Towards a Combinatorial Representation of First-Order Bicategories

Leo Lobski
University College London, UK
leo.lobski.21@ucl.ac.uk

Ralph Sarkis
University College London, UK
au@ralph.sarkis.cc

Paul Wilson
Hellas AI, UK
paul@statusfailed.com

Fabio Zanasi
University College London, UK
f.zanasi@ucl.ac.uk

We present work in progress towards a combinatorial characterisation of first-order bicategories, as introduced in [1]. These structures provide a categorical setting for studying first-order logic through a string-diagrammatic syntax. Unlike string diagrams for monoidal categories, those of first-order bicategories do not admit a straightforward combinatorial representation in terms of hypergraphs, due to two interacting monoidal structures. Developing such a representation is a necessary step towards a computationally feasible implementation of *string diagram rewriting*: structurally equivalent string diagrams can then be interpreted as the same hypergraph, substantially simplifying matching procedures [2].

Background. In [5], the authors introduce a string diagrammatic syntax for the *regular* fragment of first-order logic: i.e. the allowed connectives are restricted to $\exists$, $\wedge$, and $\top$. They provide a combinatorial representation based on a notion of a *hypergraph with interfaces*—this allows applying the results of string diagram rewriting [2–4] to the regular fragment of first-order logic. Crucially, the structural equations of the theory are absorbed by the equality of hypergraphs, while existence of a hypergraph morphism corresponds to entailment of formulas.

In [1], the authors note that the diagrammatic syntax dual to the regular fragment—corresponding to the formulas with the connectives $\forall$, $\vee$, and $\bot$—interacts in a natural, principled manner with the original syntax, giving rise to a diagrammatic syntax for full first-order logic (FOL). While the diagrammatic terms visually resemble string diagrams, the presence of two interacting monoidal structures prevents a direct interpretation of this syntax as hypergraphs. Such terms, with certain equations and inequations, are proved sound and complete with respect to first-order logic by characterising the syntactic category as the free *first-order bicategory* (FOB) over a fixed relational signature. We depict this by the bidirectional arrows on the right of (1) below.

Objectives. In this work, we introduce a variant of hypergraphs (with additional labelling) and a translation from the terms of the free first-order bicategory to said hypergraphs. We summarise this characterisation with four desiderata:

1. The introduced notion of hypergraph should form a bicategory equivalent to the free FOB.
2. The structural equalities of FOBs should be absorbed by equalities between hypergraphs.
3. When restricted to the regular fragment, we should recover precisely the hypergraphs of [5].
4. The combinatorial structure should decompose into hypergraphs for string diagrams of symmetric monoidal categories, providing the conditions under which the hypergraph rewriting theory of [2–

Submitted to:
DIALOCO 2026

4] can be applied; this also means the data structures for usual hypergraphs (e.g. [8]) can be used in an implementation.

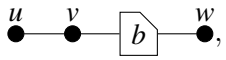
We suggest that a combinatorial characterisation satisfying all desiderata is given by "flattening" a tree of hypergraphs: the hypergraphs at even depth describe the regular fragment, while those at odd depth correspond to the dual fragment, mimicking the string diagrams with alternating white and black backgrounds in [1]. In our definition of such trees, which we call *branching hypergraphs*, we follow the work on *nested application conditions* by Rensink [6] and Rensink & Corradini [7], where FOL formulas over a signature of binary relations are encoded as trees of graphs.

Contribution. In following diagram, we summarise the relationship between the free first-order bicategory as presented by the diagrams of [1], FOL formulas and the hypergraph structures we are about to introduce—*branching hypergraphs* and *flattened branching hypergraphs*:

$$\begin{array}{ccc} \mathbf{Byp}_\Sigma & \longrightarrow & \mathbf{FOL}_\Sigma \\ \uparrow\downarrow & & \uparrow\downarrow \\ \mathbf{Fyp}_\Sigma & \longleftarrow & \mathbf{FOB}_\Sigma \end{array}. \tag{1}$$

Throughout the discussion, Σ is a fixed relational signature. By a *hypergraph* (V,H,s,t) we mean a pair of sets (V,H) called the *vertices* and the *hyperedges* together with functions $s,t : H \to V^*$ assigning each hyperedge h to its *source* $s(h)$ and *target* $t(h)$ (both of which are finite lists of vertices). It is additionally assumed that there is a labelling function $L : H \to \Sigma$ assigning to each hyperedge $h \in H$ a relation symbol from Σ in a way that is consistent with with the sources and the targets: the arity of $L(h)$ is equal to $|s(h)|$, and the coarity of $L(h)$ is equal to $|t(h)|$.

