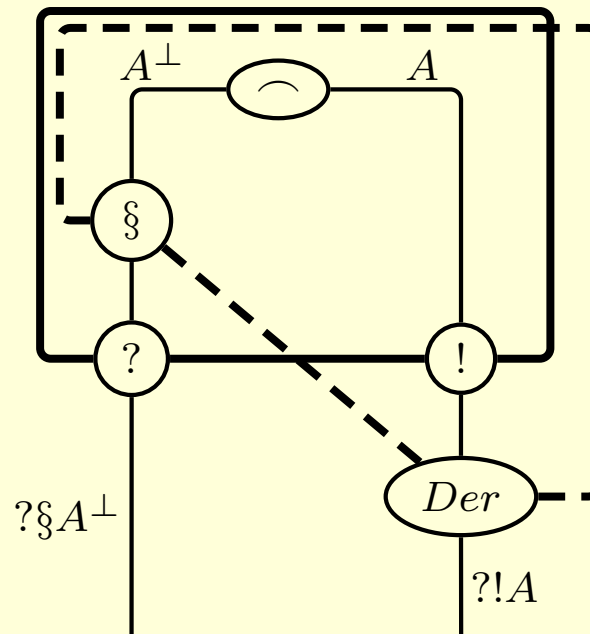
A hierarchy that
may exist

The First Step

Conclusions

SLL as a stratified system?

- **Idea:** **SLL** is not stratified as **LLL**, but it could be stratified in a weaker sense.
- **ML³** is a restriction of **MELL**, with \S
- Derelictions, diggings, paragraphs must form **fuzzy boxes**:



- **ML³** enjoys a different notion of stratification

[Linear Logic and Light Logics](#)

[Stratification and Boundedness](#)

