The Geometry of Interaction Program
Proofs, Programs, Operators and Algorithmic Complexity

Thomas Seiller

Séminaire 68nqrt, IRISA, Rennes, June 13th 2013

The Geometry of Interaction Program

Proofs, Programs, Graphs and Algorithmic Complexity

Thomas Seiller

Séminaire 68nqrt, IRISA, Rennes, June 13th 2013

1. From proofs to graphs
2. Interaction Graphs
3. Logarithmic Space

From proofs to graphs

Linear Logic

Studying models of the λ-calculus, Girard realized that the implication $A \to B$ is decomposed as:

$$!A \multimap B$$

- $\multimap$ is a *linear* implication — which uses its argument exactly once;
- ! is a modality allowing the duplication;

He then introduced linear logic to mirror these semantics remarks in the syntax. The restriction to linear implication induces a splitting of the conjunction and disjunction into *multiplicative* ($\otimes$, $\parr$) and *additive* ($\&$, $\oplus$) variants.

Example

$$\dfrac{}{\vdash A, A^{\perp}}\ \mathrm{ax} \qquad \dfrac{\vdash \Gamma, A \qquad \vdash \Delta, A^{\perp}}{\vdash \Gamma, \Delta}\ \mathrm{cut}$$

$$\dfrac{\vdash \Gamma, A \qquad \vdash \Gamma, B}{\vdash \Gamma, \Delta, A \otimes B}\ \otimes \qquad \dfrac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}\ \parr$$

Example

$$\dfrac{A \vdash B \qquad \dfrac{\dfrac{}{A \vdash A}\,\text{ax} \qquad \dfrac{}{B \vdash B}\,\text{ax}}{A, B \vdash A \wedge B}\,\wedge}{\dfrac{\dfrac{A, A \vdash A \wedge B}{A \vdash A \wedge B}\,\text{ctr}}{\vdash A \to A \wedge B}\,\to}\,\text{cut}$$

Example

$$
\dfrac{\dfrac{A \vdash B \qquad \dfrac{\dfrac{}{A \vdash A}\,\mathrm{ax} \qquad \dfrac{}{B \vdash B}\,\mathrm{ax}}{A, B \vdash A \otimes B}\,\otimes}{\dfrac{A, A \vdash A \otimes B}{\color{red}{A \vdash A \otimes B}}\,\color{red}{\mathrm{ctr}}}\,\mathrm{cut}}{\color{red}{\vdash A \to A \otimes B}}\,\color{red}{\to}
$$

Example

$$
\dfrac{\dfrac{\dfrac{\dfrac{A \vdash B \qquad \dfrac{\dfrac{}{A \vdash A}\,\mathrm{ax} \qquad \dfrac{}{B \vdash B}\,\mathrm{ax}}{A, B \vdash A \otimes B}\,\otimes}{A, A \vdash A \otimes B}\,\mathrm{cut}}{!A, !A \vdash A \otimes B}\,\mathrm{der}}{\color{red}{A \vdash A \otimes B}}\,\mathrm{ctr}}{\color{red}{\vdash A \to A \otimes B}}\,\color{red}{\to}
$$

Example

$$
\begin{prooftree}
\AxiomC{$A \vdash B$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$A \vdash A$}
\AxiomC{}
\RightLabel{ax}
\UnaryInfC{$B \vdash B$}
\RightLabel{$\otimes$}
\BinaryInfC{$A, B \vdash A \otimes B$}
\RightLabel{cut}
\BinaryInfC{$A, A \vdash A \otimes B$}
\doubleLine
\RightLabel{der}
\UnaryInfC{$!A, !A \vdash A \otimes B$}
\RightLabel{ctr}
\UnaryInfC{$!A \vdash A \otimes B$}
\RightLabel{$\color{red}\rightarrow$}
\UnaryInfC{$\color{red}\vdash A \rightarrow A \otimes B$}
\end{prooftree}
$$

Example

$$
\dfrac{A \vdash B \qquad \dfrac{\dfrac{}{A \vdash A}\,\text{ax} \qquad \dfrac{}{B \vdash B}\,\text{ax}}{A, B \vdash A \otimes B}\,\otimes}{\dfrac{\dfrac{\dfrac{A, A \vdash A \otimes B}{!A, !A \vdash A \otimes B}\,\text{der}}{!A \vdash A \otimes B}\,\text{ctr}}{\vdash !A \multimap A \otimes B}\,\multimap}\,\text{cut}
$$

Restricting modalities

- Linear Logic allows a detailed control on duplication;
- Restricting the rules of the ! connective, one can define systems with less proofs which turn out to be interesting for the study of complexity.

