Multiplicative Linear Logic from Logic Programs and Tilings

Boris Eng, Thomas Seiller

Multiplicative Linear Logic from Logic Programs and Tilings

Boris Eng
LIPN – UMR 7030
Université Sorbonne Paris Nord, France
https://www.engboris.fr
engboris@hotmail.fr

Thomas Seiller
CNRS, LIPN – UMR 7030
Université Sorbonne Paris Nord, France
https://www.seiller.org
seiller@lipn.fr

***Abstract*—We present a non-deterministic model of computation related to Robinson's first-order resolution. This model formalises and extends ideas sketched by Girard in his Transcendental Syntax programme. After establishing formal definitions and basic properties, we show its Turing-completeness by exhibiting how it naturally models logic programs as well as non-deterministic tiling constructions such as those defining the abstract tile assembly model, recently used in DNA computing. In a second part, we explain how this model of computation yields, using realisability techniques, a dynamic semantics of proofs in the multiplicative fragment of linear logic (MLL), for which we obtain full-completeness results for both MLL and MLL extended with the so-called MIX rule.**

I. Introduction

The study of logic traditionally begins from a presupposed definition of logic. Typical examples are Gentzen's natural deduction and sequent calculus [1], [2] which are attempts at representing mathematical reasoning by means of logical rules one applies successively in order to construct proofs. Several discoveries emerged from this starting point. The Curry-Howard correspondence between proofs and programs [3], [4] somehow freed logic from its isolation, as it then appears as a part of a bigger picture. For instance, the simply typed λ-calculus (isomorphic to natural deduction restricted to implication) exists within a larger realm: the untyped λ-calculus, a Turing-complete model of computation at the core of functional programming.

Inspired by the semantics of the λ-calculus [5], linear logic was introduced by Girard [6] as a refinement of intuitionistic logic (the logic behind natural deduction). Apart from defining a sequent calculus, Girard was led to introduce an alternative syntax for linear logic: proof-nets, which exhibit a non-sequential structure [7]. Intuitively, proof-nets can be understood as part of a more general model of computation, that of *proof-structures* which do not necessarily hold a logical meaning. It is then possible to characterise those being *logically correct*, namely proof-nets, as the proof-structures satisfying a specific *correctness criterion*. Although many such criteria were defined [8], [9], [10], [11], we will consider here the standard Danos-Regnier criterion [8].

Following this new understanding of proofs, Girard introduced his geometry of interaction (GoI) programme [12] aiming at defining a semantics of proofs accounting for their dynamics, inspired by the dynamics of the execution of programs. This *dynamic semantics* approach, a major inspiration behind game semantics [13], [14], distinguishes itself from denotational semantics which consider proofs as static objects. Nowadays, the term "geometry of interaction" usually refers to a static execution of λ-terms by a token machine [15], [16], inspired by a simplification of Girard's first geometry of interaction [17]. It may also refer to GoI-based categorical semantics [18], [19]. In this paper, however, we refer to Girard's original programme [20], [21], [22], [23], [24], [25].

Girard's initial programme went beyond this idea of dynamic semantics. It can be understood as a sort of *reverse engineering* reconstructing logic – in particular linear logic – from very general computational objects by using realisability techniques. In some way, the idea is the same as the reconstruction of simple types from the untyped λ-calculus. We refer the reader to Riba's work [26] for more details.

In this aspect of reconstructing linear logic, several geometry of interaction models were defined using operator algebras [20], [21], [23], [24], unification algebras [22], graphs [27], [28] and graphings [29], [30], [31]. Although all these models did define rich models, that were in particular used to study computational complexity [32], [33], [34], [35], [36], they had two main drawbacks. First, the objects used to interpret even the most basic proofs were most of the time infinite objects whereas we expect reasoning to be finite. Secondly, the obtained models did interpret soundly the fragments of linear logic considered, but no completeness results exist[1].

Recently, Girard published a series of articles [37], [38], [39], [40], [41] sketching the main lines of a new kind of model that would have the qualities of geometry of interaction models, but improve on them at least concerning the first failure mentioned above. Those articles claim great improvements, proposing in particular a Curry-Howard interpretation of first-order logic [40]. However, they are too inexact in form to serve satisfactorily as the basis of a mathematical theory[2]. The current work is the first step towards a proper formal account of the model, with underlying motivation the

[1] While this aspect is a failure somehow, it is also a feature as the models are very rich and open other paths of reflexion.

[2] The formulation is borrowed from Church's criticism of von Mises' notion of kollektiv [42].

representation of first-order logic and its possible applications in relation with descriptive complexity results [43], such as the Immerman-Vardi theorem [44], [45].

A. Contributions and Plan of the paper

The main contributions of this paper are the following:

- In the first section, we formally describe a model of computation named stellar resolution, based on Robinson's first-order resolution [46], which extends the model of computation vaguely described by Girard. We define two alternative executions. The first one (Definition 22) closer to usual logic programming and the second one (Definition 26) based on the combinatorics of geometric tiling. In Section III-C, we prove the main properties of the model and state the Turing-completeness (Proposition 28) for the two executions. In particular, while Girard claimed the failure of the Church-Rosser property, we are able to prove it holds for stellar resolution (Theorem 30).
- In Section III-B, we relate our formalism to well-known models of computation which however are rather unexpected in a proof-theoretic context: Wang tilings [47] and the abstract tile assembly model [48], [49], which has applications in DNA computing [50], [51].
- We explain how our model captures the dynamics of the cut-elimination procedure for the multiplicative fragment of linear logic (MLL) (Section IV, Theorem 41), and the correctness criterion for proof-structures (Section V, Theorem 48). This implicitly leads to a model which can express both MLL proofs (hence the linear simply typed λ-calculus) and logic programs (through disjunctive first-order clauses) with objects of the same kind. We also remark that it is possible to consider an alternative model of MLL which can cohabit with tilings models (Remark 52). This shows that proof-structures and tiling-based models share a similar dynamics.
- Finally, in Section V-C, we explain how realisability techniques similar to those used in λ-calculus can be used to define types that organise into a denotational semantics for MLL (a $*$-autonomous category), and prove soundness and completeness of the model w.r.t. both MLL (Section V-E, Theorem 74) and MLL+MIX (Section V-D, Theorem 70 & Theorem 72), an extension of MLL with the so-called MIX rule [52]. The construction of types suggests the possibility of speaking about types and describing computational behaviours in the context of atypical models in type theory such as logic programs, tiling-based computation, or structure-based computation (e.g boolean circuits and automata).

II. Stellar Resolution

The stellar resolution model is basically a graph-theoretic reformulation of Robinson's first-order clausal resolution [46] used as a model of computation for different purposes. Instead of reaching an empty clause (a contradiction) or considering goals/queries, we are interested in the set of all maximal graphs of connexions between the atoms (which is reminiscent of the resolution operator and its refinements [53]). This is also very similar resolution-based graph models such as Kowalski's connection graphs [54], Sickel's clause interconnectivity graphs [55] or more generally logic programming [56] except that we only keep the computational content of resolution without any reference to logic. We will then show how multiplicative linear logic proofs can naturally emerge from this model.

As first-order resolution is based on the theory of unification [57], [46], we here recall basic definitions and refer the reader to the article of Lassez et al. [58] for more details which are often omitted in the literature or Baader et al. [59] for a broader view.

A. First-order term unification

A *signature* $\mathbb{S} = (\mathtt{Vars}, \mathtt{Func}, \mathtt{arity})$ consists of an infinite countable set $\mathtt{Vars}$ of variables, a countable set $\mathtt{Func}$ of function symbols whose arities are given by $\mathtt{arity} : \mathtt{Func} \to \mathbf{N}$.

We set a signature for this section. The set of *terms* $\mathtt{Terms}(\mathbb{S})$ is inductively defined by the following grammar:

$$t, u ::= x \mid f(t_1, \dots, t_n) \qquad \text{(Terms)}$$

with $x \in \mathtt{Vars}$, $f \in \mathtt{Func}, \mathtt{arity}(f) = n$.

A *substitution* is a function $\theta : \mathtt{Vars} \to \mathtt{Terms}(\mathbb{S})$. We extend substitutions from variables to terms by $\theta(f(u_1, ..., u_k)) = f(\theta u_1, ..., \theta u_k)$.

A *renaming* is a substitution α such that $\alpha(x) \in \mathtt{Vars}$ for all $x \in \mathtt{Vars}$.

An *equation* is an unordered pair $t \stackrel{?}{=} u$ of terms in $\mathtt{Terms}(\mathbb{S})$. A *unification problem* is a set of equations.

A *solution* for a unification problem is a substitution θ such that for all equations $t \stackrel{?}{=} u \in P$, $\theta t = \theta u$.

Two terms t and u are *matchable* if there exists a renaming α such that $\{\alpha t \stackrel{?}{=} u\}$ has a solution called *matching*. A matching between two terms is *exact* when it is a renaming.

The problem of deciding if a solution to a given unification problem P exists is decidable [60]. Moreover, there exists a minimal solution $\mathtt{Solution}(P)$ w.r.t. the preorder $\theta \preceq \psi \Leftrightarrow \exists \theta'. \psi = \theta' \circ \theta$, unique up to renaming.

Let us note that several algorithms were designed to compute the unique solution when it exists, such that the Martelli-Montanari unification algorithm [60].

We define a notation for the unary encoding of natural numbers which will be useful through the article.

Notation 1 (Natural number). The *encoding of a natural number* $n \in \mathbf{N}$ is defined by $\underline{n} := s^n(0)$ where s^n represents n applications of the unary symbol s.

B. Stars and Constellations

We work with collections of first-order terms called *stars* meant to be connected to each other along their components called *rays*. These terms can be prefixed by function symbol called a *colour* allowing for the consideration of various types of composition, together with a $+$ or $-$ sign indicating their ability to be composed with a term of opposite polarity.

If we bear first-order resolution in mind, stars are exactly disjunctive clauses (or disjunctive logic programs [61]) where colours are predicates. However, unlike usual resolution, we allow unpolarised atoms which cannot be connected. We define the core objects of our model and later define their dynamics/evaluation.

Definition 2 (Coloured Signature)**.** A *coloured signature* is a 4-tuple $\mathbb{C} = (\texttt{Vars}, \texttt{Colours} \subseteq \texttt{Func}, \texttt{Func}, \texttt{arity})$ such that $(\texttt{Vars}, \texttt{Func}, \texttt{arity})$ is a signature. Any $c \in \texttt{Colours}$ is called a *colour*.

We will now work with the coloured signature $\mathbb{C} = (\texttt{Vars}, \texttt{Colours}, \texttt{Func}, \texttt{arity})$ unless specified otherwise.

Definition 3 (Ray)**.** A ray is a term in the grammar

$$r ::= +c(t_1, ..., t_n) \mid -c(t_1, ..., t_n) \mid t \qquad \text{(Rays)}$$

where $t_1, ..., t_n \in \texttt{Terms}(\mathbb{C})$ and $c \in \texttt{Colours}$ with $\texttt{arity}(c) = n$. The underlying term of a ray is defined by $\lfloor +c(t_1, ..., t_n) \rfloor = \lfloor -c(t_1, ..., t_n) \rfloor = c(t_1, ..., t_n)$ and $\lfloor t \rfloor = t$. The define $\texttt{Rays}(\mathbb{C})$ as the set of all rays over a coloured signature $\mathbb{C}$.

Although we allow colours to appear inside the underlying term of a ray, we do not use this feature in this paper and will only consider that colours are prefixes, similarly to predicates in first-order logic. This additional feature will be considered in future works.

Definition 4 (Star)**.** A star ϕ over a coloured signature $\mathbb{C}$ is a finite and non-empty multiset of rays, i.e. a finite set $|\phi|$ together with a map $\texttt{gen}_\phi : |\phi| \to \texttt{Rays}(\mathbb{C})$. The set of variables appearing in ϕ is written $\texttt{vars}(\phi)$. Stars are written as multisets $\phi = [r_1, ..., r_n]$.

Definition 5 (Substitutions and α-equivalence)**.** Given a substitution θ, its action extends to rays by letting $\theta(\pm c(t_1, ..., t_n)) = \pm\theta(c(t_1, ..., t_n))$ with $\pm \in \{+, -\}$. It also extends to stars: $\theta[r_1, ..., r_n] = [\theta r_1, ..., \theta r_n]$.

We say that two stars ϕ_1, ϕ_2 are α*-equivalent*, written $\phi_1 \approx_\alpha \phi_2$, when there exists a renaming α such that $\alpha\phi_1 = \phi_2$.

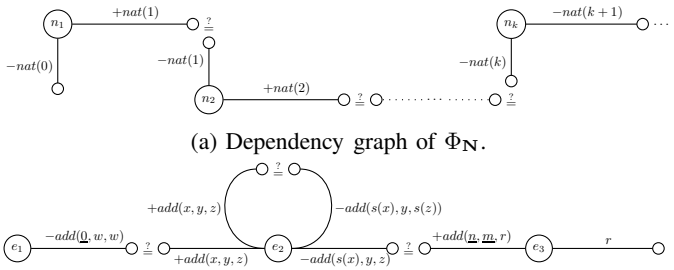
Notation 6. In this work, stars will be considered up to α-equivalence. We therefore define $\texttt{Stars}(\mathbb{S})$ as the set of all stars over a coloured signature $\mathbb{C}$, quotiented by $\approx_\alpha$.

Definition 7 (Constellation)**.** A constellation Φ is a (countable) multiset of stars, i.e. a countable (possibly infinite) set $|\Phi|$ together with a map $\texttt{gen}_\Phi : |\Phi| \to \texttt{Stars}(\mathbb{S})$. The variables appearing in Φ are considered bound to their star i.e $\bigcap_{e \in |\Phi|} \texttt{vars}(\texttt{gen}_\Phi(e)) = \emptyset$. The set of rays of Φ is defined as $\texttt{Rays}(\Phi) = \{(s, r) \mid s \in |\Phi|, r \in |\texttt{gen}_\Phi(s)|\}$. A finite constellation will sometimes be written as a sum $\Phi = \phi_1 + ... + \phi_n$.

Example 8. Here are examples of a finite and an infinite constellation:

- $\Phi_{\mathbf{N}}^{n+m} = [-add(\underline{n}, \underline{m}, r), r] + [+add(\underline{0}, y, y)] + [+add(s(x), y, s(z)), -add(x, y, z)]$;
- $\Phi_{\mathbf{N}}$ is defined by $|\Phi_{\mathbf{N}}| = \mathbf{N}$ and $\texttt{gen}_{\Phi_{\mathbf{N}}}(n) = [-nat(\underline{n}), +nat(\underline{n+1})]$

where nat is a colour and s and 0 are function symbols. The constellation $\Phi_{\mathbf{N}}^{n+m}$ is a logic program computing $n + m$.



(a) Dependency graph of $\Phi_{\mathbf{N}}$.

(b) Dependency graph of $\Phi_{\mathbf{N}}^{n+m}$.

Fig. 1: Examples of dependency graphs.

Notation 9. Let Φ_1, Φ_2 be two constellations. We write $\Phi_1 \uplus \Phi_2$ their multiset union, i.e. the coproduct $\texttt{gen}_{\Phi_1} \uplus \texttt{gen}_{\Phi_2} : |\Phi_1| \uplus |\Phi_2| \to \texttt{Stars}(\mathbb{S})$.

C. Diagrams: non-deterministic computation graphs

The stars of a constellation are meant to be connected together along their rays of opposite polarities. We define the *dependency graph* of a constellation which is a multigraph specifying which stars can be connected together: it is a finite description of all the allowed connexions. This multigraph corresponds to Kowalski's connection graphs [54] and Sickel's clause interconnectivity graphs [55]. This is also very close to the dependency graphs used in logic programming for the analysis of termination [62].

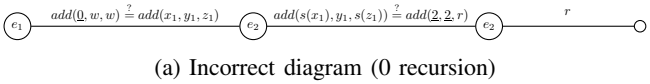
Definition 10 (Duality)**.** Two rays r, r' are dual w.r.t. a set of colours $A \subseteq \texttt{Colours}$, written $r \bowtie_A r'$, when r and r' are of the same colour, have a different polarity and when $\lfloor r \rfloor$ and $\lfloor r' \rfloor$ are matchable.