We start by considering a slightly amended version of a hypergraph, namely a *hypergraph with equality*. A hypergraph with equality is a hypergraph which, in addition to the hyperedges, comes with a reflexive and symmetric relation on the vertices called *equality edges*. We refer to a hypergraph with equality as a *hyp*. We draw an example of a very simple hyp below:



, (2)

whose vertices are $\{u,v,w\}$, the single hyperedge has source v, target w and the label $b \in \Sigma$, whose arity and coarity are thus both 1. The hyperedge is drawn as a box with its label inside. The single equality edge (u,v) is drawn as a line.

To each hyp $\mathscr{H}$, we associate a FOL formula $\varphi_{\mathscr{H}}$ in a straightforward way: the vertices are treated as variables bound by an existential quantifier, the equality edges are treated as equality symbols, and each hyperedge is treated as the relation symbol assigned to it. In symbols, denoting the equality edge relation by E, we define:

$$\varphi_E \coloneqq \bigwedge_{(u,v)\in E} u = v, \qquad \varphi_H \coloneqq \bigwedge_{h\in H} L(h)\,(s(h),t(h))\,, \qquad \varphi_{\mathscr{H}} \coloneqq \exists\{v : v \in V\}.\varphi_E \wedge \varphi_H.$$

For example, the formula assigned to the hyp (2) is $\exists u,v,w.(u = v) \wedge b(v,w)$.

We treat $\{-1\} \cup \mathbb{N}^*$ as the set of *branches* with the prefix extension order and -1 as bottom: $b \leq c$ if and only if $b = -1$ or $\exists a \in \mathbb{N}^*, ba = c$. A *tree* is a finite subset of branches that is downwards closed

under this order, its root branch is always -1. For example, if the tree contains the branch 1211, it can be treated as a path in the tree 'addressing' a node at depth 4: see Figure 1c.

A *branching hypergraph* is an assignment of a hyp to each branch of a tree T in such a way that the vertices of a hyp assigned to a branch b, denoted $\mathscr{B}(b)$, are a subset of the vertices of the hyp assigned to c whenever c extends b, and the hyp assigned to -1 has no edges nor hyperedges. We refer to a branching hypergraph as a *byp*. Each byp $\mathscr{B}$ induces a FOL formula recursively defined below (following a similar definition in [6]):

$$\varphi_{\mathscr{B}} \coloneqq \varphi_{\mathscr{B}(\varepsilon)} \wedge \bigwedge_{n \in T \cap \mathbb{N}} \neg \varphi_{\mathscr{B}|_n},$$

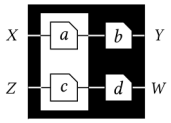
where $\mathscr{B}|_n$ is the byp defined by restricting (and shifting) to the branches extending n and whose root is assigned the vertices of $\mathscr{B}(\varepsilon)$. We depict the translation from byps to FOL formulas as the top horizontal arrow in (1). We note that the the set assigned to the root -1 contains precisely the *free variables* of $\varphi_{\mathscr{B}}$. Given a function θ from this set to an ordinary hypergraph X (without equality or branching), there is a purely combinatorial notion of *satisfaction* of $\mathscr{B}$, which does not make any reference to FOL formulas: we refer to θ as an *assignment* and write $\theta \vDash \mathscr{B}$ when θ satisfies $\mathscr{B}$. The following key lemma connects this to first-order satisfaction: it allows proving statements about FOL formulas using byps.

