

Graphs of Interaction

Thomas Seiller (IML - Marseille)

March 21st 2010

What are graphs of interaction

- 1 A localized semantic (for MLL).
- 2 Yields both a denotational semantic and a notion of truth.

What are graphs of interaction

- 1 A localized semantic (for MLL).
- 2 Yields both a denotational semantic and a notion of truth.
- 3 It is inspired from the latest version of geometry of interaction GoI5 (Girard), and turns out to be a combinatorial version of it.

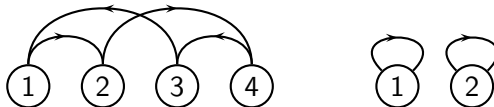
- 1 Graphs and loops
 - Plugging and Execution
 - Adjunction

- 2 Graphs of Interaction
 - Probabilistic Gol
 - Geometry of Interaction 5

- 3 Generalizations
 - Quantum Gol
 - Additives

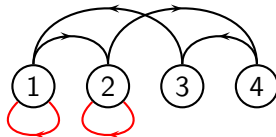
Plugging

Plugging corresponds to the recognition of a cut.



Plugging

Plugging corresponds to the recognition of a cut.



Plugging

Plugging corresponds to the recognition of a cut.

Definition

If $G = (V, E)$ and $H = (V', F)$ are two graphs, we define $G \square H$ as the graph $G \uplus H = (V \cup V', E \uplus F)$ together with a coloring function δ such that

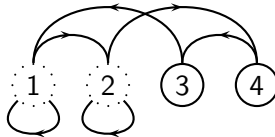
$$\delta(e) = \begin{cases} 0 & \text{si } e \in E \\ 1 & \text{si } e \in F \end{cases}$$

Execution (Permutations)

We wish to define the execution in the same way it was defined on the first version of Gol with permutations (*Multiplicatives*).

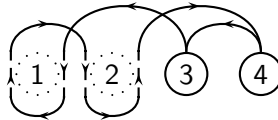
Execution (Permutations)

We wish to define the execution in the same way it was defined on the first version of Gol with permutations (*Multiplicatives*).



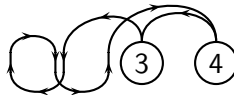
Execution (Permutations)

We wish to define the execution in the same way it was defined on the first version of Gol with permutations (*Multiplicatives*).



Execution (Permutations)

We wish to define the execution in the same way it was defined on the first version of Gol with permutations (*Multiplicatives*).

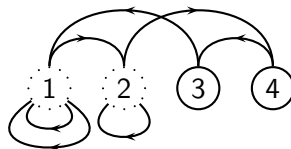


Execution (Graphs)

However, as we are working with graphs, there can be more than one edge from a vertex v to a vertex w .

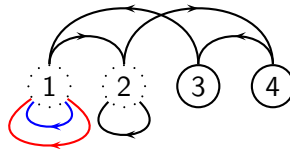
Execution (Graphs)

However, as we are working with graphs, there can be more than one edge from a vertex v to a vertex w .



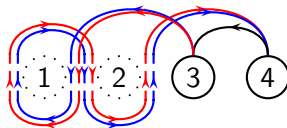
Execution (Graphs)

However, as we are working with graphs, there can be more than one edge from a vertex v to a vertex w .



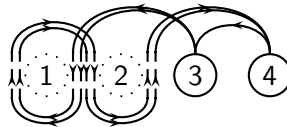
Execution (Graphs)

However, as we are working with graphs, there can be more than one edge from a vertex v to a vertex w .



Execution (Graphs)

However, as we are working with graphs, there can be more than one edge from a vertex v to a vertex w .



Execution (Graphs)

However, as we are working with graphs, there can be more than one edge from a vertex v to a vertex w .

Definition

We define the execution $F :: G$ between two graphs $F = (V, E)$ et $G = (V', F)$ as

$$F :: H = (V \Delta V', Path^{alt}(F \square G))$$

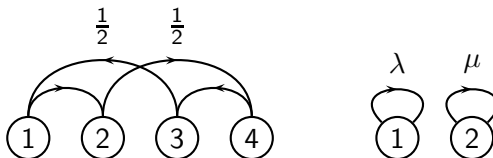
where $Path^{alt}(F \square G)$ is the set of alternating paths in $F \square G$.

Weighted graphs

Moreover, our graphs are weighted, so we define the weight of a path as the product of the edges it is composed of.