Definition 11 (Dependency graph)**.** The *dependency graph* $\mathfrak{D}[\Phi; A]$ of a constellation Φ w.r.t. a set of colours $A \subseteq \texttt{Colours}$ is the undirected multigraph $(V^{\mathfrak{D}[\Phi;A]}, E^{\mathfrak{D}[\Phi;A]}, \texttt{extract}^{\mathfrak{D}[\Phi;A]})$ where $V^{\mathfrak{D}[\Phi;A]} = |\Phi|$, $E^{\mathfrak{D}[\Phi;A]} = \{((s, r), (s', r')) \in \texttt{Rays}(\Phi) \mid r \bowtie_A r'\}$, and $\texttt{extract}^{\mathfrak{D}[\Phi;A]}$ is the function defined by $\texttt{extract}^{\mathfrak{D}[\Phi;A]}((s, r), (s', r')) = \{s, s'\}$. We write $\mathfrak{D}[\Phi; -]$ when we connect all the colours of Φ.

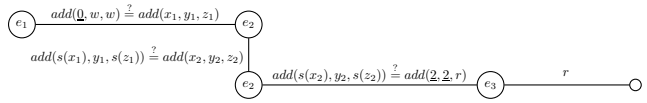
Example 12. We illustrate the dependency graphs of Example 8 in Figure 1.

From a dependency graph, we extract graphs by composing occurrences of stars in a constellation along dual rays. They represent actual connexions between the stars of a constellation, following the connexions allowed by the dependency graph. In presence of cycle in the dependency graph, there may be infinitely many ones.

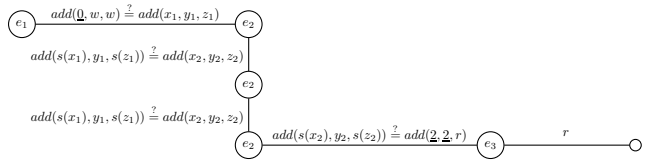
It is similar to extracting execution flow graphs from a program and computing all the possible unfolding of loops. In



(a) Incorrect diagram (0 recursion)

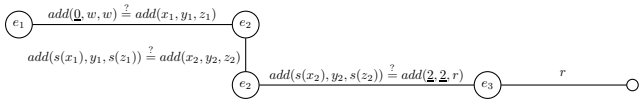


(b) Correct diagram computing $2 + 2$ (1 recursion)

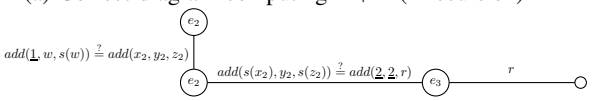


(c) Incorrect diagram (2 recursions)

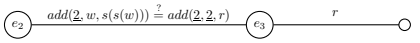
Fig. 2: Examples of diagrams.



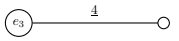
(a) Correct diagram computing $2 + 2$ (1 recursion)



(b) After 1 step of resolution



(c) After 2 steps of resolution



(d) After 3 steps of resolution

Fig. 3: Example of diagram evaluation by the resolution rule.

the context of resolution, Sickel [55] used such graphs called *solutions* for different purposes.

Definition 13 (Diagram)**.** An A-diagram δ on a set of colour $A \subseteq \texttt{Colours}$ over a constellation Φ is a finite connected graph D_δ and a graph homomorphism $\delta : D_\delta \to \mathfrak{D}[\Phi; \mathcal{A}]$ such that rays are uniquely connected, i.e. for all $v \in V^{D_\delta}$ the function $\{e \in E^{D_\delta} \mid v \in \texttt{extract}^{D_\delta}(e)\} \to |\delta(v)|; \{(\delta(v), r), (s', r')\} \mapsto r$ is injective. The graph D_δ is considered up to renaming of the vertices and edges. For practical purpose and without loss of generality, we will consider that $V^{D_\delta} = \mathbf{N}$.

Notation 14. Given an A-diagram δ, we define its set of *links* as the set $\texttt{links}(\delta) = \{(v, r) \mid \exists e \in E^{D_\delta}, v \in \texttt{extract}^{D_\delta}(e), (\delta(v), r) \in \texttt{Rays}(\Phi)\}$ and its set of free rays as $\texttt{free}(\delta) = \{(v, r) \mid v \in V^{D_\delta}, (\delta(v), r) \in \texttt{Rays}(\Phi), (v, r) \notin \texttt{links}(\delta)\}$. If $\texttt{free}(\delta) = \emptyset$, we say that δ is *closed*. A diagram is *exact* if all the solutions of its equations are exact matchings (c.f Section II-A).

Example 15. In the Figure 2, we illustrate three diagrams for the constellation $\Phi_{\mathbf{N}}^{2+2}$ coming from the constellation $\Phi_{\mathbf{N}}^{n+m}$ of Example 8. What we did is basically unfolding loops of the dependency graph $\mathfrak{D}[\Phi_{\mathbf{N}}^{2+2}; -]$ (Figure 1). There exists infinitely many diagrams by only one is a complete computation.

A diagram links occurrences of stars along rays. These rays induce equations between terms i.e a unification problem. However, since these occurrences of stars have to be considered distinct in order to connect a star to itself to represent a loop in a computation, we have to rename the variables. Since we consider that $V^{D_\delta} = \mathbf{N}$, we obtain a family of renamings $\alpha_k(x) = x_k$ for $n \in \mathbf{N}$ used to make the occurrences of star different and to make explicit the source of their variables.

Definition 16 (Underlying unification problem)**.** The underlying unification problem of an A-diagram δ over a constellation Φ is defined as

$$\mathcal{P}(\delta) = \{\alpha_v \lfloor r \rfloor \stackrel{?}{=} \alpha_{v'} \lfloor r' \rfloor \mid e \in E^{D_\delta},$$
$$\texttt{extract}^{D_\delta}(e) = \{v, v'\}, \delta(e) = ((\delta(v), r), (\delta(v'), r'))\}.$$

In some ways, diagrams generalise trails in a graph. Where paths describe a possible trajectory for a particle, diagrams describe the possible trajectories of a wave that can simultaneously spread in several directions when encountering forks, similarly to the run on a non-deterministic state machine. We now introduce the notion of *saturated diagrams*, which corresponds to *maximal paths*.

Definition 17 (Saturated diagram)**.** We define a preorder $\sqsubseteq$ on A-diagrams over a constellation Φ by: $\delta \sqsubseteq \delta'$ if there exists an isomorphism φ from a graph D of $D_{\delta'}$ to D_δ such that $\delta = \delta' \circ \varphi$. A maximal A-diagram w.r.t. $\sqsubseteq$ is called *saturated*. In a saturated diagram, we cannot add and connect further occurrences of stars from the original constellation.

Example 18. For the constellation $\Phi_{\mathbf{N}}^{2+2}$ from Example 15, all the diagrams presented in Figure 2 are saturated. There are infinitely many saturated diagrams formed by unfolding the loop. Any subtree of these diagrams constitutes a partial (non-saturated diagram).

III. Execution of constellations

We consider two alternative ways of evaluating diagrams and more generally constellations. The first one reduces tree-shaped diagrams with Robinson's resolution and the second one constructs tilings by connecting stars to each other, possibly forming cycles or grids.

A. Evaluation by resolution

In this section, as in usual resolution, we restrict diagrams to trees. A diagram δ represents a scheme of computation to be done. We reduce it by a linear contraction of the edges of its underlying graph D_δ using Robinson's first-order resolution rule. This rule, we call *fusion*, merges two stars $\phi_1 \cup \{r_1\}, \phi_2 \cup \{r_2\}$ connected along r_1 and r_2 into a new star $\phi_1 \cup \phi_2$ and apply $\texttt{Solution}(\{\lfloor r_1 \rfloor \stackrel{?}{=} \lfloor r_2 \rfloor\})$ on it. Diagrams are reduced until reaching a single star (ensured

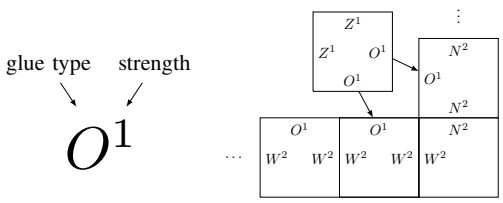


Fig. 4: Illustration of an assembly in an aTAM. Assume we are at temperature $\tau = 2$. We can connect a new tile to an assembly because the glue types match and the sum of the strengths of the connexions is $1 + 1 \geq \tau$.

because diagrams are trees). It is the reduction used in Girard's original paper.

For convenience we use here an equivalent formulation we call *actualisation* which solves $\mathcal{P}(\delta)$ and applies the solution θ on the star induced by the free rays $\texttt{free}(\delta)$. The fact that this approach leads to the same result, and more precisely that a sequence of fusions coincides step-by-step to an execution of the Martelli-Montanari unification algorithm [60] has a straightforward proof by induction on the number of links in the diagram. However, the proof is omitted in this paper.

Definition 19 (Correct diagram and actualisation)**.** Consider a constellation Φ and a set of colours $A \subseteq \texttt{Colours}$. An A-diagram δ is correct if $\texttt{free}(\delta) \neq \emptyset$ and the associated unification problem $\mathcal{P}(\delta)$ has a solution.

The *actualisation* of a correct diagram δ is the star $\Downarrow\delta$ defined as $|\Downarrow\delta| = \texttt{free}(\delta)$ and $\texttt{gen}_{\Downarrow\delta} : (e, r) \in \texttt{free}(\delta) \mapsto \psi(\theta(e, r))$, where $\psi = \texttt{Solution}(\mathcal{P}(\delta))$ and $\theta(e, \_)$ is the renaming used in Definition 16.

Example 20. Since it is more intuitive, we illustrate the reduction by fusion of the diagram of Figure 2b. The actualisation produce exactly the same result. In case of possible unification failure, the diagram is not correct and cannot be actualised.

Notation 21. We write $\texttt{CSat}^k_A(\Phi)$ for the set of correct and saturated A-diagrams over Φ with k vertices and $\texttt{CSat}^k_A(\Phi)$. We write $\texttt{CSat}^k_{A,tree}(\Phi)$ if we restrict the shape of diagrams to be trees.

Definition 22 (Resolution)**.** The *resolution* of a constellation Φ w.r.t. a set of colours $A \subseteq \texttt{Colours}$ is defined by $\texttt{Res}_A(\Phi) = \bigcup_{k=0}^{\infty} \Downarrow \texttt{CSat}^k_{A,tree}(\Phi)$, where $\Downarrow \texttt{CSat}^k_{A,tree}(\delta) := \{\Downarrow\delta \mid \delta \in \texttt{CSat}^k_{A,tree}(\delta)\}$ (we restrict the diagrams to trees). We simply write $\texttt{Res}(\Phi)$ when considering all the colours appearing in Φ.

Example 23. We have $\texttt{Res}(\Phi^{2+2}_{\mathbf{N}}) = [\underline{4}]$ since the only correct and saturated diagram is the diagram of Figure 2b and that it evaluates to $[\underline{4}]$ according to Example 20. All other diagrams will fail. More generally, $\texttt{Res}(\Phi^{n+m}_{\mathbf{N}}) = [\underline{n+m}]$. Even more generally, if Φ_P is a logic program and ϕ_Q a query, then $\texttt{Res}(\Phi_P + \phi_Q)$ is the set of answers of the query.

B. Evaluation as tiling construction

In another point of view, the stellar resolution can be seen as a non-deterministic annihilating interaction between kind of molecules (stars) through matching atoms (rays) of opposite polarities, triggering the propagation of a reaction (principal unifier). Our dependency graphs may also be seen as directed hypergraphs of (un)polarised objects. This is very similar to chemical reactions networks [63].

We here focus on a simulation of the *abstract tile assembly model* (aTAM) [48], [49] used in DNA computing [50]. This is the generalisation of a combinatorial model known as Wang tiles [47] in which one constructs tilings from basic square tiles (e.g ,) in $\mathbf{Z}^2$ by connecting the sides of matching colours. They both can be seen as strong geometric restrictions on stellar resolution (diagrams represent a tiling on a grid).

The elementary objects of the aTAM are *tiles* from a set T, which are squares with each sides associated to a *glue type* $g \in G$ and a natural number $\texttt{str}(g)$ called its *strength*. An *assembly* is a partial function $a : \mathbf{Z}^2 \to T$. If $a(x, y) = t_i$ then we can have $a(x', y') = t_i$ adjacent to t_i (i.e $(x', y') \in \{(x-1, y), (x+1, y), (x, y-1), (x, y+1)\}$) through the sides g^i_d and $g^j_{d'}$ if $d, d' \in \{n, s, w, e\}$ are opposite directions (w.r.t. n/s and w/e) and both their colour and strength match (i.e $g^i_d = g^j_{d'}$ and $\texttt{str}(g^i_d) = \texttt{str}(g^j_{d'})$).

The computation starts with an initial assembly s called the *seed* and a non-negative natural number τ called the *temperature*. If a tile t can be connected to sides of glue types $g_1, ..., g_n$ in s then t can be added to a if $\sum_{i=0}^{n} \texttt{str}(g_i) \geq \tau$. We are then interested in all the saturated assemblies (which cannot be extended). The temperature acts as a threshold for a degree of cooperation. If we have $\tau = 1$, the model is said to be *non-cooperative* [64]. We illustrate an example in the Figure 4 with a tile connecting to an assembly.

Definition 24 (Encoding of tiles)**.** Let $t_i = (g^i_w, g^i_e, g^i_s, g^i_n)$ be a tile with $i \in I$, a countable but potentially infinite set. We define $gl(g)(x) := g(x) \cdot \underline{str(g)}$ with $str(g) \in \mathbf{N}$. We define the encoding $t^{\bigstar}_i$ of t_i by:

$$[-\overset{\bullet}{h}(\text{gl}(g^i_w)(x), x, y), -\overset{\bullet}{v}(\text{gl}(g^i_s)(y), x, y),$$
$$+\overset{\circ}{h}(\text{gl}(g^i_e)(s(x)), s(x), y), +\overset{\circ}{v}(\text{gl}(g^i_n)(s(y)), x, s(y))].$$

Definition 25 (Environment constellation)**.** The environment constellation for a temperature $\tau \in \mathbf{N}\backslash\{0\}$ is defined by $\Phi^{\tau}_{env} :=$

$$[+temp(\underline{\tau})]+$$
$$\begin{bmatrix} +\overset{\bullet}{v}(g_1(x_1) \cdot n_1, x_1, y_1), & -\overset{\circ}{v}(g_2(x_3) \cdot n_2, x_3, y_3), \\ +\overset{\bullet}{h}(g_3(x_5) \cdot n_3, x_5, y_5), & -\overset{\circ}{h}(g_4(x_7) \cdot n_4, x_7, y_7), \\ -\overset{\circ}{v}(g_1(x_2) \cdot n_1, x_2, y_2), & +\overset{\bullet}{v}(g_2(x_4) \cdot n_2, x_4, y_4), \\ -\overset{\circ}{h}(g_3(x_6) \cdot n_3, x_6, y_6), & +\overset{\bullet}{h}(g_4(x_8) \cdot n_4, x_8, y_8), \\ -add(n_1, n_2, r_1), & -add(n_3, n_4, r_2), \\ -add(r_1, r_2, r), -geq(r, t, \underline{1}), -temp(t) & \end{bmatrix}$$

$$+[-\overset{\bullet}{v}(g(x)\cdot\underline{0}, x, y)]+[+\overset{\circ}{v}(g(x)\cdot\underline{0}, x, y)]+[-\overset{\bullet}{h}(g(x)\cdot\underline{0}, x, y)]$$

$$+[+\mathring{h}(g(x)\cdot\underline{0},x,y)]+[+\mathring{v}(g(x)\cdot\underline{0},x,y)]+[-\bullet\!\!\!\!\circ v(g(x)\cdot\underline{0},x,y)]$$

$$+[+\mathring{h}(g(x)\cdot\underline{0},x,y)]+[-\bullet\!\!\!\!\circ h(g(x)\cdot\underline{0},x,y)]$$

$$+[+geq(\underline{0},\underline{0},\underline{1})]+[+geq(s(x),s(y),r),-geq(x,y,r)]$$

$$+[+geq(s(x),\underline{0},\underline{0})]+[+geq(\underline{0},s(y),\underline{0})]$$

$$+[+add(\underline{0},y,y)]+[+add(s(x),y,s(z)),-add(x,y,z)]$$

The representation of a tile $t^\star$ is connected to a tiling through auxiliary ports of the environment in order to ensure that the sum of strengths of the connexions is a least τ. We stress how the additions are performed within the model: the attentive reader will recognise part of the constellation $\Phi_{\mathbf{N}}^{n+m}$ implementing addition. We need several unary stars in order to plug the remaining unconnected auxiliary ports of the environment.