LLL and SLL

LLL: Stratification

SLL: Boundedness

Bounded LLL

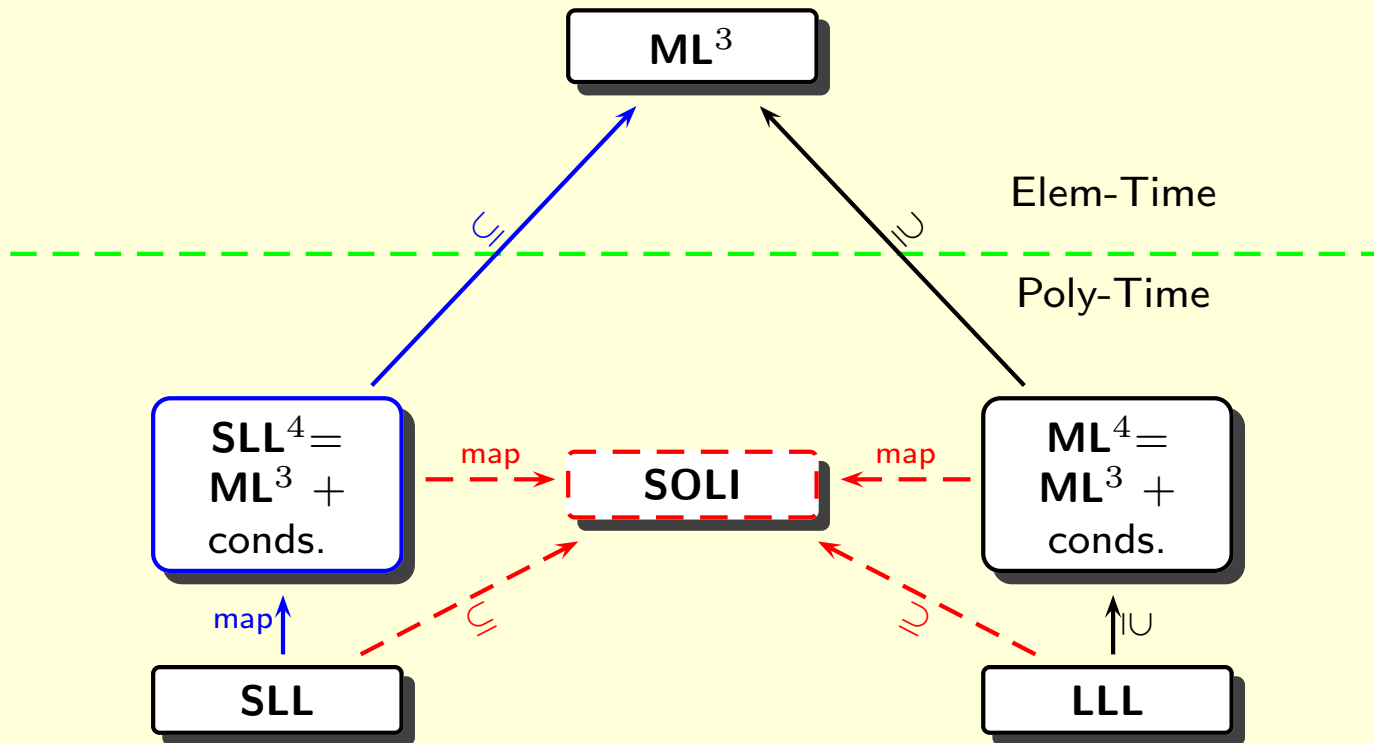
Stratified SLL

A hierarchy that may exist

[The First Step](#)

[Conclusions](#)

A hierarchy that may exist



Linear Logic and
Light Logics

Stratification and
Boundedness

LLL and **SLL**

LLL: Stratification

SLL: Boundedness

Bounded **LLL**

Stratified **SLL**

A hierarchy that
may exist

The First Step

Conclusions

Linear Logic and
Light Logics

Stratification and
Boundedness

The First Step
Embedding MELL in
ML³

Conclusions

The First Step

Theorem 4 (GRV, 2009). *There is an algorithm $@(\cdot)$ that takes every proof net Π of **propositional MELL**, and returns a proof net $@(\Pi)$ of **ML³**.
The proof nets Π and $@(\Pi)$ only differ for the possible presence of some new paragraph nodes.
The algorithm respects the cut elimination procedure.*

This holds in particular for **propositional SLL**.

[Linear Logic and Light Logics](#)

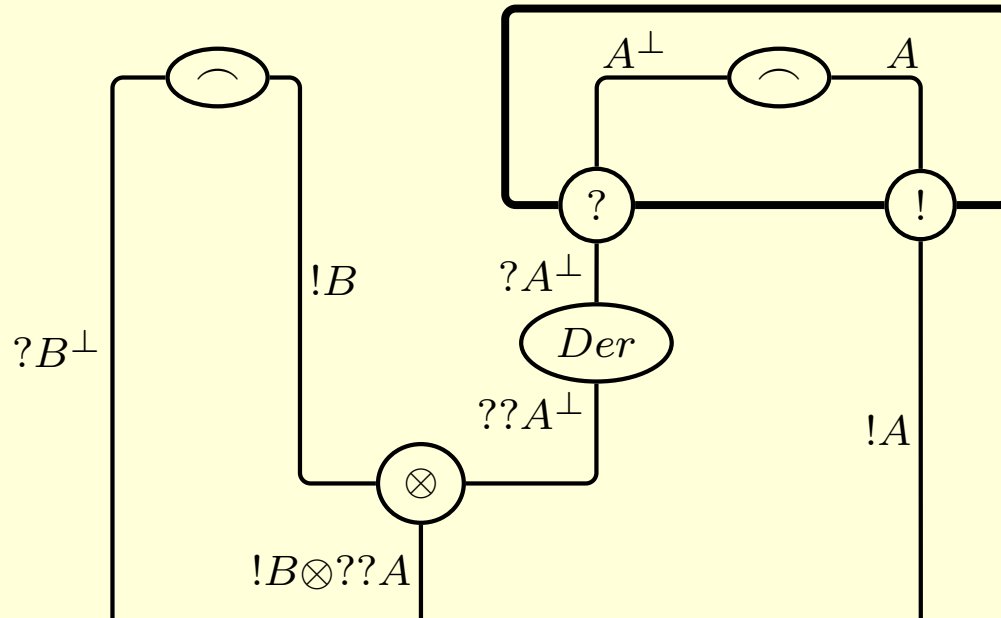
[Stratification and Boundedness](#)

[The First Step](#)

[Embedding MELL in ML³](#)

[Conclusions](#)

An example:



[Linear Logic and Light Logics](#)