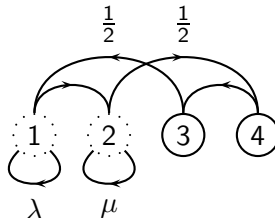
Weighted graphs

Moreover, our graphs are weighted, so we define the weight of a path as the product of the edges it is composed of.



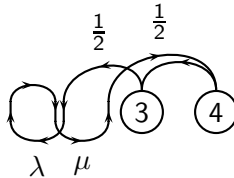
Weighted graphs

Moreover, our graphs are weighted, so we define the weight of a path as the product of the edges it is composed of.



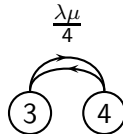
Weighted graphs

Moreover, our graphs are weighted, so we define the weight of a path as the product of the edges it is composed of.



Weighted graphs

Moreover, our graphs are weighted, so we define the weight of a path as the product of the edges it is composed of.

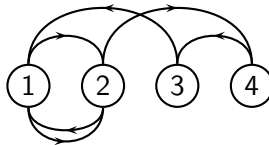


Counting loops

Sometimes, cycles can appear between two graphs.

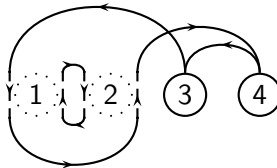
Counting loops

Sometimes, cycles can appear between two graphs.



Counting loops

Sometimes, cycles can appear between two graphs.



Counting loops

Let $G_1 = (V_1, F_1)$ and $G_2 = (V_2, F_2)$ be two graphs, with $V \cap V' = \emptyset$. We define $G_1 \cup G_2$ as the graph $(V_1 \cup V_2, F_1 \uplus F_2)$.

Counting loops

Let $G_1 = (V_1, F_1)$ and $G_2 = (V_2, F_2)$ be two graphs, with $V \cap V' = \emptyset$. We define $G_1 \cup G_2$ as the graph $(V_1 \cup V_2, F_1 \uplus F_2)$.

Proposition

Let $F = (V_1 \cup V_2, E)$ be a graph. Denoting $\text{cycl}(F, G)$ the number of alternating loops between F and G , we get :

$$\text{cycl}(F, G_1 \cup G_2) = \text{cycl}(F, G_1) + \text{cycl}(F :: G_1, G_2)$$

Counting loops

We will associate to a given couple of directed weighted graphs a number which is linked to the number of (alternating) loops. In fact, we will just sum the weights of the loops, normalized by their length.

$$\sum_{\pi \in \text{Path}^{x,x}(F,G)} \frac{\omega_{F \square G}(\pi)}{\text{lg}(\pi)}$$

This can be rewritten, by quotienting the set of loops by their starting point, as

$$\sum_{\pi \in \text{AltLoops}(F,G)} \frac{\omega_{F \square G}(\pi)}{\text{pow}(\pi)}$$

where $\text{pow}(\pi)$ is the greatest integer k such that there exists a loop ρ satisfying $\pi = \rho^k$.

Loops

Definition

We define a cycle as a loop π such that $\text{pow}(\pi) = 1$. We write $\text{Cycl}(F, G)$ the set of alternating cycles in $F \square G$.

Definition

$$\ll F, G \gg = \sum_{\pi \in \text{Cycl}(F, G)} \sum_{k \geq 1} \frac{\omega(\pi)^k}{k} = \sum_{\pi \in \text{Cycl}(F, G)} -\log(1 - \omega(\pi))$$

Adjunction

Proposition (Adjunction)

$$\ll F, G_1 \cup G_2 \gg = \ll F, G_1 \gg + \ll F :: G_1, G_2 \gg$$

Remark

If weights are in $\{0, 1\}$, the third term $\ll F, G_1 \gg$ of the adjunction is either equal to 0 or to ∞ .

Graph reduction

Definition

We define the operation $\hat{}$ that associates, to each weighted graph G , a simple weighted graph by replacing the set $E_{v,w}$ of edges from v to w by a single edge which weight is equal to $\sum_{x \in E_{v,w}} \omega(x)$.

Graph reduction

Definition

We define the operation $\hat{}$ that associates, to each weighted graph G , a simple weighted graph by replacing the set $E_{v,w}$ of edges from v to w by a single edge which weight is equal to $\sum_{x \in E_{v,w}} \omega(x)$.