Definition 26 (Connections)**.** The set of *connections* of a constellation Φ w.r.t. a set of colours $A \subseteq \texttt{Colours}$ is defined by $\texttt{Connect}_A(\Phi) = \bigcup_{k=0}^{\infty} \texttt{CSat}_A^k(\Phi)$. We simply write $\texttt{Connect}(\Phi)$ when considering all the colours appearing in Φ.

Theorem 27 (Simulation of finite aTAM)**.** *Let T be a set of tiles. The set of non-empty finite assemblies constructible from T at temperature τ is bijective to $\texttt{Connect}(T^\star \uplus \Phi_{env}^\tau)$.*

C. Properties of the two models

Proposition 28 (Turing-completeness)**.** *The stellar resolution can simulate non-deterministic Turing machines.*

Proof. The stellar resolution with Res is Turing-complete by reduction to Horn clauses [65], [66]. As for Connect, it is by simulation of aTAM, since aTAM [48], [67]. □

We fix a generic execution $\texttt{Ex}_A(\Phi)$ which can coincide with either $\texttt{Res}_A(\Phi)$ or $\Downarrow\texttt{Connect}_A(\Phi)$.

Definition 29 (Strong normalisation)**.** A constellation Φ is strongly normalising w.r.t. a set of colours $A \subseteq \texttt{Colours}$ if and only if $\texttt{Ex}(\Phi)$ is a finite constellation. When $A = \texttt{Colours}$, we simply say that Φ is strongly normalising.

Theorem 30 (Confluence)**.** *Let Φ be a constellation, and A, B be two disjoint sets of colours, i.e. $A, B \subseteq \texttt{Colours}$ and $A \cap B = \emptyset$. We have:*

$$\texttt{Ex}_B(\texttt{Ex}_A(\Phi)) = \texttt{Ex}_{A\cup B}(\Phi) = \texttt{Ex}_A(\texttt{Ex}_B(\Phi)).$$

Remark 31. In Girard's first paper on Transcendental Syntax [38], the constellation $\Phi = [+a.x, -a.x, +b.x]$ is mentioned as a counter-example for the confluence of $\texttt{Res}(\_)$. Here, we have $\texttt{Res}_{\{a\}}(\texttt{Res}_{\{b\}}(\Phi)) = \texttt{Res}_{\{b\}}(\texttt{Res}_{\{a\}}(\Phi)) = \emptyset$ (because no saturated diagram on a nor on b can be constructed). Our understanding of Girard's failure comes from his limitation to strongly normalising constellations, so that $\texttt{Res}_{\{a\}}(\Phi)$ was not defined.

IV. Interpreting the dynamics of proofs

Mathematical proofs, despite their static appearance, have a dynamic side. The cut rule of mathematical logic:

$$\frac{\Gamma \vdash A \qquad \Gamma', A \vdash C}{\Gamma, \Gamma' \vdash C}\ \text{cut}$$

represents the use of lemma or auxiliary theorems in a proof: to prove a statement, we may prove a formula A and show that A leads to the conclusion we want. As stated by the cut-elimination theorem (the so-called *Hauptsatz* [1], [2]), this rule is not necessary. Similarly to the fact that imperative programs do not need functions, one can choose to remove cuts by proving the lemmas each time they are needed. This cut-elimination procedure is defined concretely and corresponds to the execution of programs, leading to an idea of *execution of proof* which "unfolds" the use of lemmas.

While this can be expressed in a sequent calculus, Linear logic [6] possesses an alternative syntax for proofs: proof-nets, a hypergraph linking occurrences of formulas. Proof nets identify some sequent calculus proofs modulo commutation of rules, and is therefore closer to program execution. More than a mere syntactical convenience, proof-nets reveal the geometric mechanisms of cut-elimination: the cut rule becomes a simple bridge (hyperedge) between two formulas and the bureaucratic cut-elimination procedure from sequent calculus becomes simple graph-rewriting rules.

A. Multiplicative Linear Logic

Multiplicative linear logic (MLL) is a fragment of linear logic [6] restricted to the tensor $\otimes$ and par $\parr$ connectives. The set $\mathcal{F}_{\text{MLL}}$ of MLL formulas is defined by the grammar of the Figure 5a. Linear negation $(\cdot)^\perp$ is extended to formulas by involution and de Morgan laws: $X^{\perp\perp} = X$, $(A \otimes B)^\perp = A^\perp \otimes B^\perp$, and $(A \parr B)^\perp = A^\perp \parr B^\perp$.

Proofs of MLL can be written in the traditional sequent calculus fashion [1], [2] using the set of rules shown in Figure 5b. However, we choose to work with Girard's proof-structures [6], an alternative and more general syntax, akin to *natural deduction*, based on a graph-theoretic representation of proofs. In this syntax, we consider directed hypergraph with vertices labelled by formulas and constructed from hyperedges[3] labelled within $\{ax, cut, \otimes, \parr\}$ and satisfying the arities and labelling constraints shown in Figure 5c. A proof-structure also satisfies the additional constraint that each vertex must be (1) the target of exactly one hyperedge, and (2) the source of at most one hyperedge. When needed, a proof-structure will be defined as a 6-tuple $(V, E, s, t, \ell_V, \ell_E)$, where (V, E, s, t) is a directed hypergraph and $\ell_V : V \to \mathcal{F}_{\text{MLL}}$, $\ell_E : E \to \{\otimes, \parr, \text{ax}, \text{cut}\}$ are labelling maps.

The cut-elimination procedure, which is defined in the natural way for MLL sequent calculus, becomes a graph-rewriting system on proof-structures, defined by the two rewriting rules

[3]For practical purposes, the source edges are ordered, and we will talk about the "left" and "right" sources since there never are more than two; illustrations implicitly represent the left (resp. right) source on the left (resp. right).

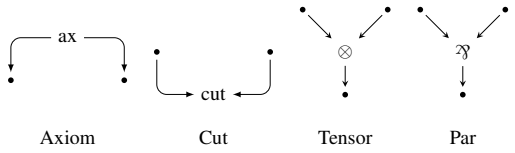
$$A, B = X_i \mid X_i^\perp \mid A \otimes B \mid A \parr B \quad i \in \mathbf{N} \qquad (\mathcal{F}_{\mathrm{MLL}})$$

(a) MLL Formulas

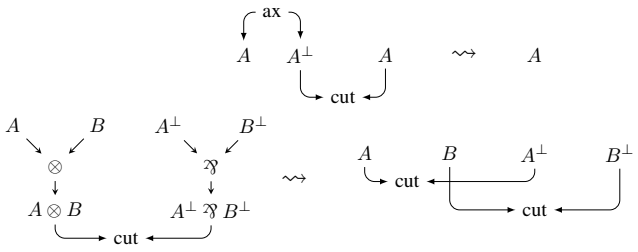
$$\frac{}{\vdash A, A^\perp}\ \mathrm{ax} \qquad \frac{\vdash \Gamma, A \quad \vdash \Delta, B}{\vdash \Gamma, \Delta, A \otimes B}\ \otimes$$

$$\frac{\vdash \Gamma, A \quad \vdash \Delta, A^\perp}{\vdash \Gamma, \Delta}\ \mathrm{cut} \qquad \frac{\vdash \Gamma, A, B}{\vdash \Gamma, A \parr B}\ \parr$$

(b) MLL sequent calculus rules



(c) MLL proof-structures



(d) Cut-elimination reductions

Fig. 5: Syntax of Multiplicative Linear Logic (MLL)

(Figure 5d). The following definition explains how sequent calculus proofs can be represented as proof-structures.

Definition 32 (Translation of MLL sequent calculus)**.** In Figure 6, we define a translation $[\![\cdot]\!]$ from MLL sequent calculus derivations to proof-structures.

Note that this translation is not surjective, and some proof-structures do not represent sequent calculus proofs. This is tackled by the *correctness criterion* which characterises those proof-structures that do translate sequent calculus proofs through topological properties. This is discussed in the next section but for the time being we give a preliminary definition of proof-net, the proof-structures coming from sequent proofs.

Definition 33 (Proof-nets)**.** A *proof-net* is a proof-structure $\mathcal{S}$ such that there exists a MLL sequent calculus proof π with $\mathcal{S} = [\![\pi]\!]$.

In the next two sections, we reconstruct proof-structures within the stellar resolution. This interpretation is based on the execution $\mathtt{Ex}(\Phi)$. We implicitly assume that $\mathtt{Ex}(\Phi) := \mathtt{Res}(\Phi)$ but later observe (see Remark 52) that $\mathtt{Ex}(\Phi) := \Downarrow \mathtt{Connect}(\Phi)$ can also be considered, thus leading to two alternative interpretation of multiplicative linear logic.

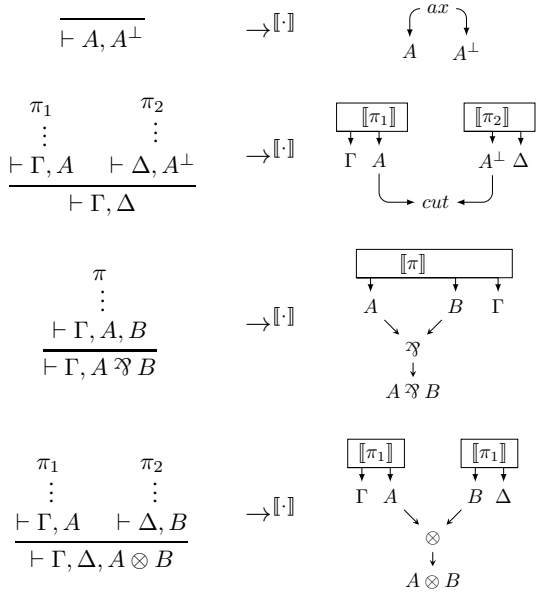


Fig. 6: Translation of sequent proofs into proof-structures.

B. Reconstruction of cut-elimination

In order to encode proof-structures, we set a *basis of representation*, which is a coloured signature defined by $\mathbb{B} = (\{x\}, \{c, t\}, \{c, t, \mathtt{l}, \mathtt{r}, \cdot, p_A, q_A\}, \mathtt{arity})$ for $A \in \mathcal{F}_{\mathrm{MLL}}$, $c, t, p_A, q_A, \mathtt{l}, \mathtt{r}$ with arity 1, $\mathtt{arity}(\cdot)$ with arity 2 and $\mathtt{g}$ with arity 0.

Notation 34. Let $\mathcal{S}$ be a proof-structure. We write $\mathtt{Ax}(\mathcal{S})$ (resp. $\mathtt{Cut}(\mathcal{S})$) the set of axioms (resp. cut) hyperedges in $\mathcal{S}$. Given $e \in \mathtt{Ax}(\mathcal{S})$ ($e \in \mathtt{Cut}(\mathcal{S})$), we write $\mathcal{A}_e^l$ and $\mathcal{A}_e^r$ the left and right conclusions (resp. sources) of e respectively.

Let us remark that proof-structures can be defined inductively. A proof-structure with only one hyperedge is necessarily an axiom with two conclusions. Then a proof-structure with n hyperedges is either built from the union of two proof-structures with respectively k and $n-k$ hyperedges, or from a proof-structure with $n-1$ hyperedges extended by either a $\otimes$, $\parr$, or *cut* hyperedge on two of its conclusions. In the following, we use this inductive definition to define the *address* of occurrences of atoms in a proof-structure.

Definition 35. A vertex v is *above* another vertex u in a proof-structure if there exists a directed path from v to u going through only $\otimes$ and $\parr$ hyperedges.

We would like to localise the atoms inside a proof-structure and associate them uniquely with an address encoded in $\mathbb{B}$.

Definition 36 (Address)**.** We define the partial address $\mathtt{pAddr}_{\mathcal{S}}(d, x)$ of an occurrence of atom d in a MLL proof-structure $\mathcal{S}$, with respect to the variable x, inductively[4]:

[4]The set of formulas $\mathcal{F}_{\mathrm{MLL}}$ is countable, and there are only finite numbers of occurrences of a given formula in a given proof-structure, hence the set $\mathcal{F}_{\mathrm{MLL}} \times \mathbf{N}$ suffices and is still countable.

- $\texttt{pAddr}_{\mathcal{S}}(d,x) = x$ when $\mathcal{S}$ consists only of an axiom hyperedge;
- $\texttt{pAddr}_{\mathcal{S}}(d,x) = \texttt{pAddr}_{\mathcal{S}_i}(d,x)$ if $\mathcal{S}$ is the union of two smaller proof-structures $\mathcal{S}_1, \mathcal{S}_2$ and d appears in $\mathcal{S}_i$;
- $\texttt{pAddr}_{\mathcal{S}}(d,x) = \texttt{l}(\texttt{pAddr}_{\mathcal{S}'}(d,x))$ (resp. $\texttt{pAddr}_{\mathcal{S}}(d,x) = \texttt{r}(\texttt{pAddr}_{\mathcal{S}'}(d,x))$) if $\mathcal{S}$ is obtained from $\mathcal{S}'$ by adding a $\otimes$ or $\parr$ hyperedge e, and if d is above the left source (resp. the right source) of e.
- $\texttt{pAddr}_{\mathcal{S}}(d,x) = \texttt{pAddr}_{\mathcal{S}'}(d,x)$ otherwise.

The partial address of d is defined with respect to either a conclusion of the structure of the source of a cut hyperedge, which is uniquely defined as the occurrence of formula c such that d is above c and c is not source of either a $\otimes$ or a $\parr$; the *address* of d is then defined as the term $\texttt{addr}_{\mathcal{S}}(d,x) = p_c(\texttt{pAddr}_{\mathcal{S}}(d,x))$.

We define $\mathrm{Addr}_x(\mathcal{S})$ as the *set of addresses* $\texttt{addr}_{\mathcal{S}}(\_, x)$, i.e the countable set of all terms of the form $p_A(t(x))$ where A ranges over conclusions of $\mathcal{S}$ and $t(x)$ is a term constructed from the unary symbols $\texttt{l}$ and $\texttt{r}$.

Definition 37 (Vehicle and cuts)**.** The *vehicle* and the *cuts* of a proof-structure $\mathcal{S}$ are respectively defined by the following constellations:

$$\Phi^{\mathrm{ax}}_{\mathcal{S}} := \sum_{e \in \texttt{Ax}(\mathcal{S})} [\texttt{addr}_{\mathcal{S}}(\mathcal{A}^l_e), \texttt{addr}_{\mathcal{S}}(\mathcal{A}^r_e)];$$

$$\Phi^{\mathrm{cut}}_{\mathcal{S}} := \sum_{e \in \mathrm{Cut}(\mathcal{S})} [p_{\mathcal{A}^l_e}(x), p_{\mathcal{A}^r_e}(x)].$$

We define a function of colouration in order to enable an interaction between the uncoloured rays of two constellations. We also use a function of discolouration in order to keep a canonical representation of proofs.

Definition 38 (Colouration of constellation)**.** The *colouration* $\pm c.\Phi$ of a constellation Φ with a colour $c \in \mathcal{C}$ and a polarity $\pm \in \{+,-\}$ updates all the uncoloured rays of its stars by the function: $\varphi^{\pm}_c(t) = \pm c(t)$ when t is uncoloured and $\varphi^{\pm}_c(r) = r$ otherwise.

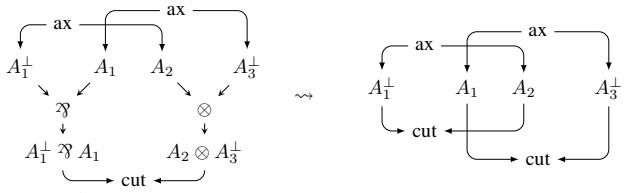
Definition 39 (Discolouration of constellation)**.** The *discolouration* $\natural\Phi$ of a constellation Φ updates all the rays of its stars by the function: $\natural(+c(t)) = \natural(-c(t)) = \natural(t) = t$.