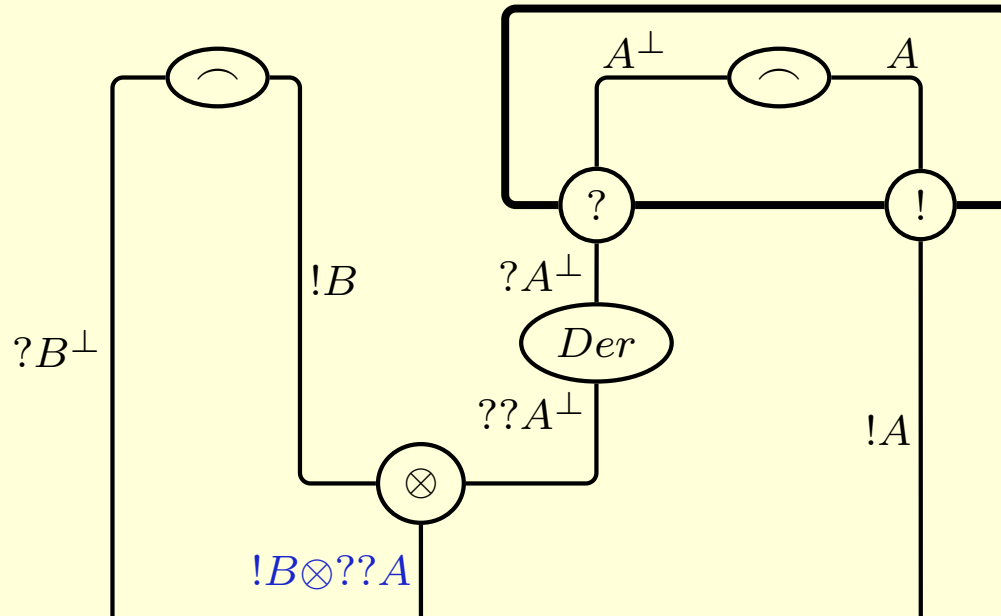
[Stratification and Boundedness](#)

[The First Step](#)

[Embedding MELL in ML³](#)

[Conclusions](#)

An example:



The **formula level** is at most 2

[Linear Logic and Light Logics](#)

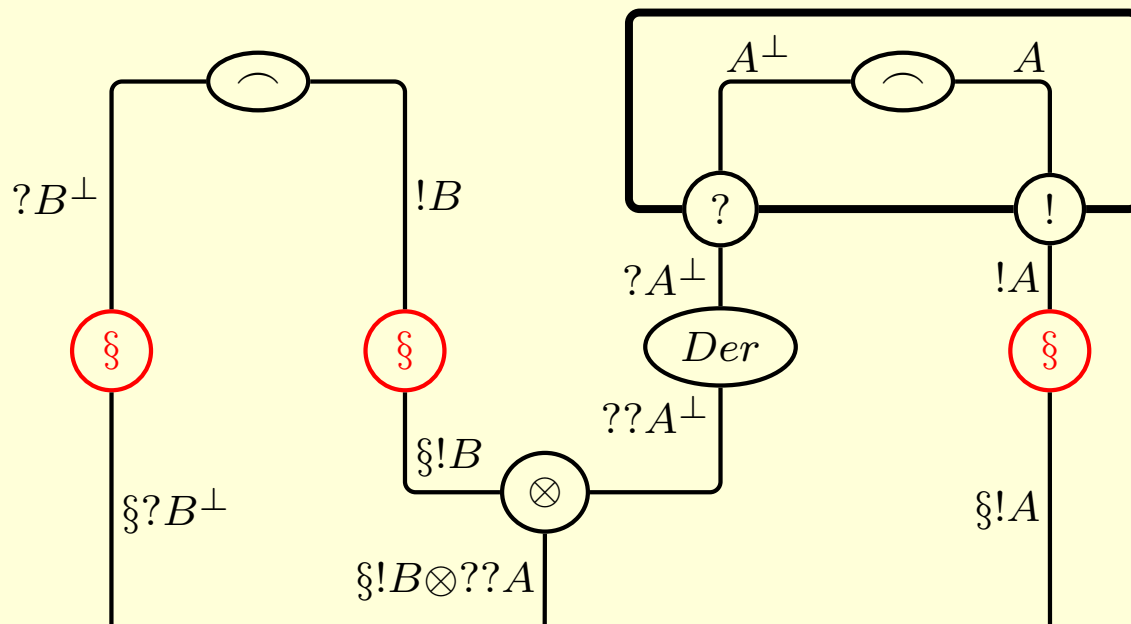
[Stratification and Boundedness](#)