Theorem

$$\ll F, G \gg = \ll F, \hat{G} \gg$$

Projects

Definition (Projects — Proofs)

A *project* is a couple $\alpha = (a, A)$ where $a \in \mathbb{R}_+ \cup \{\infty\}$ is the *wager* and A is a directed weighted graph (with weights in $[0, 1]$). The finite set V_A of vertices of A will be called the *carrier* of α .

Projects

Definition (Projects — Proofs)

A *project* is a couple $\alpha = (a, A)$ where $a \in \mathbb{R}_+ \cup \{\infty\}$ is the *wager* and A is a directed weighted graph (with weights in $[0, 1]$). The finite set V_A of vertices of A will be called the *carrier* of α .

Definition

Let α and $\flat$ be two projects. We define

$$\ll \alpha, \flat \gg = a + b + \ll A, B \gg$$

Conducts

Definition (Duality (Orthogonality))

Two projects $\mathfrak{a}$ and $\mathfrak{b}$ of same carrier are said to be *polar* when $\llbracket \mathfrak{a}, \mathfrak{b} \rrbracket \neq 0, \infty$.

Conducts

Definition (Duality (Orthogonality))

Two projects $\mathfrak{a}$ and $\mathfrak{b}$ of same carrier are said to be *polar* when $\llbracket \mathfrak{a}, \mathfrak{b} \rrbracket \neq 0, \infty$.

Definition (Conducts — Formulas)

A conduct is a set of projects (with same carrier) equal to its bi-polar.

Connectives on projects

Definition

Let a, b be two projects of disjoint carriers, we define

$$a \otimes b = (\llbracket a, b \rrbracket, A \cup B)$$

Connectives on projects

Definition

Let $\mathfrak{a}, \mathfrak{b}$ be two projects of disjoint carriers, we define

$$\mathfrak{a} \otimes \mathfrak{b} = (\llbracket \mathfrak{a}, \mathfrak{b} \rrbracket, A \cup B)$$

Definition (Cut)

Let $\mathfrak{f}$ and $\mathfrak{g}$ be two projects of respective carriers $V \cup V'$ and $V' \cup V''$. We define the cut between $\mathfrak{f}$ and $\mathfrak{g}$ as

$$\mathfrak{f} :: \mathfrak{g} = (\llbracket \mathfrak{f}, \mathfrak{g} \rrbracket, F :: G)$$

Connectives on Conducts

Definition

Let $\mathbf{A}, \mathbf{B}$ be two conducts of disjoint carriers, we define

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

Connectives on Conducts

Definition

Let $\mathbf{A}, \mathbf{B}$ be two conducts of disjoint carriers, we define

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

Definition (Linear Implication)

Let $\mathbf{A}, \mathbf{B}$ be two conducts of disjoint carrier, we define

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

Duality of the connectives

Proposition

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

Proof.

$$\begin{aligned} \llbracket f, \mathbf{a} \otimes \mathbf{b} \rrbracket &= f + (a + b) + \llbracket F, A \cup B \rrbracket \\ &= f + a + b + \llbracket F, A \rrbracket + \llbracket F :: A, B \rrbracket \\ &= (a + f + \llbracket F, A \rrbracket) + b + \llbracket F :: A, B \rrbracket \\ &= \llbracket f :: \mathbf{a}, \mathbf{b} \rrbracket \end{aligned}$$

Denotational semantics

Definition

We define the category $\mathcal{G}raph_{MLL}$

$$\begin{aligned}\mathcal{O}bj &= \{\mathbf{A} \mid \mathbf{A} = \mathbf{A}^{\downarrow\downarrow} \text{ of carrier } X_{\mathbf{A}} \in \mathcal{P}_f(\mathbb{N})\} \\ \mathcal{M}or[A, B] &= \{f \in \psi_0(\mathbf{A}) \multimap \psi_1(\mathbf{B})\}\end{aligned}$$

Proposition

We define functors from the connectives $\otimes$, $\multimap$ (with delocalizations) and the negation. With these definitions, the category $\mathcal{G}raph_{MLL}$ is a $*$ -autonomous category.

Truth

Definition (Successful project)

A project $\alpha = (0, A)$ is successful when $\hat{A}$ is symmetric and satisfies $\hat{A}^3 = \hat{A}$, $Tr(\hat{A}) = 0$.

Truth

Definition (Successful project)

A project $\alpha = (0, A)$ is successful when $\hat{A}$ is symmetric and satisfies $\hat{A}^3 = \hat{A}$, $Tr(\hat{A}) = 0$.