Lemma 40. *Let $\mathcal{R}$ be a MLL proof-structure such that $\mathcal{R} \rightsquigarrow \mathcal{S}$. We have* $\texttt{Ex}(+c.\Phi^{\mathrm{ax}}_{\mathcal{R}} \uplus -c.\Phi^{\mathrm{cut}}_{\mathcal{R}}) = \texttt{Ex}(+c.\Phi^{\mathrm{ax}}_{\mathcal{S}} \uplus -c.\Phi^{\mathrm{cut}}_{\mathcal{S}})$.

Theorem 41 (Dynamics)**.** *For a proof-net $\mathcal{R}$ of normal form $\mathcal{S}$, we have* $\natural\texttt{Ex}(+c.\Phi^{\mathrm{ax}}_{\mathcal{R}} \uplus -c.\Phi^{\mathrm{cut}}_{\mathcal{R}}) = \mathcal{S}^{\bigstar}$.

Proof. This result is a consequence of Lemma 40 by induction of the number of cut-elimination steps from $\mathcal{R}$ to $\mathcal{S}$, as well as the fact that $\natural\texttt{Ex}(+c.\Phi^{\mathrm{ax}}_{\mathcal{S}} \uplus -c.\Phi^{\mathrm{cut}}_{\mathcal{S}}) = \texttt{Ex}(\mathcal{S}^{\bigstar}) = \mathcal{S}^{\bigstar}$ since $\mathcal{S}$ does not contain cuts. □

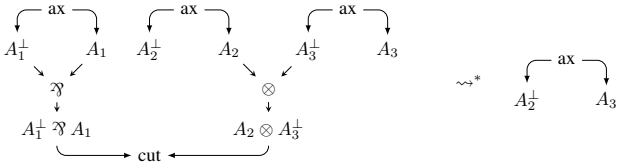
Example 42. Take the following reduction $\mathcal{S} \rightsquigarrow \mathcal{S}'$ of proof-structure:



Then, we have:

$$\begin{aligned}+c.\Phi^{\mathrm{ax}}_{\mathcal{S}} \uplus -c.\Phi^{\mathrm{cut}}_{\mathcal{S}} = {} & [+c.p_{A_1^\perp \parr A_1}(\texttt{l}x), +c.p_{A_2 \otimes A_3^\perp}(\texttt{l}x)] \\ & + [+c.p_{A_2 \otimes A_2^\perp}(\texttt{r}x), +c.p_{A_1^\perp \parr A_1}(\texttt{r}x)] \\ & + [-c.p_{A_1^\perp \parr A_1}(x), -c.p_{A_2 \otimes A_3^\perp}(x)]\end{aligned}$$

and $\natural\texttt{Ex}(+c.\Phi^{\mathrm{ax}}_{\mathcal{S}} \uplus -c.\Phi^{\mathrm{cut}}_{\mathcal{S}}) = \emptyset$ because the diagrams we would like to construct, by duplicating the cut, are closed (c.f Notation 14). If we look at the following reduction $\mathcal{S} \rightsquigarrow^* \mathcal{S}'$ instead:



we have:

$$\begin{aligned}+c.\Phi^{\mathrm{ax}}_{\mathcal{S}} \uplus -c.\Phi^{\mathrm{cut}}_{\mathcal{S}} = {} & [+c.p_{A_1^\perp \parr A_1}(\texttt{l}x), +c.p_{A_1^\perp \parr A_1}(\texttt{r}x)] \\ & + [+c.p_{A_2^\perp}(x), +c.p_{A_2 \otimes A_3^\perp}(\texttt{l}x)] \\ & + [+c.p_{A_2 \otimes A_3^\perp}(\texttt{r}x), +c.p_{A_3}(x)] \\ & + [-c.p_{A_1^\perp \parr A_1}(x), -c.p_{A_2 \otimes A_3^\perp}(x)]),\end{aligned}$$

and $\natural\texttt{Ex}(+c.\Phi^{\mathrm{ax}}_{\mathcal{S}} \uplus -c.\Phi^{\mathrm{cut}}_{\mathcal{S}}) = [p_{A_2^\perp}(x), p_{A_3}(x)] = \mathcal{S}'^{\bigstar}$.

V. Interpreting the logical content

A. Correctness of proof-structures

As mentioned above, proof-structures are more permissive than sequent calculus proofs. In other words, some proof-structures do not represent proofs, and the syntax of MLL is therefore restricted to proof-nets, i.e. proof-structures that do represent sequent calculus proofs. A beautiful result of Girard, analysed by many subsequent works [8], [68], [69], [70], [9], [71], [72], is that those proof-structures that are proof-nets can be characterised by a topological/combinatorial property called a *correctness criterion*. While Girard's original criterion, called the long-trip criterion [6], is about the set of walks in a proof-structure, we will here work with Danos and Regnier's simplified criterion [8] which is the most standard.

Notation 43. Given a proof-structure $\mathcal{S} = (V, E, s, t, \ell_V, \ell_E)$, we write $\parr(\mathcal{S})$ the subset $P \subseteq E$ of $\parr$-labelled edges, i.e. $\parr(\mathcal{S}) = \{e \in E \mid \ell_E(e) = \parr\}$.

We now define *correctness graphs*, which are the undirected hypergraphs obtained by removing one source of each $\parr$-labelled edge. The Danos-Regnier criterion then states that a proof-structure is a proof-net if and only if all correctness graphs are trees.

Definition 44 (Correctness graph)**.** Let $\mathcal{S} = (V, E, s, t, \ell_V, \ell_E)$ be a proof-structure. A switching is

a map $\varphi : \mathfrak{P}(\mathcal{S}) \to \{l, r\}$. The correctness hypergraph $\mathcal{S}_\varphi$ is the undirected hypergraph (V, E, s') induced by the switching φ is defined by letting $s'(e) = \{v\} \cup t(e)$ where v is the left (resp. right) source of v in $\mathcal{S}$ when $e \in \mathfrak{P}(\mathcal{S})$ and $\varphi(e) = l$ (resp. $\varphi(e) = r$), and $s'(e) = s(e) \cup t(e)$ for $e \notin \mathfrak{P}(\mathcal{S})$.

Theorem 45 (Danos-Regnier correctness criterion [8])**.** *A proof-structure $\mathcal{S}$ is a proof-net if and only if $\mathcal{S}_\varphi$ is a tree for all switching φ.*

Remark 46. Each correctness graph can be defined as the union of two graphs: one which comes from the axioms and is uniquely defined by the proof-structure, and one which is obtained from edges that are not axioms and is dependent on the switching. This point of view allows for an *interactive* formulation of the correctness criterion in which the set of axioms is *tested* against graphs corresponding to switchings [73] in order to certify that the proof-structure is a proof-net.

B. Reconstruction of correctness

We have already seen in the previous section how constellations can represent proofs. We now explain how to define tests to allow for an interactive, internal, representation of the correctness criterion. This is done by translating the Danos-Regnier criterion within the framework of stellar resolution. We translate the lower part of switching graphs and consider their connexion with a vehicle.

We now use three colours: c (computation), t (testing) and f (format). A vehicle will be coloured with the colour c when we want its execution by connecting it with cuts and it will be coloured with the colour t when being subject to tests against ordeals. The colour f corresponds to the internal connexions of an ordeal.

Definition 47 (Ordeal)**.** Let $\mathcal{S}$ be a MLL proof-structure and φ one of its switchings. The ordeal $\mathcal{S}_\varphi^\star$ associated to $\mathcal{S}_\varphi$ is the constellation obtained by translating all the vertices of $\mathcal{S}_\varphi$ in the following way:

- $(\mathcal{A}_e^d)^\star = [-t.\mathtt{addr}_S(\mathcal{A}_e^d), +f.q_{\mathcal{A}_e^d}(x)]$ for $e \in \mathrm{Ax}(\mathcal{S})$,
- $(A \parr_L B)^\star = [-f.q_A(x)] + [-f.q_B(x), +f.q_{A \parr B}(x)]$,
- $(A \parr_R B)^\star = [-f.q_A(x), +f.q_{A \parr B}(x)] + [-f.q_B(x)]$,
- $(A \otimes B)^\star = [-f.q_A(x), -f.q_B(x), +f.q_{A \otimes B}(x)]$,
- We add $[-f.q_A(x), p_A(x)]$ for each conclusion A.

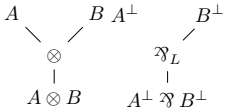
We define $q_A(t)$ as a shortcut for $p_A(\mathtt{g} \cdot t)$ with $\mathtt{g}$ a constant only used for that definition so that the cut can act on the ordeals while being disjoints to vehicles.

Theorem 48 (Stellar correctness criterion)**.** *A proof-structure $\mathcal{S}$ is a proof-net if and only if for all switchings φ, we have $\natural\mathtt{Ex}(+t.\Phi_\mathcal{S}^{\mathrm{ax}} \uplus -c.\Phi_\mathcal{S}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star) = [p_{A_1}(x), ..., p_{A_n}(x)]$ where $A_1, ..., A_n$ are the conclusions of $\mathcal{S}$.*

Corollary 49 (Corollary of Theorem 48)**.** *All correctness graphs of a proof-structure $\mathcal{S}$ are:*

- *acyclic if and only if $\mathfrak{D}[+t.\Phi_\mathcal{S}^{\mathrm{ax}} \uplus -c.\Phi_\mathcal{S}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star; -]$ is acyclic, hence $+t.\Phi_\mathcal{S}^{\mathrm{ax}} \uplus -c.\Phi_\mathcal{S}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star$ is strongly normalising and*
- *trees if and only if $\mathfrak{D}[+t.\Phi_\mathcal{S}^{\mathrm{ax}} \uplus -c.\Phi_\mathcal{S}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star; -]$ is a tree, hence $+t.\Phi_\mathcal{S}^{\mathrm{ax}} \uplus -c.\Phi_\mathcal{S}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star$ normalises into a single star.*

Example 50. We translate an ordeal and show how it behaves when connected with a right vehicle.



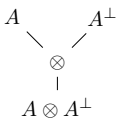
$$\begin{bmatrix} -t.p_{A\otimes B}(\mathtt{l}x) \\ +f.q_A(x) \end{bmatrix} + \begin{bmatrix} -t.p_{A\otimes B}(\mathtt{r}x) \\ +f.q_B(x) \end{bmatrix} + \begin{bmatrix} -t.p_{A^\perp \parr B^\perp}(\mathtt{l}x) \\ +f.q_{A^\perp}(x) \end{bmatrix} + \begin{bmatrix} -t.p_{A^\perp \parr B^\perp}(\mathtt{r}x) \\ +f.q_{B^\perp}(x) \end{bmatrix} + \begin{bmatrix} -f.q_A(x) & -f.q_B(x) \\ +f.q_{A\otimes B}(x) \end{bmatrix} + [-f.q_{A^\perp}(x)] + \begin{bmatrix} -f.q_{B^\perp}(x) \\ +f.q_{A^\perp \parr B^\perp}(x) \end{bmatrix} + \begin{bmatrix} -f.q_{A\otimes B}(x) \\ p_{A\otimes B}(x) \end{bmatrix} + \begin{bmatrix} -f.q_{A^\perp \parr B^\perp}(x) \\ p_{A^\perp \parr B^\perp}(x) \end{bmatrix}$$

Remark that the matchability of rays exactly reproduces the structure of the lower part of the proof-structure. Thanks to the confluence of execution, we can focus the execution on the ordeal. We obtain the following constellation:

$$\begin{bmatrix} -t.p_{A\otimes B}(\mathtt{l}x) & -t.p_{A\otimes B}(\mathtt{r}x) & -t.p_{A^\perp \parr B^\perp}(\mathtt{l}x) \\ & p_{A\otimes B}(x) & \end{bmatrix} + \begin{bmatrix} -t.p_{A^\perp \parr B^\perp}(\mathtt{r}x) \\ p_{A^\perp \parr B^\perp}(x) \end{bmatrix}$$

This contracted ordeal corresponds to a generalisation of the pointed partitions presented by Acclavio and Maieli [74]. When connected to the vehicle $[+t.p_{A\otimes B}(\mathtt{l}x), +t.p_{A^\perp \parr B^\perp}(\mathtt{l}x)] + [+t.p_{A\otimes B}(\mathtt{r}x), +t.p_{A^\perp \parr B^\perp}(\mathtt{r}x)]$, it normalises into $[p_{A\otimes B}(x), p_{A^\perp \parr B^\perp}(x)]$ by forming a tree using all coloured rays.

We now consider the following incorrect proof-structure:



$$\begin{bmatrix} -t.p_{A\otimes A^\perp}(\mathtt{l}x) \\ +f.q_A(x) \end{bmatrix} + \begin{bmatrix} -t.p_{A\otimes A^\perp}(\mathtt{r}x) \\ +f.q_{A^\perp}(x) \end{bmatrix} + \begin{bmatrix} -f.q_A(x) & -f.q_{A^\perp}(x) \\ +f.q_{A\otimes A^\perp}(x) \end{bmatrix} + \begin{bmatrix} -f.q_{A\otimes A^\perp}(x) \\ p_{A\otimes A^\perp}(x) \end{bmatrix}$$

It normalises into

$$[-t.A \otimes A^\perp(\mathtt{l}x), -t.A \otimes A^\perp(\mathtt{r}x), p_{A\otimes A^\perp}(x)]$$

When we connect the ordeal to the vehicle $[+t.p_{A\otimes A^\perp}(\mathtt{l}x), +t.p_{A\otimes A^\perp}(\mathtt{r}x)]$, we can construct infinitely many correct saturated diagrams because of the cycle of the dependency graph. So the vehicle does not satisfy the stellar correctness criterion.

We can finally translate a whole proof-structure into a constellation made of three components. These components allow us to study both the computational and the logical aspects of a proof by using adequate colourings. This decomposition renders explicit the fact that proof-structures come with a kind of pre-made type.

Definition 51 (Translation of proof-structures)**.** The translation of a proof-structure $\mathcal{S}$ is defined as the constellation $\mathcal{S}^\star = \Phi_{\mathcal{S}}^{\text{ax}} \uplus \Phi_{\mathcal{S}}^{\text{cut}} \uplus \Phi_{\mathcal{S}}^{\text{format}}$ where

$$\Phi_{\mathcal{S}}^{\text{format}} := \{\mathcal{S}_\varphi^\star \mid \varphi \text{ is a switching of } \mathcal{S}\}$$

is called the *format* of $\mathcal{S}$ (*gabarit* in Girard's original papers).

In this work, we will only consider connexions between a vehicle and a single element of $\Phi_{\mathcal{S}}^{\text{format}}$ (an *ordeal*) at once. In order to consider a simultaneous connexion between a vehicle and the set of all elements of $\Phi_{\mathcal{S}}^{\text{format}}$, one needs to use a coherence relation on constellations [39].

Remark 52 (Connections as an alternative model). Remark that our results do not change whether we consider $\texttt{Ex}(\Phi) := \texttt{Res}(\Phi)$ or $\texttt{Ex}(\Phi) := \Downarrow \texttt{Connect}(\Phi)$. In the usual theory of linear logic, or in Girard's transcendental syntax, only tree-like connexions are considered, corresponding to $\texttt{Res}(\Phi)$. However, we also consider the less standard case of cyclic diagrams, corresponding to $\Downarrow \texttt{Connect}(\Phi)$. Let us explain why the above results are still correct in this generalised setting.

In the case of cut-elimination for a constellation Φ, any cycle is closed since the stars induced by axioms are binary. Closed diagrams are not correct, hence the only correct diagrams are tree-like and $\Downarrow \texttt{Connect}(\Phi) = \texttt{Res}(\Phi)$.

As for the correctness, cycles are created by connecting the vehicle to an ordeal, each one being a forest. Notice that only exact diagrams are considered (c.f Notation 14) because the axioms match exactly by design, and only the variable x is used, forcing equal connexions everywhere. It follows that if a cycle passes through a conclusion, one can construct infinitely many correct and saturated diagrams by duplicating stars and construct arbitrarily large cycles. Otherwise, the cycle is necessarily closed since the conclusion are the only free rays, hence not correct. Therefore, we also have $\Downarrow \texttt{Connect}(\Phi) = \texttt{Res}(\Phi)$ for the correctness criterion.