For instance:
The ELL system (Elementary Linear Logic):

The proofs of type $!\mathrm{nat} \multimap \mathrm{nat}$ in ELL are exactly the functions computable in elementary time.

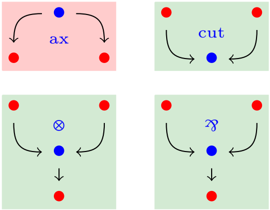
Restricting modalities

- Linear Logic allows a detailed control on duplication;
- Restricting the rules of the ! connective, one can define systems with less proofs which turn out to be interesting for the study of complexity.

For instance:
The LLL system (Light Linear Logic):

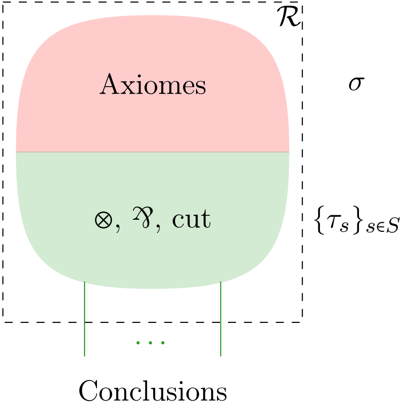
The proofs of type $!\mathrm{nat} \multimap \mathrm{nat}$ in LLL are exactly the functions computable in polynomial time.

Proof structures:



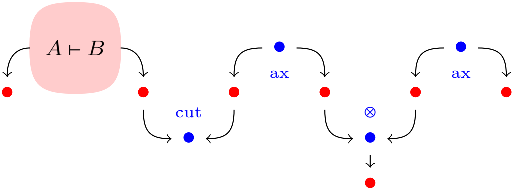
Théorème

$\mathcal{R}$ *is sequentializable*
iif
$\forall s \in S$, $\sigma\tau_s$ *is cyclic.*



Example

$$
\cfrac{A \vdash B \qquad \cfrac{\cfrac{}{A \vdash A}\,\text{ax} \qquad \cfrac{}{B \vdash B}\,\text{ax}}{A, B \vdash A \otimes B}\,\otimes}{\cfrac{A, A \vdash A \otimes B}{\cfrac{A \otimes A \vdash A \otimes B}{\vdash A \otimes A \multimap A \otimes B}\,\multimap}\,\otimes}\,\text{cut}
$$

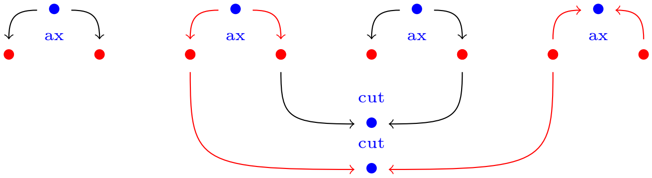


Geometry of Interaction: Interpretation of proofs

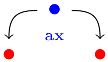
In a first approximation, the geometry of interaction program aims at defining a semantics of proofs that accounts for the dynamics of cut-elimination.

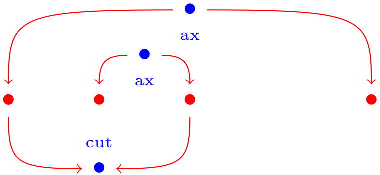
- A proof is interpreted by its set of axiom links;
- Cut-elimination corresponds to the computation of paths in the graph.

Example

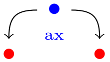


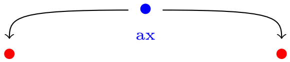
Example



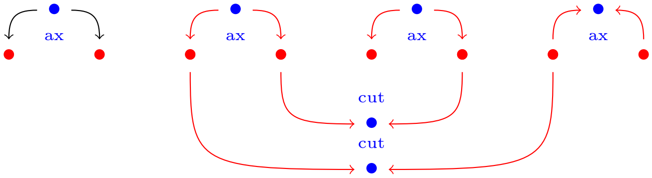


Example





Example



A remark on Proof Nets

- One can notice that the tests for A coming from the correctness criterion are in correspondence with the *proofs* of $A^\perp$.

$$\begin{array}{rcl} \text{Tests for A} & = & \text{Proofs of } A^\perp \\ \text{Proofs of A} & = & \text{Tests for } A^\perp \end{array}$$

Geometry of Interaction: Negation

Geometry of interaction aims at more than a mere interpretation of proofs. It is a complete reconstruction of logic around the dynamics of cut-elimination.