Definition (Truth)

A conduct $\mathbf{A}$ is true when it contains a successful project.

Truth

Proposition

The conducts $\mathbf{A}$ and $\mathbf{A}^\sim$ can't both be true.

Truth

Proposition

The conducts $\mathbf{A}$ and $\mathbf{A}^\sim$ can't both be true.

Proposition

If $f \in \mathbf{A} \multimap \mathbf{B}$ and $g \in \mathbf{B} \multimap \mathbf{C}$ are successful, then $f :: g \in \mathbf{A} \multimap \mathbf{C}$ is successful.

Truth

Proposition

The conducts $\mathbf{A}$ and $\mathbf{A}^\perp$ can't both be true.

Proposition

If $f \in \mathbf{A} \multimap \mathbf{B}$ and $g \in \mathbf{B} \multimap \mathbf{C}$ are successful, then $f :: g \in \mathbf{A} \multimap \mathbf{C}$ is successful.

Proposition (Weak internal completeness)

If $f \in \mathbf{A} \otimes \mathbf{B}$ is successful, there exists successful projects $a \in \mathbf{A}$ and $b \in \mathbf{B}$ such that $f = a \otimes b$.

From graphs to operators

- We restrict ourselves to simple graphs G such that the adjacency matrix of G has norm ≤ 1 and is hermitian (i.e. if $(v, w) \in E$, then $(w, v) \in E$ and $\omega(w) = \overline{\omega(v)}$).

From graphs to operators

- We restrict ourselves to simple graphs G such that the adjacency matrix of G has norm ≤ 1 and is hermitian (i.e. if $(v, w) \in E$, then $(w, v) \in E$ and $\omega(w) = \overline{\omega(v)}$).
- We define an embedding Ψ associating to a graph an operator in the hyperfinite factor, and extend it to projects.

From graphs to operators

- We restrict ourselves to simple graphs G such that the adjacency matrix of G has norm ≤ 1 and is hermitian (i.e. if $(v, w) \in E$, then $(w, v) \in E$ and $\omega(w) = \overline{\omega(v)}$).
- We define an embedding Ψ associating to a graph an operator in the hyperfinite factor, and extend it to projects.
- We have :

Graphs		Gol5
$a \downarrow b$	$\Leftrightarrow$	$\Psi(a) \downarrow \Psi(b)$
$\Psi(a :: b)$	$=$	$\Psi(a) \hat{::} \Psi(b)$
$\ll a, b \gg$	$=$	$\ll (a), (b) \gg$
a is successful	$\Rightarrow$	$\Psi(a)$ is successful

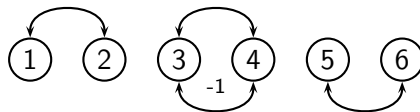
Complex weights

- We extend our weights, and consider graphs weighted by complex numbers of module ≤ 1 .

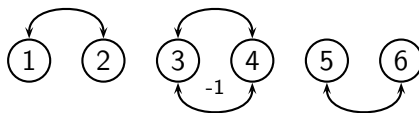
Complex weights

- We extend our weights, and consider graphs weighted by complex numbers of module ≤ 1 .
- Everything works nicely except the notion of truth which does not compose.

Example



Example

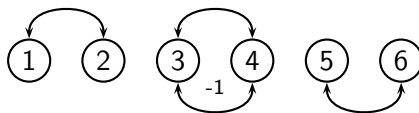


This is because we get "successful" projects that operate on other bases. For instance, the graph whose adjacency matrix is

$$\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$$

is written as a simple transposition in a well-chosen basis.

Example



This is because we get "successful" projects that operate on other bases. For instance, the graph whose adjacency matrix is

$$\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}$$

is written as a simple transposition in a well-chosen basis.

The solution : a subjective notion of truth, i.e. a notion of truth that depends on a basis.

Additives

Two solutions come to mind for an extension to additives :

Additives

Two solutions come to mind for an extension to additives :

- 1 Expand the range of weights to manage slices. We could for instance introduce boolean variables as in additive proof nets.

Additives

Two solutions come to mind for an extension to additives :

- ① Expand the range of weights to manage slices. We could for instance introduce boolean variables as in additive proof nets.
- ② We can consider *stratified* graphs, i.e. graphs with multiple sets of edges. All the definitions we had for the multiplicatives work, we just have to define the operations slice by slice. The advantage : two notions of union arise naturally, one being the $\otimes$ and the other being the $\&$.