This shows that $\texttt{Connect}(\Phi)$ which is expressive enough for tiling-based computation is as valid as $\texttt{Res}(\Phi)$ as a model of multiplicative linear logic.

C. Reconstruction of connectives

We are now interested in the construction of formulas/types. For that purpose, we follow standard realisability constructions for linear logic [75], [76], [77]. Note that we explicit the *trefoil property* [28] instead of the special case that is usually called *adjunction*. The definition of formula is based on a notion of orthogonality which opposes constellations w.r.t. a specific point of view. The choice of orthogonality influences the notion of formula we obtain.

Definition 53. A *pre-type* is a non-empty set of constellations.

Definition 54 (Orthogonality)**.** We define two notions of orthogonality:

- Two constellations Φ_1, Φ_2 are finitely orthogonal w.r.t. a set of colours $A \subseteq \texttt{Colours}$, written $\Phi_1 \perp_A^{\text{fin}} \Phi_2$, when $|\texttt{Ex}_A(\Phi_1 \uplus \Phi_2)|$ is finite. The orthogonal of a pre-type is defined by:
$$\mathbf{A}^{\perp_A^{\text{fin}}} = \{\Phi \mid \forall \Phi' \in \mathbf{A}, \Phi \perp_A^{\text{fin}} \Phi'\};$$
- Similarly, Φ_1, Φ_2 are 1-orthogonal w.r.t. $A \subseteq \texttt{Colours}$, written $\Phi_1 \perp_A^1 \Phi_2$, when $|\texttt{Ex}_A(\Phi_1 \uplus \Phi_2)| = 1$, and we define:
$$\mathbf{A}^{\perp_A^1} = \{\Phi \mid \forall \Phi' \in \mathbf{A}, \Phi \perp_A^1 \Phi'\}.$$

The orthogonality $\perp^{\text{fin}}$ will define a fully complete model of MLL+MIX, while $\perp^1$, will define a fully complete model of MLL. However, those two notions of orthogonality share most of the properties needed, and we therefore use the generic notation $\perp$ in the following to state results valid for both.

In order to allow partial evaluation, the orthogonality relation $\perp_A$ is parametrised by a set of colours A. We omit this parameter when considering all colours in $\texttt{Colours}$.

We now define the locations to which a ray (and more generally a star, a constellation and a pre-type) refers to. The idea is that a term $f(x)$ is a finite representation of $\{f(t) \mid t \in \texttt{Terms}(\mathbb{S})\}$ for a given signature $\mathbb{S}$. Hence, $f(t), f(u)$ and $f(x)$ refer to the same general location $f(x)$.

Notation 55. Let $(P, \preceq)$ be a partially ordered set, and $X \subseteq P$. We write $\texttt{prefix}(X)$ the set of *prefixes* in X i.e $\texttt{prefix}(X) = \{x \in X \mid \forall y \in X, x \neq y \Rightarrow \neg(y \prec x)\}$.

Definition 56 (Order on rays)**.** We define the following partial order: given r, r' two rays, $r \preceq r'$ if and only if there exists a substitution θ such that $\theta r = r'$. We consider the order up-to-renaming i.e $r = r'$ when $r \approx_\alpha r'$.

We leave the verification that this defines a partial order to the reader. More intuitively, we have $r \preceq r'$ when r is less specialised (thus more general) than r'.

Definition 57 (Location)**.** We define:

- the location of a star $\texttt{gen}_\phi : |\phi| \to \texttt{Rays}(\mathbb{C})$ as the set
$$\sharp\, \phi := \texttt{prefix}(\{\texttt{gen}_\phi(s) \mid s \in |\phi|\});$$
- the location of a constellation $\texttt{gen}_\Phi : |\Phi| \to \texttt{Stars}(\mathbb{S})$ as the set
$$\sharp\, \Phi := \texttt{prefix}(\cup_{\phi \in |\Phi|} \sharp\, \texttt{gen}_\Phi(\phi));$$
- the location of a set $\mathbf{A}$ of constellations as the set
$$\sharp\, \mathbf{A} := \texttt{prefix}(\cup_{\Phi \in \mathbf{A}} \sharp\, \Phi).$$

Definition 58 (Type)**.** A pre-type $\mathbf{A}$ is a *type* w.r.t. a set of colours $A \subseteq \texttt{Colours}$ if there exists a pre-type $\mathbf{B}$ such that $\mathbf{A} = \mathbf{B}^{\perp_A}$.

Proposition 59 (Bi-orthogonal closure)**.** *A pre-type* $\mathbf{A}$ *is a type w.r.t. a set of colours* $A \subseteq \texttt{Colours}$ *if and only if* $\mathbf{A} = (\mathbf{A}^{\perp_A})^{\perp_A}$.

Definition 60 (Intersection up to unification)**.** Let R and Q be sets of rays. We define their *intersection up to unification* as the set:

$$R \Cap Q = \texttt{prefix}(\{m \in \texttt{Rays}(\mathbb{C}) \mid \exists r \in R, q \in Q,$$

$$r \preceq m \text{ and } q \preceq m\}).$$

We say that R and Q are *disjoint* when $R \Cap Q = \emptyset$; by extension, we say that two pre-types $\mathbf{A}, \mathbf{B}$ are disjoint when $\sharp\,\mathbf{A} \Cap \sharp\,\mathbf{B} = \emptyset$.

Definition 61 (Tensor)**.** Let $\mathbf{A}, \mathbf{B}$ be disjoint types. We define their tensor by

$$\mathbf{A} \otimes_A \mathbf{B} = (\{\Phi_1 \uplus \Phi_2 \mid \Phi_1 \in \mathbf{A}, \Phi_2 \in \mathbf{B}\}^{\perp_A})^{\perp_A}.$$

Proposition 62 (Associativity/commutativity)**.** *Given $\mathbf{A}, \mathbf{B}, \mathbf{C}$ pairwise disjoint types w.r.t. a set of colours $A \subseteq \mathtt{Colours}$, we have $\mathbf{A} \otimes_A \mathbf{B} = \mathbf{B} \otimes_A \mathbf{A}$ and $\mathbf{A} \otimes_A (\mathbf{B} \otimes_A \mathbf{C}) = (\mathbf{A} \otimes_A \mathbf{B}) \otimes_A \mathbf{C}$.*

Proof. By definition, we have $\Phi \in \mathbf{A} \otimes_A \mathbf{B}$ if and only if $\Phi \in (\{\Phi_1 \uplus \Phi_2 \mid \Phi_1 \in \mathbf{A}, \Phi_2 \in \mathbf{B}\}^{\perp_A})^{\perp_A}$. By the commutativity of $\uplus$, we have $\Phi \in (\{\Phi_2 \uplus \Phi_1 \mid \Phi_1 \in \mathbf{A}, \Phi_2 \in \mathbf{B}\}^{\perp_A})^{\perp_A}$ which corresponds to $\mathbf{B} \otimes_A \mathbf{A}$. It follows that $\mathbf{A} \otimes_A \mathbf{B} = \mathbf{B} \otimes_A \mathbf{A}$. The same argument can be used for the associativity since the multiset union $\uplus$ is associative. □

Definition 63 (Par and linear implication)**.** Let $\mathbf{A}, \mathbf{B}$ be types w.r.t. a set of colours $A \subseteq \mathtt{Colours}$. We define: $\mathbf{A} \parr_A \mathbf{B} = (\mathbf{A}^{\perp_A} \otimes_A \mathbf{B}^{\perp_A})^{\perp_A}$ and $\mathbf{A} \multimap_A \mathbf{B} = \mathbf{A}^{\perp_A} \parr_A \mathbf{B}$.

As described in several works by Seiller [27], [77], [28], the associativity of execution and the trefoil property, which are stated below, ensure that one can define a $*$-autonomous category with types as objects and vehicles as morphisms.

Theorem 64 (Associativity of execution)**.** *Choose a set of colours $A \subseteq \mathtt{Colours}$. For constellations Φ_1, Φ_2, Φ_3 such that $\sharp\,\Phi_1 \Cap \sharp\,\Phi_2 \Cap \sharp\,\Phi_3 = \emptyset$, we have*

$$\mathtt{Ex}_A(\Phi_1 \uplus \mathtt{Ex}_A(\Phi_2 \uplus \Phi_3)) = \mathtt{Ex}_A(\mathtt{Ex}_A(\Phi_1 \uplus \Phi_2) \uplus \Phi_3).$$

Proof. Since all constellations have disjoint locations, their stars cannot be connected together and we have $\mathtt{CSat}_A(\Phi_2 \uplus \Phi_3) = \mathtt{CSat}_A(\Phi_2) \uplus \mathtt{CSat}_A(\Phi_3)$. These diagrams cannot be connected to the ones of Φ_1 which has its own saturated diagrams. Hence, $\mathtt{CSat}_A(\Phi_1 \uplus \mathtt{Ex}_A(\Phi_2 \uplus \Phi_3)) = \mathtt{CSat}_A(\Phi_1) \uplus \mathtt{CSat}_A(\Phi_2) \uplus \mathtt{CSat}_A(\Phi_3)$. With the same reasoning, we obtain $\mathtt{CSat}_A(\Phi_1 \uplus \mathtt{Ex}_A(\Phi_2 \uplus \Phi_3)) = \mathtt{CSat}_A(\mathtt{Ex}_A(\Phi_1 \uplus \Phi_2) \uplus \Phi_3)$. Hence the result. □

As a consequence of associativity, we obtain the *trefoil property* [28].

Theorem 65 (Trefoil Property)**.** *Choose a set of colours $A \subseteq \mathtt{Colours}$. For all constellations Φ_1, Φ_2, Φ_3 s.t. $\sharp\,\Phi_1 \Cap \sharp\,\Phi_2 \Cap \sharp\,\Phi_3 = \emptyset$:*

$$\Phi_1 \perp_A \mathtt{Ex}_A(\Phi_2 \uplus \Phi_3) \textit{ iff } \mathtt{Ex}_A(\Phi_1 \uplus \Phi_2) \perp_A \Phi_3.$$

The trefoil property implies that for all Φ_f, $\Phi_a \in \mathbf{A}$, and $\Phi_b \in \mathbf{B}^{\perp_A}$:

$$\mathtt{Ex}_A(\Phi_f \uplus \Phi_a) \perp_A \Phi_b \text{ iff } \Phi_f \perp_A \Phi_a \uplus \Phi_b,$$

which proves the following standard result.

Theorem 66 (Alternative linear disjunction)**.** *Let $\mathbf{A}, \mathbf{B}$ be two types w.r.t. a set of colours $A \subseteq \mathtt{Colours}$. We have: $\mathbf{A} \multimap_A \mathbf{B} = \{\Phi_f \mid \forall\ \Phi_a \in \mathbf{A}, \mathtt{Ex}_A(\Phi_f \uplus \Phi_a) \in \mathbf{B}\}$.*

Due to the lack of space, we chose to omit the construction of a $*$-autonomous category, which do not require new proof techniques and involves lots of bureaucratic definitions to deal with locations. We instead prove full soundness and completeness results.

D. A fully complete model of MLL+MIX

We will now define the interpretation of MLL formulas, which depends on a *basis of interpretation*, and then prove full soundness and full completeness for MLL extended by the MIX rule, corresponding to the axiom scheme $A \parr B \multimap A \otimes B$. It is known that the correctness criterion for MLL+MIX consists in taking the Danos-Regnier correctness graphs and checking for acyclicity (but not connectedness) [52].

Theorem 48 shows that asking for a strongly normalising union vehicle/ordeal corresponds to MLL+MIX correctness. This is the key ingredient in the proof of full completeness. In this section, we consider the orthogonality $\perp^{\mathrm{fin}}$ exclusively.

We use a notion of *localised* formulas, following previous works of Seiller [27], [77], [28]: it is defined using the same grammar as MLL formulas, except that variables are of the form $X_i(j)$, where j is a term (here constructed from unary symbols $\mathtt{l}, \mathtt{r}$ and p_A for all occurrences of formulas A) used to distinguish occurrences, and one expect each occurrence to appear at most once in a formula.

Definition 67. A *basis of interpretation* is a function Ω associating to each integer $i \in \mathbf{N}$ a type $\Omega(i)$ in such a way that the types $(\Omega(i))_{i \in \mathbf{N}}$ are pairwise disjoint.

In the next definition, we use the substitutions θ_r and θ_l which are defined as the identity for all variables except for x, and are defined respectively by $\theta_r(x) = \mathtt{r}(x)$ and $\theta_l(x) = \mathtt{l}(x)$. We also fix a bijection $\sigma : \mathrm{Addr}_x(\mathcal{S}) \times \mathbf{N} \to \mathbf{N}$, where $\mathrm{Addr}_x(\mathcal{S})$ is the set of addresses (c.f Definition 36).

Definition 68 (Interpretation of formulas in proof-structures)**.** Given a basis of interpretation Ω, and a MLL formula occurrence A identified by a unique unary function symbol p_A (cf. Definition 36). We define the *interpretation* $I_\Omega(A, t)$ along Ω and a term t inductively:

- $I_\Omega(A, t) = \Omega(\sigma(t, i))$ when $A = X_i$;
- $I_\Omega(A, t) = \Omega(\sigma(t, i))^{\perp^{\mathrm{fin}}}$ when $A = X_i^{\perp}$;
- $I_\Omega(A \otimes B, t) = I_\Omega(A, u) \otimes I_\Omega(B, v)$ where $u = \theta_l(t)$ and $v = \theta_r(t)$;
- $I_\Omega(A \parr B, t) = I_\Omega(A, u) \parr I_\Omega(B, v)$ where $u = \theta_l(t)$ and $v = \theta_r(t)$.

We then define $I_\Omega(A)$ as $I_\Omega(A, p_A(x))$. We extend the interpretation to sequents by letting $I_\Omega(\vdash A_1, ..., A_n) = I_\Omega(A_1) \parr ... \parr_\Omega (A_n)$.

Definition 69 (Proof-like constellations)**.** A constellation Φ is *proof-like* w.r.t. a set of locations $\mathfrak{A}$ if $\sharp\Phi = \mathfrak{A}$ and Φ consists of binary stars only.

Theorem 70 (Full soundness)**.** *Let $\mathcal{S}$ be a MLL+MIX proof-net of conclusion $\vdash \Gamma$ and Ω a basis of interpretation. We have $\mathtt{Ex}(+c.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}}) \in I_\Omega(\vdash \Gamma)$, and $\mathtt{Ex}(+c.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}})$ is proof-like w.r.t. $\sharp\mathcal{S}$.*

Proof. Proved by simple induction, combined with Theorem 41 to ensure that $\mathtt{Ex}(+c.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}})$ is proof-like w.r.t. the set $\sharp\mathcal{S}$ of locations of conclusions of axioms in $\mathcal{S}$. □

We now consider syntax trees of formulas as incomplete proof-structures, where axioms are missing. We can extend the notion of switching to those *pre-proof-structures*, and define their ordeal (as ordeals are defined without considering axioms hyperedges[5]). This is extended to sequents and used in the next lemma: given a sequent $\vdash \Gamma$, one can consider *switchings* φ of $\vdash \Gamma$ and ordeals $(\vdash \Gamma)^\star_\varphi$. We also define $\sharp\Gamma$ as the set of locations of occurrences of atoms of Γ seen as a pre-proof-structure.