Lemma 1. *For every byp $\mathscr{B}$ and assignment θ, we have $\theta \vDash \mathscr{B}$ if and only if θ satsifies $\varphi_{\mathscr{B}}$ in the usual sense of FOL.*

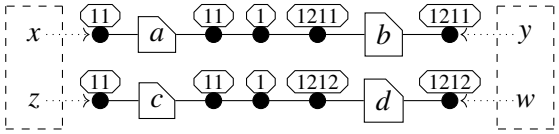
In order to interpret the diagrammatic terms of free first-order bicategories as hypergraphs, it is useful to consider a flattened version of a byp, called a *flattened branching hypergraph* or *fyp*. A fyp is like a hyp, but each vertex is additionally labelled with a branch obeying the rule that two vertices must be comparable when they are connected by an equality edge or a hyperedge. The branch of each equality edge and hyperedge can then be deduced as the longest branch among its vertices. The branch of hyperedges with zero arity and coarity has to be recorded explicitly. Fyps carry the same information as byps: every fyp induces a byp, and for every byp there is a fyp such that the induced byp results in the same (up to FOL equivalence) formula as the original byp. We depict this situation by the bidirectional arrows on the left of (1). We are interested in both structures as byps admit a straightforward translation to FOL formulas and suggest a natural notion of a morphism, while it is more convenient to translate the string diagrammatic terms of the free FOB to fyps (the bottom horizontal map (1) briefly described below).

The translation $\mathbf{FOB}_\Sigma \to \mathbf{Fyp}_\Sigma$ in (1) maps each diagrammatic term of the free FOB to a fyp with interfaces: the interfaces are assigned the branch -1 (so that they are treated as the free variables), while the non-interface vertices are assigned a non-root branch in $\mathbb{N}^*$ (so that they are treated as bound variables).

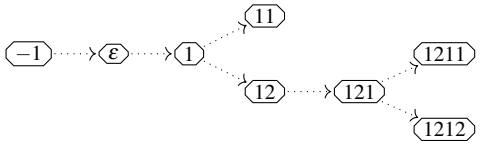
We conclude by giving an example of this translation (Figure 1). Consider the term (1a) in $\mathbf{FOB}_\Sigma$ presented as a string diagram, where a, b, c and d are arbitrary terms. Its translation to a fyp is drawn in 1b. In order to obtain the first-order formula encoded by the fyp, we translate it to the corresponding byp. The tree of the byp is shown in 1c. The root -1 is simply indexing the free variables $\{x, y, z, w\}$ (i.e. the interfaces of the fyp). The hyps indexed by other branches have as vertices all the vertices of the fyp 1b labelled by a label that is in the downward closure of the branch, while the equality edges and the hyperedges are those which have precisely the label of the branch. The graph of the function mapping the interfaces to the vertices is treated as equality edges. For example, the hyp indexed by the node 1211



(a) A term in $\mathbf{FOB}_\Sigma$ from [1].



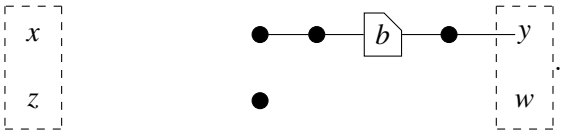
(b) Translation of the term 1a to a fyp in $\mathbf{Fyp}_\Sigma$.



(c) The underlying tree of the byp obtained from the fyp 1b.

Figure 1: Equivalent representations of the same first-order formula.

is given by



The FOL formula induced by the byp is given by

$$\neg\exists i, j.\neg\left(a(x,i)\wedge c(z,j)\right)\wedge\neg\neg\left(\neg b(i,y)\wedge\neg d(j,w)\right),$$

which simplifies to

$$\forall i, j.\left(a(x,i)\wedge c(z,j)\right)\vee b(i,y)\vee d(j,w).$$

This is the same formula that is obtained by directly translating the string diagrammatic term 1a.