Based on the duality between tests for a A and proofs of $A^{\perp}$:

- one considers a set of more general objects (paraproofs) representing both proofs and tests;
- one defines a notion of orthogonality that translates the correctness criterion.

Geometry of Interaction: Formulas and Connectives

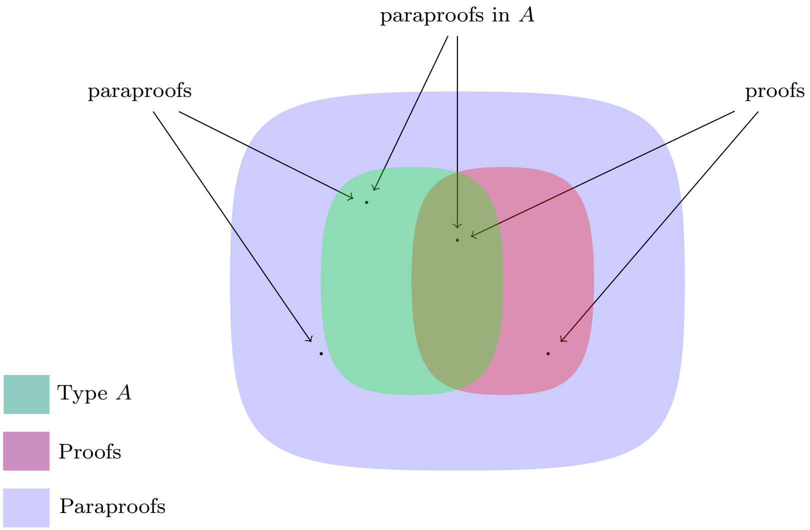
One defines formulas (or types) as sets of paraproofs closed by bi-orthogonality or, equivalently:

Definition

A formula is a set of paraproofs A such that $A = B^{\perp}$ for B a given set of tests.

Connectives are defined on paraproofs, and this definition induces a construction on types.

Principle


paraproofs in A
paraproofs
proofs
Type A
Proofs
Paraproofs

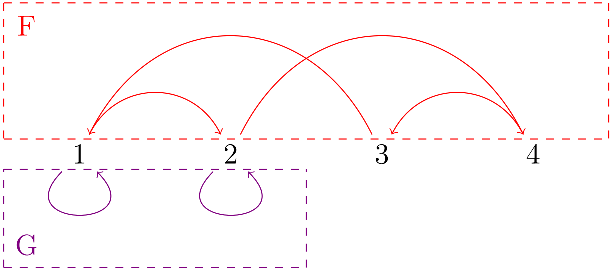
Interaction Graphs

Basics

- Paraproofs are graphs (with a natural number);
- Cut-elimination corresponds to taking the graph of paths;
- Orthogonality is defined by counting the cycles between two graphs.

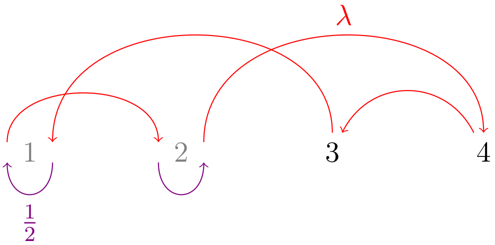
Execution

The execution $F :: G$ of two graphs F, G is the graph of alternating paths of source and target in $V^F \Delta V^G$.



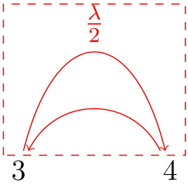
Execution

The execution $F :: G$ of two graphs F, G is the graph of alternating paths of source and target in $V^F \Delta V^G$.