Lemma 71. *Let Ω be a basis of interpretation, $\vdash \Gamma$ a sequent of MLL+MIX, and φ a switching of $\vdash \Gamma$. Then $(\vdash \Gamma)^\star_\varphi \in (I_\Omega(\vdash \Gamma))^{\perp^{\mathrm{fin}}}$.*

Proof sketch.. The proof is done by induction:

- If $\vdash \Gamma$ has only formulas X_i or $X_i^\perp$, then there is a single switching (there are no $\parr$), and $(\vdash \Gamma)^\star_\varphi = \sum[-t.p_{X_i}(x), p_{X_i}(x)]$. Since an element $\Phi \in I_\Omega(\vdash \Gamma)$ is necessarily strongly normalisable and this implies that $\Phi \uplus (\vdash \Gamma)^\star_\varphi$ is strongly normalisable, this shows the result.
- If $\vdash \Gamma$ is $\vdash \Delta, A \parr B$, then a switching φ of $\vdash \Gamma$ is a switching $\bar{\varphi}$ of $\vdash \Delta, A, B$ extended to the additional $\parr$ connective linking A and B. It should be clear that $(\vdash \Delta, A \parr B)^\star_\varphi = (\vdash \Delta, A, B)^\star_{\bar{\varphi}}$. This shows the result, since $I_\Omega(\vdash \Delta, A \parr B) = I_\Omega(\vdash \Delta, A, B)$.
- If $\vdash \Gamma$ is $\vdash \Delta, A \otimes B$, a switching of $\vdash \Gamma$ is a switching of $\vdash \Delta, A, B$ extended to the additional $\otimes$ connective linking A and B, and $(\vdash \Delta, A \otimes B)^\star_\varphi$ can be defined from $(\vdash \Delta, A, B)^\star_\varphi$ by colouring the terms starting by p_A and p_B with a fresh colour $+u$ to obtain a constellation Θ and considering $\Theta \uplus [-u.p_A(x), -u.p_B(x), p_{A\otimes B}(x)]$. Moreover, one can show that $I_\Omega(\vdash \Delta, A \otimes B)$ is generated (in the sense of bi-orthogonal closure) by a pre-type E in which no star connects locations of A with locations of B. This shows the result since this implies that $(\vdash \Delta, A \otimes B)^\star_\varphi \in E^{\perp^{\mathrm{fin}}}$ and it is known that $E^{\perp^{\mathrm{fin}}} = E^{\perp^{\mathrm{fin}}\perp^{\mathrm{fin}}\perp^{\mathrm{fin}}}$ in general. □

Theorem 72 (Full completeness)**.** *If a constellation $\Phi \in I_\Omega(\vdash \Gamma)$ is proof-like w.r.t. $\sharp\Gamma$, there exists a MLL+MIX proof-net $\mathcal{S}$ of conclusion $\vdash \Gamma$ such that $\natural\Phi = \Phi_{\mathcal{S}}^{\mathrm{ax}}$.*

Proof. A proof-like constellation $\Phi \in I_\Omega(\vdash \Gamma)$ w.r.t. to $\sharp\Gamma$ can always be considered as the interpretation of a proof-structure with only axioms; we can then construct a proof-structure $\mathcal{S}$ by considering the union of the latter with the syntax forest of $\vdash \Gamma$. Since Φ belongs to $I_\Omega(\vdash \Gamma)$, and for all switchings φ of $\vdash \Gamma$ (equivalently, of $\mathcal{S}$) the ordeal $(\vdash \Gamma)^\star_\varphi = \mathcal{S}^\star_\varphi$ is orthogonal to Φ, Corollary 49 shows that $\mathcal{S}$ is acyclic, i.e. satisfies the correctness criterion for MLL+MIX. □

[5]We adapt the first case of Definition 47 and introduce the stars for atoms, i.e. for vertices that are not the target of an hyperedge.

E. Full completeness for MLL

Now, Corollary 49 also characterises the correctness criterion for MLL, and we wish to use that to define a fully complete model for MLL (without MIX). The issue here is that the proof of Lemma 71 uses the orthogonality $\perp^{\mathrm{fin}}$ in an essential way (in the base case), and is no longer true for general interpretations of MLL formulas. We therefore restrict our attention to those interpretations that are defined by the ordeals.

Definition 73. Given a basis of interpretation Ω, and a MLL sequent $\vdash \Gamma$. We define the *strict interpretation* $I^1_\Omega(\vdash \Gamma)$ along Ω as the 1-orthogonal of the set of ordeals for $\vdash \Gamma$. In other words, if $\mathcal{S}(\vdash \Gamma)$ denotes the set of switchings of $\vdash \Gamma$:

$$I^1_\Omega(\vdash \Gamma) = \{(\vdash \Gamma)^\star_\varphi \mid \phi \in \mathcal{S}(\vdash \Gamma)\}^{\perp^1}.$$

We then obtain the following theorem as a consequence of Corollary 49.

Theorem 74 (Full soundness and completeness for MLL)**.** *Let $\vdash \Gamma$ be a MLL sequent, and Ω a basis of interpretation. Then $\Phi \in I^1_\Omega(\vdash \Gamma)$ is proof-like w.r.t. $\sharp\Gamma$ if and only if there exists a MLL proof-net $\mathcal{S}$ of conclusion $\vdash \Gamma$ such that $\natural\Phi = \Phi_{\mathcal{S}}^{\mathrm{ax}}$.*

Proof. A proof-like constellation $\Phi \in I_\Omega(\vdash \Gamma)$ w.r.t. to $\sharp\Gamma$ can always be considered as the interpretation of a proof-structure with only axioms; we can then construct a proof-structure $\mathcal{S}$ by considering the union of the latter with the syntax forest of $\vdash \Gamma$. But Φ belongs to $I^1_\Omega(\vdash \Gamma)$ if and only if for all switchings φ of $\vdash \Gamma$ the ordeal $(\vdash \Gamma)^\star_\varphi = \mathcal{S}^\star_\varphi$ is 1-orthogonal to Φ. By Corollary 49 shows that this is equivalent to saying that $\mathcal{S}$ is a tree, i.e. it satisfies the correctness criterion for MLL. □

VI. Perspectives and future works

While we have shown here how to reconstruct the multiplicative fragment of linear logic, an interpretation of additive and exponential connectives should also be possible using stellar resolution. To interpret the additive connectives $\oplus, \&$, we need a way to exclude or force some choices in the construction of diagrams. For this purpose, Girard's second article on Transcendental Syntax [39] mentions some involved coherence relations between stars that is not completely satisfying. This idea was already properly developed in Seiller's PhD thesis [77] in the setting of interaction graphs; an improved and extended account can be found in a recent article by Nguyen and Seiller [80]. We can expect to build on the latter to interpret additive connectives.

This idea of coherence can also be useful for an extension to the MELL fragment, which will be particularly interesting since it allows for the interpretation of System F [78] and pure λ-calculus [68], [79]. In fact, Girard's first article on Transcendental Syntax [38] sketches some reconstruction of the exponentials but limited to the intuitionistic implication.

Last, but not the least, the third article on Transcendental Syntax [40] suggests to interpret the terms of first-order logic as multiplicative propositions and the equality as the linear equivalence. Thanks to the expressivity of stellar resolution, it is possible to construct objects living outside the usual theory of linear logic, which will be useful in this interpretation of first-order logic. For instance, we can mention Girard's logical constant $\supset$ [41] corresponding to a self-dual type for atomic proof-nets. This extension to first-order logic is the initial motivation behind the present work, and the authors expect to provide a formal account of these ideas in the near future. While Girard introduces yet another version of his model for this purpose to allow the use of colours in rays, Stellar resolution already captures that distinctive feature. This would provide computational content for first-order logic in the sense of the Curry-Howard correspondence, something new and fascinating that would open numerous applications.

REFERENCES

[1] G. Gentzen, "Untersuchungen über das logische schließen. i," *Mathematische zeitschrift*, vol. 39, no. 1, pp. 176–210, 1935.

[2] ——, "Untersuchungen über das logische schließen. ii," *Mathematische Zeitschrift*, vol. 39, no. 1, pp. 405–431, 1935.

[3] H. B. Curry, "Functionality in combinatory logic," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 20, no. 11, p. 584, 1934.

[4] W. A. Howard, "The formulae-as-types notion of construction," *To HB Curry: essays on combinatory logic, lambda calculus and formalism*, vol. 44, pp. 479–490, 1980.

[5] J.-Y. Girard, "Normal functors, power series and λ-calculus," *Annals of pure and applied logic*, vol. 37, no. 2, pp. 129–177, 1988.

[6] ——, "Linear logic," *Theoretical computer science*, vol. 50, no. 1, pp. 1–101, 1987.

[7] ——, "Proof-nets: the parallel syntax for proof-theory," *Lecture Notes in Pure and Applied Mathematics*, pp. 97–124, 1996.

[8] V. Danos and L. Regnier, "The structure of multiplicatives," *Archive for Mathematical logic*, vol. 28, no. 3, pp. 181–203, 1989.

[9] P. J. De Naurois and V. Mogbil, "Correctness of linear logic proof structures is nl-complete," *Theoretical Computer Science*, vol. 412, no. 20, pp. 1941–1957, 2011.

[10] S. Guerrini, "A linear algorithm for mll proof net correctness and sequentialization," *Theoretical Computer Science*, vol. 412, no. 20, pp. 1958–1978, 2011.

[11] T. Ehrhard, "A new correctness criterion for mll proof nets," in *Proceedings of the Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2014, pp. 1–10.

[12] J.-Y. Girard, "Towards a geometry of interaction," *Contemporary Mathematics*, vol. 92, no. 69-108, p. 6, 1989.

[13] J. M. E. Hyland and C.-H. Ong, "On full abstraction for pcf: I, ii, and iii," *Information and computation*, vol. 163, no. 2, 2000.

[14] S. Abramsky, R. Jagadeesan, and P. Malacaria, "Full abstraction for pcf," *Information and Computation*, vol. 163, no. 2, pp. 409–470, 2000.

[15] V. Danos and L. Regnier, "Reversible, irreversible and optimal λ-machines," *Theoretical Computer Science*, vol. 227, no. 1-2, pp. 79–97, 1999.

[16] A. Asperti and C. Laneve, "Paths, computations and labels in the λ-calculus," *Theoretical Computer Science*, vol. 142, no. 2, pp. 277–297, 1995.

[17] V. Danos and L. Regnier, "Proof-nets and the hilbert space," *London Mathematical Society Lecture Note Series*, pp. 307–328, 1995.

[18] S. Abramsky, E. Haghverdi, and P. Scott, "Geometry of interaction and linear combinatory algebras," *Mathematical Structures in Computer Science*, vol. 12, no. 5, pp. 625–665, 2002.

[19] E. Haghverdi and P. Scott, "A categorical model for the geometry of interaction," *Theoretical Computer Science*, vol. 350, no. 2-3, pp. 252–274, 2006.

[20] J.-Y. Girard, "Geometry of interaction I: interpretation of system f," in *Studies in Logic and the Foundations of Mathematics*. Elsevier, 1989, vol. 127, pp. 221–260.

[21] ——, "Geometry of interaction II: deadlock-free algorithms," in *International Conference on Computer Logic*. Springer, 1988, pp. 76–93.

[22] ——, "Geometry of interaction III: accommodating the additives," *London Mathematical Society Lecture Note Series*, pp. 329–389, 1995.

[23] ——, "Geometry of interaction IV: the feedback equation," in *Logic Colloquium*, vol. 3, 2006, pp. 76–117.

[24] ——, "Geometry of interaction V: logic in the hyperfinite factor," *Theoretical Computer Science*, vol. 412, no. 20, pp. 1860–1883, 2011.

[25] ——, "Geometry of interaction VI: a blueprint for transcendental syntax," *preprint*, 2013.

[26] C. Riba, "Strong normalization as safe interaction," in *22nd Annual IEEE Symposium on Logic in Computer Science (LICS 2007)*. IEEE, 2007, pp. 13–22.

[27] T. Seiller, "Interaction graphs: multiplicatives," *Annals of Pure and Applied Logic*, vol. 163, no. 12, pp. 1808–1837, 2012.

[28] ——, "Interaction graphs: additives," *Annals of Pure and Applied Logic*, vol. 167, no. 2, pp. 95–154, 2016.

[29] ——, "Interaction graphs: Graphings," *Annals of Pure and Applied Logic*, vol. 168, no. 2, pp. 278–320, 2017.

[30] ——, "Interaction graphs: Exponentials," *Logical Methods in Computer Science*, vol. 15, 2019.

[31] ——, "Interaction graphs: Full linear logic," in *2016 31st Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*. IEEE, 2016, pp. 1–10.

[32] P. Baillot and M. Pedicini, "Elementary complexity and geometry of interaction," *Fundamenta Informaticae*, vol. 45, no. 1-2, pp. 1–31, 2001.

[33] C. Aubert and T. Seiller, "Characterizing co-nl by a group action," *Mathematical Structures in Computer Science*, vol. 26, no. 4, pp. 606–638, 2016.

[34] ——, "Logarithmic space and permutations," *Information and Computation*, vol. 248, pp. 2–21, 2016.

[35] T. Seiller, "Interaction graphs: Non-deterministic automata," *ACM Transactions on Computational Logic (TOCL)*, vol. 19, no. 3, pp. 1–24, 2018.

[36] ——, "Probabilistic complexity classes through semantics," *arXiv preprint arXiv:2002.00009*, 2020.

[37] J.-Y. Girard, "Three lightings of logic (invited talk)," in *Computer Science Logic 2013 (CSL 2013)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2013.

[38] ——, "Transcendental syntax I: deterministic case," *Mathematical Structures in Computer Science*, vol. 27, no. 5, pp. 827–849, 2017.

[39] ——, "Transcendental syntax II: non-deterministic case," 2016.

[40] ——, "Transcendental syntax III: equality," 2016.

[41] ——, "Transcendental syntax IV: logic without systems," 2020.

[42] A. Church, "On the concept of a random sequence," *Bulletin of the American Mathematical Society*, vol. 46, no. 2, pp. 130–135, 1940.

[43] N. Immerman, *Descriptive complexity*. Springer Science & Business Media, 2012.

[44] M. Y. Vardi, "The complexity of relational query languages," in *Proceedings of the fourteenth annual ACM symposium on Theory of computing*, 1982, pp. 137–146.

[45] N. Immerman, "Relational queries computable in polynomial time," *Information and Control*, vol. 68, no. 1, pp. 86 – 104, 1986.

[46] J. A. Robinson *et al.*, "A machine-oriented logic based on the resolution principle," *Journal of the ACM*, vol. 12, no. 1, pp. 23–41, 1965.

[47] H. Wang, "Proving theorems by pattern recognition —II," *Bell system technical journal*, vol. 40, no. 1, pp. 1–41, 1961.

[48] E. Winfree, "Algorithmic self-assembly of dna," Ph.D. dissertation, Citeseer, 1998.

[49] M. J. Patitz, "An introduction to tile-based self-assembly and a survey of recent results," *Natural Computing*, vol. 13, no. 2, pp. 195–224, 2014.

[50] N. C. Seeman, "Nucleic acid junctions and lattices," *Journal of theoretical biology*, vol. 99, no. 2, pp. 237–247, 1982.

[51] D. Woods, D. Doty, C. Myhrvold, J. Hui, F. Zhou, P. Yin, and E. Winfree, "Diverse and robust molecular algorithms using reprogrammable dna self-assembly," *Nature*, vol. 567, no. 7748, pp. 366–372, 2019. [Online]. Available: https://doi.org/10.1038/s41586-019-1014-9

[52] A. Fleury and C. Retoré, "The mix rule," *Mathematical Structures in Computer Science*, vol. 4, no. 2, pp. 273–285, 1994.

[53] A. Leitsch, *The resolution calculus*. Springer Science & Business Media, 2012.