Future work. The proposed hypergraph structures fulfills the desiderata 2, 3 and 4, with the caveat that we now have hypergraphs with equality rather than plain hypergraphs. Indeed, the structural equations of first-order bicategories are absorbed by fyps, restricting to the regular fragment (i.e. fyps where all labels are ε) reduces to ordinary hypergraphs, and the translation to byps decomposes each fyp into hypergraphs. Crucially, what is missing to establish the equivalence of desideratum 1 are the *inequalities*, corresponding to first-order entailment. While we have some partial results towards characterising the inequalities as morphisms of byps, this is significantly more complex than in the regular case [5] (expectedly so, as first-order entailment is undecidable). An additional challenge is posed by the equality edges: while we found them necessary to encode the nesting of the fragments—the two different monoidal structures and compositions—we now have to introduce additional *rewrites* that manipulate the equality edges and the branching structure to show e.g. unitality of composition. We are currently studying a more efficient way to encode both the equality edges and their rewrites.

References

[1] Filippo Bonchi, Alessandro Di Giorgio, Nathan Haydon & Pawel Sobocinski (2024): *Diagrammatic Algebra of First Order Logic*. In: *Proceedings of the 39th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '24, Association for Computing Machinery, New York, NY, USA, doi:10.1145/3661814.3662078. Available at `https://doi.org/10.1145/3661814.3662078`.

[2] Filippo Bonchi, Fabio Gadducci, Aleks Kissinger, Pawel Sobocinski & Fabio Zanasi (2022): *String diagram rewrite theory. I: Rewriting with Frobenius structure*. *J. ACM* 69(2), p. 58, doi:10.1145/3502719. Available at `discovery.ucl.ac.uk/id/eprint/10147745/`. Id/No 14.

[3] Filippo Bonchi, Fabio Gadducci, Aleks Kissinger, Pawel Sobocinski & Fabio Zanasi (2022): *String diagram rewrite theory. II: Rewriting with symmetric monoidal structure*. *Math. Struct. Comput. Sci.* 32(4), pp. 511–541, doi:10.1017/S0960129522000317.

[4] Filippo Bonchi, Fabio Gadducci, Aleks Kissinger, Paweł Sobociński & Fabio Zanasi (2022): *String diagram rewrite theory. III: Confluence with and without Frobenius*. *Math. Struct. Comput. Sci.* 32(7), pp. 829–869, doi:10.1017/S0960129522000123.

[5] Filippo Bonchi, Jens Seeber & Pawel Sobocinski (2018): *Graphical Conjunctive Queries*. In Dan R. Ghica & Achim Jung, editors: *27th EACSL Annual Conference on Computer Science Logic, CSL 2018, Birmingham, UK, September 4-7, 2018*, LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 13:1–13:23, doi:10.4230/LIPICS.CSL.2018.13. Available at `https://doi.org/10.4230/LIPIcs.CSL.2018.13`.

[6] Arend Rensink (2004): *Representing First-Order Logic Using Graphs*. In Hartmut Ehrig, Gregor Engels, Francesco Parisi-Presicce & Grzegorz Rozenberg, editors: *Graph Transformations, Second International Conference, ICGT 2004, Rome, Italy, September 28 - October 2, 2004, Proceedings*, Lecture Notes in Computer Science, Springer, pp. 319–335, doi:10.1007/978-3-540-30203-2_23. Available at `https://doi.org/10.1007/978-3-540-30203-2_23`.

[7] Arend Rensink & Andrea Corradini (2024): *On Categories of Nested Conditions*. In Nils Jansen, Sebastian Junges, Benjamin Lucien Kaminski, Christoph Matheja, Thomas Noll, Tim Quatmann, Mariëlle Stoelinga & Matthias Volk, editors: *Principles of Verification: Cycling the Probabilistic Landscape - Essays Dedicated to Joost-Pieter Katoen on the Occasion of His 60th Birthday, Part I*, Lecture Notes in Computer Science, Springer, pp. 393–418, doi:10.1007/978-3-031-75783-9_16. Available at `https://doi.org/10.1007/978-3-031-75783-9_16`.

[8] Paul W. Wilson & Fabio Zanasi (2023): *Data-Parallel Algorithms for String Diagrams*. *CoRR* abs/2305.01041, doi:10.48550/ARXIV.2305.01041. arXiv:2305.01041.