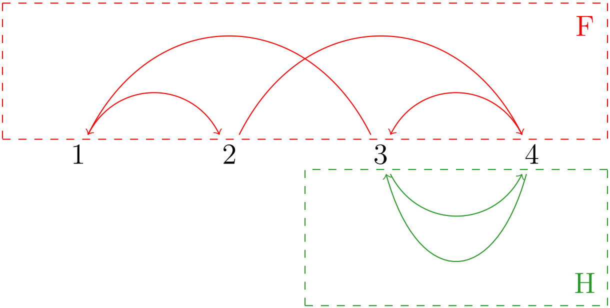
Execution

The execution $F :: G$ of two graphs F, G is the graph of alternating paths of source and target in $V^F \Delta V^G$.



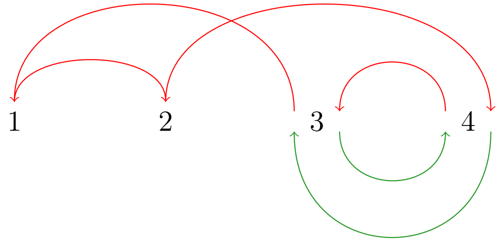
Cycles

In some cases, cycles appear during this operation.

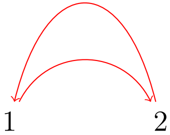


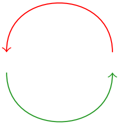
Cycles

In some cases, cycles appear during this operation.

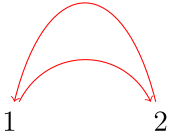


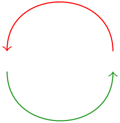
In some cases, cycles appear during this operation.





In some cases, cycles appear during this operation.





We will write $n_{F;G}$ the number of cycles appearing during the execution of the graphs F and G.

Counting Cycles

To take into account the appearance of cycles, we will count them in order to keep track of them.

Definition

A *paraproof* $\mathfrak{f}$ will be a graph F together with a natural number n_F.

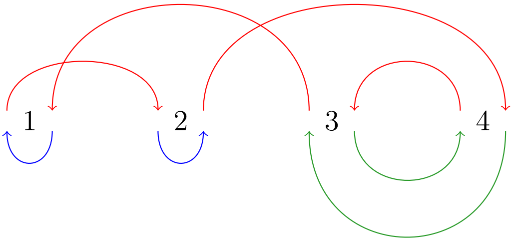
Definition

$$\ll\mathfrak{f},\mathfrak{g}\gg_m = n_F + n_G + n_{F;G}$$

Definition

The execution $\mathfrak{f} :: \mathfrak{g}$ of two paraproofs $\mathfrak{f} = (F, n_F)$ and $\mathfrak{g} = (G, n_G)$ is equal to $(F :: G, \ll\mathfrak{f},\mathfrak{g}\gg_m)$.
When F and G have no common vertices, $F :: G$ is the union of F and G, and $n_{F;G} = 0$. In this case, we will write $\mathfrak{f} \otimes \mathfrak{g}$ instead of $\mathfrak{f} :: \mathfrak{g}$.

The adjunction



Proposition

$$\langle\!\langle \mathfrak{f}, \mathfrak{g} \otimes \mathfrak{h} \rangle\!\rangle_m = \langle\!\langle \mathfrak{f} :: \mathfrak{g}, \mathfrak{h} \rangle\!\rangle_m$$

Definition

Two paraproofs $\mathfrak{f} = (F, n_F)$ and $\mathfrak{g} = (G, n_G)$ are orthogonal when they have the same sets of vertices and $\ll\mathfrak{f}, \mathfrak{g}\gg_m = 1$.

- This is the correctness criterion of proof nets.

Definition

A *type* is a set of paraproofs equal to its bi-orthogonal.

Definition

If $\mathbf{A}, \mathbf{B}$ are types, one can define:

$$\mathbf{A} \otimes \mathbf{B} = \{\mathfrak{a} \otimes \mathfrak{b} \mid \ \mathfrak{a} \in \mathbf{A}, \mathfrak{b} \in \mathbf{B}\}^{\perp\perp}$$

$$\mathbf{A} \multimap \mathbf{B} = \{\mathfrak{f} \mid \ \forall \mathfrak{a} \in \mathbf{A}, \mathfrak{f} :: \mathfrak{a} \in \mathbf{B}\}$$