[54] R. Kowalski, "A proof procedure using connection graphs," *Journal of the ACM (JACM)*, vol. 22, no. 4, pp. 572–595, 1975.
[55] S. Sickel, "A search technique for clause interconnectivity graphs," *IEEE Transactions on Computers*, no. 8, pp. 823–835, 1976.
[56] R. Kowalski, "Predicate logic as programming language," in *IFIP congress*, vol. 74, 1974, pp. 569–544.
[57] J. Herbrand, "Recherches sur la théorie de la démonstration," Ph.D. dissertation, Université de Paris, 1930.
[58] J.-L. Lassez, M. J. Maher, and K. Marriott, "Unification revisited," in *Foundations of logic and functional programming*. Springer, 1988, pp. 67–113.
[59] F. Baader and T. Nipkow, *Term rewriting and all that*. Cambridge university press, 1999.
[60] A. Martelli and U. Montanari, "An efficient unification algorithm," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 2, pp. 258–282, 1982.
[61] J. Minker, "Overview of disjunctive logic programming," *Annals of Mathematics and Artificial Intelligence*, vol. 12, no. 1-2, pp. 1–24, 1994.
[62] M. T. Nguyen, J. Giesl, P. Schneider-Kamp, and D. De Schreye, "Termination analysis of logic programs based on dependency graphs," in *International Symposium on Logic-based Program Synthesis and Transformation*. Springer, 2007, pp. 8–22.
[63] J. Jost and R. Mulas, "Hypergraph laplace operators for chemical reaction networks," *Advances in mathematics*, vol. 351, pp. 870–896, 2019.
[64] P.-É. Meunier and D. Woods, "The non-cooperative tile assembly model is not intrinsically universal or capable of bounded turing machine simulation," in *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, 2017, pp. 328–341.
[65] A. Horn, "On sentences which are true of direct unions of algebras," *The Journal of Symbolic Logic*, vol. 16, no. 1, pp. 14–21, 1951.
[66] S.-Å. Tärnlund, "Horn clause computability," *BIT Numerical Mathematics*, vol. 17, no. 2, pp. 215–226, 1977.
[67] D. Woods, "Intrinsic universality and the computational power of self-assembly," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 373, no. 2046, p. 20140214, 2015.
[68] V. Danos, "La logique linéaire appliquée à l'étude de divers processus de normalisation (principalement du lambda-calcul)," Ph.D. dissertation, Paris 7, 1990.
[69] Y. Lafont, "From proof nets to interaction nets," *London Mathematical Society Lecture Note Series*, pp. 225–248, 1995.
[70] A. S. Murawski and C.-H. Ong, "Dominator trees and fast verification of proof nets," in *Proceedings Fifteenth Annual IEEE Symposium on Logic in Computer Science (Cat. No. 99CB36332)*. IEEE, 2000, pp. 181–191.
[71] C. Retoré, "Handsome proof-nets: perfect matchings and cographs," *Theoretical Computer Science*, vol. 294, no. 3, pp. 473–488, 2003.
[72] M. Bagnol, A. Doumane, and A. Saurin, "On the dependencies of logical rules," in *International Conference on Foundations of Software Science and Computation Structures*. Springer, 2015, pp. 436–450.
[73] A. Naibo, M. Petrolo, and T. Seiller, "On the computational meaning of axioms," in *Epistemology, Knowledge and the Impact of Interaction*. Springer, 2016, pp. 141–184.
[74] M. Acclavio and R. Maieli, "Generalized connectives for multiplicative linear logic," in *28th EACSL Annual Conference on Computer Science Logic (CSL 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.
[75] J.-Y. Girard, "Locus solum: From the rules of logic to the logic of rules," *Mathematical structures in computer science*, vol. 11, no. 3, p. 301, 2001.
[76] M. Hyland and A. Schalk, "Glueing and orthogonality for models of linear logic," *Theoretical computer science*, vol. 294, no. 1-2, pp. 183–231, 2003.
[77] T. Seiller, "Logique dans le facteur hyperfini: géometrie de l'interaction et complexité," Ph.D. dissertation, Aix-Marseille Université, 2012.
[78] J.-Y. Girard, "Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur," Ph.D. dissertation, Éditeur inconnu, 1972.
[79] L. Regnier, "Lambda-calcul et réseaux," Ph.D. dissertation, Paris 7, 1992.
[80] L. T. D. Nguyen and T. Seiller, "Coherent interaction graphs," *arXiv preprint arXiv:1904.06849*, 2019.

PROOF OF THEOREMS

Equivalence between fusion and actualisation

Definition 75 (Solved form)**.** A unification problem $P = \{x_1 \stackrel{?}{=} t_1, ..., x_n \stackrel{?}{=} t_n\}$ is in *solved form* if $\{x_1, ..., x_n\} \cap \bigcup_{j=1}^{n} \mathtt{fv}(t_j) = \emptyset$. Its associated substitution is defined by $\overrightarrow{P} = \{|x_1 \mapsto t_1, ..., x_n \mapsto t_n|\}$.

Definition 76 (Unification algorithm)**.** We use the unification algorithm from Martelli and Montanari [60] with a presentation as inference rules read from top to bottom:

$$\frac{P \cup \{t \stackrel{?}{=} t\}}{P}\ \text{clear} \qquad \frac{P \cup \{f(t_1, ..., t_n) \stackrel{?}{=} f(u_1, ..., u_n)\}}{P \cup \{t_1 \stackrel{?}{=} u_1, ..., t_n \stackrel{?}{=} u_n\}}\ \text{open}$$

$$\frac{P \cup \{t \stackrel{?}{=} x\} \text{ with } t \notin \mathtt{vars}}{P \cup \{x \stackrel{?}{=} t\}}\ \text{direct}$$

$$\frac{P \cup \{x \stackrel{?}{=} t\} \text{ with } x \in \mathtt{vars}(P) \text{ and } x \notin \mathtt{fv}(t)}{\{|x \mapsto t|\}P \cup \{x \stackrel{?}{=} t\}}\ \text{replace}$$

$$\frac{P \text{ (in solved form)}}{}\ \text{success} \qquad \frac{P \text{ (not in solved form)}}{\bot}\ \text{fail}$$

A tree constructed by these rules and ending with a success or fail rule when no other rule can be used is called an *execution* of the unification algorithm. The last step of the tree is written $\mathtt{Solution}(P)$ for a problem P. If we can apply more rules, it is called a *partial execution*.

Theorem 77 (Confluence of the unification algorithm)**.** *Two orders of successful partial executions of the unification algorithm on a problem P induce the same solution up to renaming.*

Proof. Assume two arbitrary partial orders of execution on P producing the solutions $\overrightarrow{S}$ and $\overrightarrow{S'}$. Since the solution of the unification algorithm is unique up to renaming, we have $\overrightarrow{S}$ equivalent to $\overrightarrow{S'}$ up to renaming. □

Definition 78 (Fusion)**.** Let δ be a diagram. The *reduction by fusion* is a graph contraction of D_δ. A link between x, y associated to the equation $t \stackrel{?}{=} u$ is contracted in the following way:

1) We compute $\theta := \mathtt{Solution}(t \stackrel{?}{=} u)$.
2) We delete the rays t and u and get two stars $\phi_1 := \delta(x)\backslash\{t\}$ and $\phi_2 := \delta(y)\backslash\{u\}$.
3) The two stars merge in order to form $\theta\phi_1 \cup \theta\phi_2$.

We use the notation $\delta \rightsquigarrow \delta'$ for a step of this procedure.

Lemma 79 (Simulation of fusion)**.** *For all diagram δ, there exists δ' such that $\delta \rightsquigarrow \delta'$ with a reduction of a link of equation $t \stackrel{?}{=} t'$ then there exists a partial execution*

$$\dfrac{\dfrac{\texttt{links}(\delta)}{\vdots}}{\texttt{links}(\delta') \cup \{x_1 \stackrel{?}{=} t_1, ..., x_k \stackrel{?}{=} t_k\}}$$

with $\{x_1, ..., x_k\} \cap \bigcup_{j=1}^{k} \texttt{fv}(t_j) = \emptyset$ *and* $\texttt{free}(\delta') = \{|x_1 \mapsto t_1, ..., x_k \mapsto t_k|\}\texttt{free}(\delta)$.

Proof. If the fusion succeed then the equation $\{t \stackrel{?}{=} t'\}$ linking the two stars ϕ and ϕ' has a solution. By the confluence of the unification algorithm, we can focus on the equation $t \stackrel{?}{=} t'$ to obtain $P \cup \{x_1 \stackrel{?}{=} t_1, ..., x_k \stackrel{?}{=} t_k\}$ with $\{x_1 \stackrel{?}{=} t_1, ..., x_k \stackrel{?}{=} t_k\}$ in solved form, i.e $\{x_1, ..., x_k\} \cap \bigcup_{j=1}^{k} \texttt{fv}(t_j) = \emptyset$ for $P = \texttt{links}(\delta) \backslash \{t \stackrel{?}{=} t'\}$. Then we have $\{|x_1 \mapsto t_1, ..., x_k \mapsto t_k|\}P \cup \{x_1 \stackrel{?}{=} t_1, ..., x_k \stackrel{?}{=} t_k\}$ because $\{x_1 \stackrel{?}{=} t_1, ..., x_k \stackrel{?}{=} t_k\}$ is in solved form (we can postpone the "replace" steps) with $\{x_1, ..., x_k\} \cap \bigcup_{j=1}^{k} \texttt{fv}(t_j) = \emptyset$.

When we do a fusion of ϕ and ϕ', the other rays of ϕ and ϕ' are updated with $\texttt{Solution}(t \stackrel{?}{=} t') = \{|x_1 \mapsto t_1, ..., x_k \mapsto t_k|\}$. Therefore, we have $\texttt{links}(\delta') = \{|x_1 \mapsto t_1, ..., x_k \mapsto t_k|\}P$. After the application of $\texttt{Solution}(t \stackrel{?}{=} t')$ on P, the variables $x_1, ..., x_k$ are "fixed" i.e they appear nowhere else, which prevents them to be altered during the execution of the algorithm and hence the substitutions of $\{|x_1 \mapsto t_1, ..., x_k \mapsto t_k|\}$ will appear in the last substitution applied on the free rays. We finally obtain $\texttt{free}(\delta') = \{|x_1 \mapsto t_1, ..., x_k \mapsto t_k|\}\texttt{free}(\delta)$. □

Theorem 80 (Equivalence between fusion and actualisation)**.** *For all diagram δ, we have $\delta \rightsquigarrow^{|\texttt{links}(\delta)|} (\Downarrow \delta)$.*

Proof. By induction on $|\texttt{links}(\delta)|$:

- Base case. We have 0 links, hence δ does not reduce. Since the diagrams are trees and the only tree without edge is a vertex D_δ representing the star $\Downarrow \delta$.
- Induction. We show that there exists a diagram δ' such that $\delta \rightsquigarrow \delta' \rightsquigarrow^{|\texttt{links}(\delta')|} (\Downarrow \delta)$ knowing $\delta' \rightsquigarrow^{|\texttt{links}(\delta')|} (\Downarrow \delta)$ (induction hypothesis). The simulation of fusion tells us that we can simulate a step of fusion by a prefix partial execution on δ. By the confluence of the algorithm, we can reorganise the computation of $\Downarrow D$ such that it simulates a step of fusion on δ ot obtain a certain δ' such that $\texttt{links}(\delta') := \theta\texttt{links}(\delta)$ and $\texttt{free}(\delta') := \theta\texttt{free}(\delta)$ where θ is the substitution obtained by the partial execution.

□

Confluence

Theorem 30 (Confluence)**.** *Let Φ be a constellation, and A, B be two disjoint sets of colours, i.e. $A, B \subseteq \texttt{Colours}$ and $A \cap B = \emptyset$. We have:*

$$\texttt{Ex}_B(\texttt{Ex}_A(\Phi)) = \texttt{Ex}_{A \cup B}(\Phi) = \texttt{Ex}_A(\texttt{Ex}_B(\Phi)).$$

Proof (Sketch). The proof is similar for $\texttt{Res}$ and $\Downarrow \texttt{Connect}_A(\Phi)$. We establish the first isomorphism, which is enough by symmetry. First note that the disjointness of the sets of colours implies that the set of edges in $\mathfrak{D}[\Phi; A \cup B]$ is the disjoint union of the sets of edges in $\mathfrak{D}[\Phi; A]$ and those in $\mathfrak{D}[\Phi; B]$. As a consequence, any diagram on $\mathfrak{D}[\Phi; A]$ (resp. $\mathfrak{D}[\Phi; B]$) can be thought of as a diagram on $\mathfrak{D}[\Phi; A \cup B]$.

Let $\delta : D_\delta \to \mathfrak{D}[\texttt{Ex}_A(\Phi); B]$ be a correct saturated B-diagram of $\texttt{Ex}_A(\Phi)$. For each $s \in |\texttt{Ex}_A(\Phi)|$, the star $\texttt{Ex}_A(\Phi)(s)$, which we will write ϕ_s, corresponds to a diagram $\delta_s : D_s \to \mathfrak{D}[\Phi; A]$. We can therefore build a diagram $\bar{\delta}$ over $\mathfrak{D}[\Phi; A \cup B]$ by *blowing up* δ along the diagrams δ_s. More precisely, we construct the graph $\bar{D}_\delta$ obtained by replacing each vertex s by the graph D_s; this is well-defined as each ray in $\delta(s)$ comes from a unique ray from a star $\delta_s(s')$ for some $s' \in V^{D_s}$, and therefore each edge e in D_δ of source s becomes an edge of source the unique star $s' \in V^{D_s}$. The morphism δ then extends uniquely to a morphism $\bar{\delta} : \bar{D}_\delta \to \mathfrak{D}[\Phi; A \cup B]$ whose action on the subgraphs D_s coincides with that of δ_s (as a morphism into $\mathfrak{D}[\Phi; A \cup B]$).

To check that this mapping from B-diagrams on $\texttt{Ex}_A(\Phi)$ to $(A \cup B)$-diagrams on Φ is indeed an isomorphism, one can directly define an inverse mapping. For this purpose, the essential remark is that given a $(A \cup B)$-diagram $\bar{\delta}$ on Φ, one can recover the underlying A-diagrams on $\texttt{Ex}_A(\Phi)$ as the restriction of $\bar{\delta}$ to the connected components of the graph obtained from $D_{\bar{\delta}}$ by removing the edges mapped to A-coloured edges in $\mathfrak{D}[\Phi; A \cup B]$. The underlying graph of the corresponding B-diagram on $\texttt{Ex}_A(\Phi)$ is then defined from $D_{\bar{\delta}}$ by contracting each of these connected components to a single vertex. □

Intepretation of multiplicative linear logic

Lemma 40. *Let $\mathcal{R}$ be a MLL proof-structure such that $\mathcal{R} \rightsquigarrow \mathcal{S}$. We have $\texttt{Ex}(+c.\Phi_{\mathcal{R}}^{\text{ax}} \uplus -c.\Phi_{\mathcal{R}}^{\text{cut}}) = \texttt{Ex}(+c.\Phi_{\mathcal{S}}^{\text{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\text{cut}})$.*

Proof. We remark that all rays are unique so the connexions are always non-ambiguous. Moreover, only the cuts cause the interaction/execution and they only connect identical addresses because of the share variable x. Hence, the diagrams are always exact (c.f Notation 14) and if for two constellations Φ_1, Φ_2 corresponding to proofs, we have $\mathfrak{D}[\Phi_1; -] \cong \mathfrak{D}[\Phi_2; -]$ (isomorphic as graphs with same free rays) then $\texttt{Ex}(\Phi_1) = \texttt{Ex}(\Phi_2)$. We have two cases of reduction.

- If we have a cut/axiom cut between two proofs $\vdash \pi_1 : \Gamma, A_2^\perp$ and $\vdash \pi_2 : \Delta, A_3$ with $A_1, A_2, A_3 \in \mathcal{F}_{\text{MLL}}$ where $A_1 := \mathcal{A}_e^l, A_2^\perp := \mathcal{A}_e^r$ and $e \in \text{Ax}(\pi_1)$, then we have $+c.\Phi_{\mathcal{R}}^{\text{ax}} \uplus -c.\Phi_{\mathcal{R}}^{\text{cut}} = \Gamma^\star \uplus \pi_1^\star \uplus \pi_2^\star + [-c.p_{A_2^\perp}(x), -c.p_{A_3}(x)]$ and $+c.\Phi_{\mathcal{S}}^{\text{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\text{cut}} = \Delta^\star \uplus \pi_1'^\star \uplus \pi_2'^\star$ where $\pi_1'^\star = (\pi_1 \backslash \{e\})^\star$ and $\pi_2'^\star$ is a relocalisation of $\pi_2^\star$ relatively to Γ (we update the C in $p_C(t)$), and its conclusion becomes $\mathcal{A}_e^l$. The axiom e is translated into a star $[+c.p_{A_1}(u), +c.p_{A_2^\perp}(x)]$ for some u. The cut and axiom stars will merge into $[+c.p_{A_1}(u), -c.p_{A_3}(x)]$. The ray $-c.p_{A_3}(x)$ will be connected to the $p_{A_3}(u)$ for

some u in $\pi_2^\star$ as if π_2 had $\mathcal{A}_e^l$ as conclusion. Only the cuts in $\Gamma^\star \uplus \pi_1^\star \uplus \pi_2^\star$ remain. The induction hypothesis tells us that $\mathtt{Ex}(\Gamma^\star \uplus \pi_1^\star \uplus \pi_2^\star) = \mathtt{Ex}(\Delta^\star \uplus \pi_1'^\star \uplus \pi_2'^\star)$, therefore $\mathtt{Ex}(+c.\Phi_{\mathcal{R}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{R}}^{\mathrm{cut}}) = \mathtt{Ex}(+c.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}})$.