[The First Step](#)

[Embedding MELL in ML³](#)

[Conclusions](#)

An example:



Linear Logic and
Light Logics

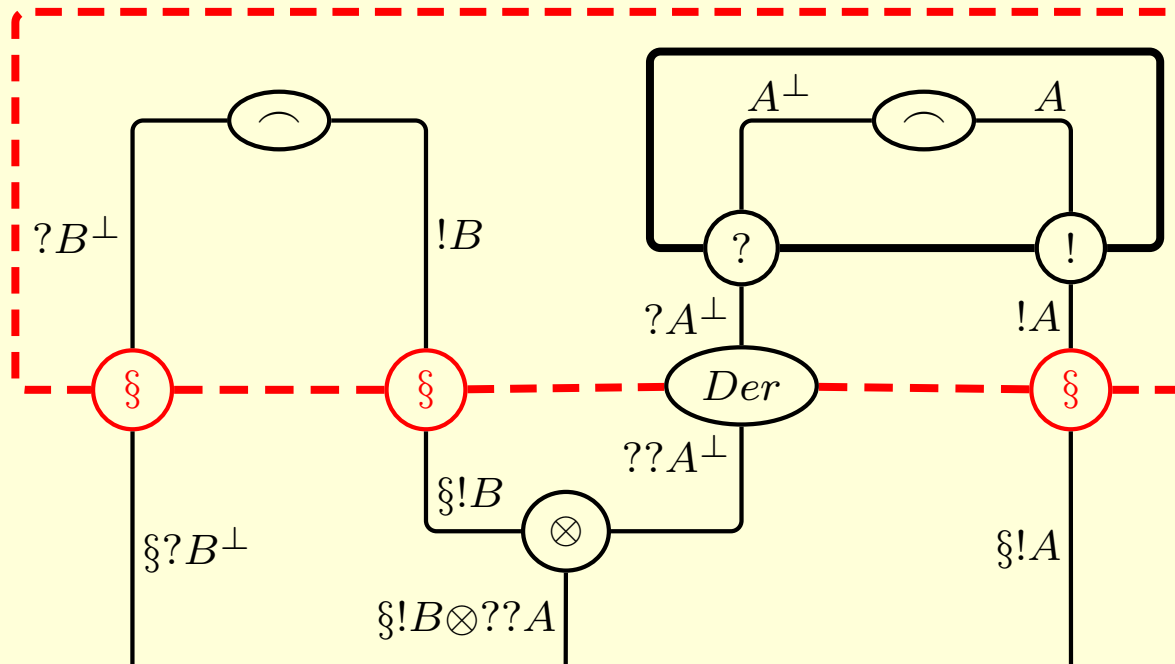
Stratification and
Boundedness

The First Step

Embedding MELL in
ML³

Conclusions

An example:



In this example, it is quite easy to find where to put the paragraph nodes. In general, it is not.

Linear Logic and
Light Logics

Stratification and
Boundedness

The First Step

Embedding MELL in
ML³

Conclusions

This algorithm **does not work** in presence of second order quantification.

The more, **it is not possible** to stratify a generic **MELL** proof net.

[Linear Logic and Light Logics](#)

[Stratification and Boundedness](#)

[The First Step](#)

[Embedding MELL in ML³](#)

[Conclusions](#)

Linear Logic and
Light Logics

Stratification and
Boundedness

The First Step

Conclusions

Conclusions

Summing Up

- Stratification and Boundedness are two principles that allow controlling complexity
- They are different, but not so much

To do (in order)

- Extend Theorem 4 to the whole **SLL**.
- Find a *simple* proof of polynomial time soundness that holds for both **LLL** and **SLL**.
- Find a system **SOLI** that extend both **LLL** and **SLL**.

THANK YOU!

Linear Logic and
Light Logics

Stratification and
Boundedness

The First Step

Conclusions