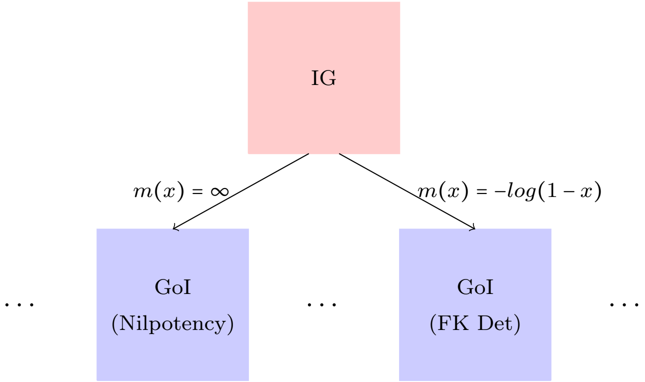
Proposition

$$\mathbf{A} \multimap \mathbf{B} = (\mathbf{A} \otimes \mathbf{B}^{\perp})^{\perp}$$

- We get a model of multiplicative linear logic.

Relation to Girard's constructions

Instead of counting cycles, we measure them (edges are weighted). We obtain a family of constructions parametrized by a "measure" m.



ELL

- In a generalization of this setting one can define several exponential connectives !.
- In particular, we can define a ! connective yielding a restricted system: ELL;

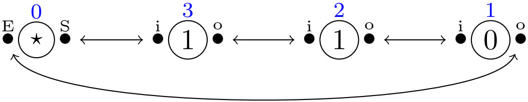
Logarithmic Space

joint work with Clément Aubert (Paris 13)

Representation of Integers

- Principle: an integer n is represented as a binary list, i.e. as a proof of
$$\underbrace{!(X \multimap X)}_{0} \multimap \underbrace{!(X \multimap X)}_{1} \multimap \underbrace{!(X \multimap X)}_{\star}$$
- The list can be read from the contraction rules.
- The GoI interpretation of these proofs are matrices (M_n) representing the sets of axiom links: we obtain a 6×6 matrix whose coefficients are $k \times k$ matrices ($k = \log_2(n)$ is the length of the list).

Representation of Integers: Example



3	$(0o,3)$	$(0i,3)$	$(1o,3)$	$(1i,3)$	$(S,3)$	$(E,3)$
2	$(0o,2)$	$(0i,2)$	$(1o,2)$	$(1i,2)$	$(S,2)$	$(E,2)$
1	$(0o,1)$	$(0i,1)$	$(1o,1)$	$(1i,1)$	$(S,1)$	$(E,1)$
0	$(0o,0)$	$(0i,0)$	$(1o,0)$	$(1i,0)$	$(S,0)$	$(E,0)$

$$\underbrace{!(X \multimap X)}_{0} \multimap \underbrace{!(X \multimap X)}_{1} \multimap \underbrace{!(X \multimap X)}_{\star}$$

- The blue indices in the previous slide are "states" of the integer (the integer is a function as in λ-calculus);
- As a consequence, a graph interacts with an integer only through the interface $\{0o, 0i, 1o, 1i, S, E\}$;
- We say a graph G (which can have "states" in the same way integers do) accepts an integer n when there are no cycles between G and M_n;
- The language $[G]$ recognized by a graph G is the set of integers it accepts.

Results

- In the formal setting:

$$\underbrace{\mathcal{M}_6(\mathbb{C}) \otimes \mathcal{R}}_{\text{integers}} \otimes \mathcal{M}_n(\mathbb{C})$$

  we define "machines" as a set R of operators in $\mathcal{M}_6 \otimes \mathcal{M}_n$ and we can show that $[R] = \{[r], r \in R\}$ is the set of regular languages.
- In the more complex setting:

$$\mathcal{M}_6(\mathbb{C}) \otimes \left((\bigotimes_{n \in \mathbb{N}} \mathcal{R}) \rtimes \mathfrak{G} \right) \otimes \mathcal{M}_n(\mathbb{C})$$

  we define two sets P_+ and $P_{+,1}$ of operators in $\mathcal{M}_6 \otimes \mathcal{M}_n$ and show:

$$[P_+] = \textbf{co-NL} \qquad [P_{+,1}] = \textbf{L}$$

Conclusion

Perspectives

- Use tools and/or invariants of operator algebras to obtain results in algorithmic complexity;
- Use the concepts of GoI to study other notions of computation:
  - Quantum computation;
  - Pi-calculus;
  - ...
- Understand the links with homotopy.