- If we have a $\parr/\otimes$ cut between two proofs $\vdash \pi_1 : \Gamma, A \otimes B$ and $\vdash \pi_2 : \Delta, A^\perp \parr B^\perp$ with $A, B \in \mathcal{F}_{\mathrm{MLL}}$, then we have $+c.\Phi_{\mathcal{R}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{R}}^{\mathrm{cut}} = \Gamma^\star \uplus \pi_1^\star \uplus \pi_2^\star + [-c.p_{A\otimes B}(x), -c.p_{A^\perp \parr B^\perp}(x)]$ and $+c.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} = \Delta^\star \uplus \pi_1'^\star \uplus \pi_2'^\star + [-c.p_A(x), -c.p_{A^\perp}(x)] + [-c.p_B(x), -c.p_{B^\perp}(x)]$ where $\pi_1'^\star$ is π_1 with all $p_{A\otimes B}(\mathtt{1}t_1)$ replaced by $p_A(t_1)$ and $\pi_2'^\star$ is π_2 with all $p_{A\otimes B}(\mathtt{r}t_2)$ replaced by $p_B(t_2)$ (a relocalisation of atoms occurs because two conclusions disappeared). Since all formulas are unique, the cut star will be duplicated in order to connect (uniquely) to the rays of the shape $p_{A\otimes B}(t)$ and $p_{A^\perp \parr B^\perp}(t)$ (they subsume the $p_C(t)$ for $C \in \{A, B, A^\perp, B^\perp\}$). The cut necessarily connects two identical addresses (because of its shared variable x). The address of the left premises begins with $\mathtt{1}$ and by $\mathtt{r}$ therefore, the cut will connect the pairs $p_A(\mathtt{1}t)/p_{A^\perp}(\mathtt{1}t)$ and $p_B(\mathtt{r}t)/p_{B^\perp}(\mathtt{r}t)$. The cuts connect exactly the same pairs of atoms as $\mathcal{S}^\star$. Since the induction hypothesis tells us that $\mathtt{Ex}(\Gamma^\star \uplus \pi_1^\star \uplus \pi_2^\star) = \mathtt{Ex}(\Delta^\star \uplus \pi_1'^\star \uplus \pi_2'^\star)$, we have $\mathtt{Ex}(+c.\Phi_{\mathcal{R}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{R}}^{\mathrm{cut}}) = \mathtt{Ex}(+c.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}})$.

□

Theorem 48 (Stellar correctness criterion)**.** *A proof-structure $\mathcal{S}$ is a proof-net if and only if for all switching φ, we have $\natural\mathtt{Ex}(+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.+c.\mathcal{S}_\varphi^\star) = [p_{A_1}(x), ..., p_{A_n}(x)]$ where $A_1, ..., A_n$ are the conclusions of $\mathcal{S}$.*

Proof. We unfold the definition of proof-net: all correctness graphs have to be trees. It is easy to show that for an ordeal $+c.\mathcal{S}_\varphi^\star$, two rays are matchable from two stars ϕ, ϕ' if and only if their corresponding vertices are adjacent w.r.t. to the hyperedges corresponding to ϕ, ϕ'. Hence, we can say that, by design, $+c.\mathcal{S}_\varphi^\star$ reproduces the structure of the lower part of $\mathcal{S}_\varphi$, i.e $\mathcal{F} := (V^{\mathcal{S}_\varphi}, E^{\mathcal{S}_\varphi} - \mathrm{Ax}(\mathcal{S}), s')$. We can also remark that the vehicle is isomorphic to its corresponding set of axioms and so is the correctness graph w.r.t. to the connexion vehicle/ordeal.

($\Rightarrow$) The hypergraph $\mathcal{F}$ is always a forest (because it is made of the syntactic tree of $A_1, ..., A_n$) and so is $\mathfrak{D}[(-c.\mathcal{S}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star); -]$. If we connect $\mathcal{F}$ to $\mathrm{Ax}(\mathcal{S})$ (in order to retrieve $\mathcal{S}_\varphi$), we get a tree since all correctness graphs are trees. The same happens when we connect $+t.\Phi_{\mathcal{S}}^{\mathrm{ax}}$ with $-c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star$. The cuts will only cancel some conclusions. Since the diagrams are always exact (c.f Notation 14) (because of the shared variable x forcing equal adresses and because the rays coloured by t exactly match by design) and $\mathfrak{D}[+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star; -]$ is connected, we obtain only one unique star containing its free rays. Since it is also acyclic, each conclusion appears only one time. Therefore, $\mathtt{Ex}(+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star) = [p_{A_1}(x), ..., p_{A_n}(x)]$.

($\Leftarrow$) Following the previous context, if $\mathtt{Ex}(+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star) = [p_{A_1}(x), ..., p_{A_n}(x)]$, $\mathfrak{D}[+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star; -]$ necessarily is connected, otherwise we would get several stars. It is also must be acyclic, otherwise, $+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star$ would not be strongly normalising because the diagrams are exact and hence, all cycle yield infinitely many correct saturated diagrams. □

Corollary 49 (Corollary of Theorem 48)**.** *All correctness graphs of a proof-structure $\mathcal{S}$ are:*

- *acyclic if and only if $\mathfrak{D}[+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star; -]$ is acyclic, hence $+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star$ is strongly normalising and*
- *trees if and only if $\mathfrak{D}[+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star; -]$ is a tree, hence $+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star$ normalises into a single star.*

Proof. The proof of Theorem 48 states that a correctness graph has exactly the same structure as the dependency graph of its translation.

- if $\mathfrak{D}[+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star; -]$ is acyclic, stars cannot be repeated, hence $+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star$ is trivially strongly normalising. If it is strongly normalising, the dependency graph must be acyclic because, in the context of proofs, a cycle always yield infinitely many correct saturated diagrams and if it is the case, the constellation cannot be strongly normalising.
- if $\mathfrak{D}[+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star; -]$ is also connected, since the diagrams are exact and that there is no branching possible, there is at most one diagram, hence a single star in the normal form. If $+t.\Phi_{\mathcal{S}}^{\mathrm{ax}} \uplus -c.\Phi_{\mathcal{S}}^{\mathrm{cut}} \uplus +c.\mathcal{S}_\varphi^\star$ normalises into a single star, its dependency graph must be both connected and acyclic, otherwise we would end up with several stars or infinitely many correct saturated diagrams.

□

Simulation of the abstract tile assembly model

Theorem 27 (Simulation of finite aTAM)**.** *Let T be a set of tiles. The set of non-empty finite assemblies constructible from T at temperature τ is bijective to $\mathtt{Connect}(T^\star \uplus \Phi_{env}^\tau)$.*

Proof. Assume we have n adjacent tiles in a. For each tile in a, the sum of its connexions with its adjacent tiles is at least τ. We now try to reproduce an isomorphic diagram with $T^\star$. Direct connexions without using Φ_{env}^τ is forbidden because of the symbols $\circ$ and $\bullet$.

Assume the absence of these symbols. By design, it is obvious that the side-matchability of tiles corresponds to the ray-matchability of star. Moreover, the colours v and h force the connexions to be on the same axis in order to follow the geometric restriction of tiling in a plane. The tiles are designed so that a plugging increment a coordinate x or y depending on the position/axis of the side. This purpose of this feature is to simulate a shifting of tile on a plane so that two tiles cannot connect on two sides at the same time.

Now, assume the have the symbols $\circ$ and $\bullet$ and that we have to use the constellation Φ_{env}^τ as an intermediate for the connexion of two tile sides. We show that dynamics of tiling

construction corresponds to the dynamics of the construction of correct saturated diagram. We consider a tile $t_i \in \mathtt{dom}(a)$. We starts with $t_i^\bigstar$. If $n = 1$ then the corresponce with the singleton assembly is trivial. If $n > 1$, t_i can be connected to k other tiles in $\mathtt{dom}(a)$. They can only be connected through Φ^τ_{env} by their connectable sides. Their glue type and strength for the connected sides have to match because of the shared variables for opposite sides in Φ^τ_{env}. All other sides of Φ^τ_{env} will be plugged by the unary stars used as fillers. By using principles of logic programming, the diagram can only be correct and saturated if the sum of connected sides of t_i is greater or equal to τ (note that the filled unused sides add 0 to the sum). The stars coming from logic programs are common logic programs. Their correctness can be simply proved by induction as stated in Theorem 82 and Theorem 81.

Since all $t_i \in \mathtt{dom}(a)$ satisfy the above property, the two operations have the same dynamics. Moreover, each tile corresponds exactly to a star and each of its sides corresponds to a ray and we have a structural isomorphism between tiles and their translation. It follows that we have a bijection between the set of non-empty finite assemblies constructible from T at temperature τ and $\mathtt{Connect}(T^\bigstar \uplus \Phi^\tau_{env})$. □

Lemma 79 (Recursion lemma)**.** *Let δ be a correct diagram such that $\delta(D_\delta)$ is a cycle in $\mathfrak{D}[\Phi; -]$. We can duplicate n times δ to form a greater correct diagram δ_+ such that $\delta(D_{\delta_+})$ is a cycle in $\mathfrak{D}[\Phi; -]$. If $\theta = \mathtt{Solution}(\mathcal{P}(\delta))$, then $\Downarrow \delta_+ = \theta^n\theta\mathtt{free}(\delta)$.*

Proof. By induction on n. If $n = 0$, we have $\delta_+ = \delta$ and $\Downarrow \delta_+ = \Downarrow \delta = \theta\mathtt{free}(\delta)$. For the inductive case, by the induction hypothesis, $\Downarrow \delta_+ = \theta^n\theta\mathtt{free}(\delta)$ for n duplications of δ. We would like to show that we add a duplication, we can construct a diagram δ'_+ such that $\Downarrow \delta'_+ = \theta^{n+1}\theta\mathtt{free}(\delta)$. By the confluence of the actualisation, we can start from δ_+. Since $\delta(D_\delta)$ is a cycle in $\mathfrak{D}[\Phi; -]$ and that $\delta(D_{\delta_+})$ is also a cycle in $\mathfrak{D}[\Phi; -]$, the diagram δ_+ can be extended δ. This give rises to a bigger diagram δ'_+. In $\mathcal{P}(\delta'_+)$ we can focus on the equations of δ. As in the proof of simulation of fusion (Lemma 79), we have $n+2$ times the equations of δ (because we have δ and its $n+1$ duplications), therefore, it corresponds to $n + 2$ applications of $\theta = \mathtt{Solution}(\mathcal{P}(\delta))$. Therefore, $\Downarrow \delta'_+ = \theta^{n+1}\theta\mathtt{free}(\delta)$. □

Lemma 80 (Simple recursion lemma)**.** *Let $\phi = [r, r']$ be a star such that $r \bowtie r'$. We can construct a diagram $\delta : D_\delta \to \mathfrak{D}[\Phi; -]$ such that for all x, $\delta(x) = \phi$. Moreover, if we have $\theta = \mathtt{Solution}(\mathcal{P}(\delta))$, then $\Downarrow \delta = \theta^n\phi$.*

Proof. This is a special case of Lemma 79. When connecting the star θ to itself n times to form a diagram δ, we have $\mathtt{free}(\delta) = \phi$ (the borders of δ). This diagram corresponds to a loop in $\mathfrak{D}[\Phi; -]$. If we have $\theta = \mathtt{Solution}(\mathcal{P}(\delta'))$ where δ is the connexion of two occurrences of ϕ, we have $\Downarrow \delta = \theta^n\phi$. □

Theorem 81 (Correctness of addition)**.** *Let $\Phi^+_{\mathbf{N}}$ be the constellation*

$$[+add(\underline{0}, y, y)] + [+add(s(x), y, s(z)), -add(x, y, z)].$$

We have $\mathtt{Res}(\Phi^+_{\mathbf{N}} + [-add(\underline{n}, \underline{m}, r), r]) = [\underline{n+m}]$.

Proof. By induction on n. If $n = 0$, then the query $[-add(\underline{n}, \underline{m}, r), r]$ only matches with $[+add(\underline{0}, y, y)]$, forming a saturated diagram actualising into $\{r \mapsto \underline{m}\}[r] = [\underline{m}] = [\underline{n+m}]$.

We now consider the case where we have $n = n' + 1$ such that $\mathtt{Res}(\Phi^+_{\mathbf{N}} + [-add(\underline{n'}, \underline{m}, r), r]) = [\underline{n'+m}]$ and we would like to show $\mathtt{Res}(\Phi^+_{\mathbf{N}} + [-add(\underline{n'+1}, \underline{m}, r), r]) = [\underline{(n'+1)+m}]$. The diagram for $n' + m$ in the induction hypothesis must come from a unique correct and saturated diagram. By analysis of the possibilities of matching, this diagram necessarily have n' repetitions of the star $[+add(s(x), y, s(z)), -add(x, y, z)]$ connected in a linear diagram. By the confluence of the unification algorithm behind the actualisation, we can focus on this subdiagram. By the simple recursion lemma (Lemma 80), this linear diagram actualises into $\{x \mapsto s^{n'}(x), y \mapsto y, z \mapsto s^{n'}(z)\}[+add(s(x), y, s(z)), -add(x, y, z)] = [+add(s^{n'}(x), y, s^{n'}(z)), -add(x, y, z)]$.

We can add an occurrence of this star in order to construct a new saturated diagram. We would like to actualises the linear diagram composed of $[+add(s^{n'}(x), y, s^{n'}(z)), -add(x, y, z)]$ and $[+add(s(x), y, s(z)), -add(x, y, z)]$. It actualises into $[+add(s^{n'+1}(x), y, s^{n'+1}(z)), -add(x, y, z)]$. When connected to $[+add(\underline{0}, y, y)]$, it actualises into $\{x \mapsto \underline{0}, y \mapsto z\}[+add(s^{n'+1}(x), y, s^{n'+1}(z))] = [+add(\underline{n'+1}, z, s^{n'+1}(z))]$. We finally connect it to the query $[-add(\underline{n}, \underline{m}, r), r]$, forming a diagram actualising into $\{z \mapsto \underline{m}, r \mapsto s^{n'+1}(z)\}[r] = \{r \mapsto \underline{(n'+1)+m}\}[r] = [\underline{(n'+1)+m}] = [\underline{n+m}]$. A simple matchability analysis shows that no other correct saturated diagram is possible. □

Theorem 82 (Correctness of greater or equal)**.** *Let $\Phi^{\geq}_{\mathbf{N}}$ be the constellation*

$$[+geq(\underline{0}, \underline{0}, \underline{1})] + [+geq(s(x), s(y), r), -geq(x, y, r)]$$
$$+[+geq(s(x), \underline{0}, \underline{1})] + [+geq(\underline{0}, s(y), \underline{0})].$$

If $n \geq m$ then

$$\mathtt{Res}(\Phi^{\geq}_{\mathbf{N}} + [-geq(\underline{n}, \underline{m}, r), r]) = [\underline{1}],$$

otherwise

$$\mathtt{Res}(\Phi^{\geq}_{\mathbf{N}} + [-geq(\underline{n}, \underline{m}, r), r]) = [\underline{0}].$$

The converse implications of the two cases are also true.

Proof. By induction on n. If $n = 0$, then we must have $m = 0$. The query $[-geq(\underline{0}, \underline{0}, r), r]$ only matches with $[+geq(\underline{0}, \underline{0}, \underline{1})]$. They form a saturated diagram actualising into $\{r \mapsto \underline{1}\}[r] = [\underline{1}]$. We now consider the inductive case $n = n' + 1$ such that $n' \geq m$ if and only if $\mathtt{Res}(\Phi^{\geq}_{\mathbf{N}} + [-geq(\underline{n'}, \underline{m}, r), r]) = [\underline{1}]$.

If $m = 0$ then, the query form a saturated diagram with $[+geq(s(x), \underline{0}, \underline{1})]$ which actualises into $\underline{1}$. Otherwise, it is connected to the star $[+geq(s(x), s(y), r), -geq(x, y, r)]$ and we complete with the induction hypothesis. We use the same reasoning for the case where $n \geq m$ is false and for the converse implications, similarly to the proof of Theorem 81. $\square$