Logic Beyond Formulas: Designing Proof Systems on Graphs

Matteo Acclavio[1]

Università Roma Tre, Roma, Italy

Keeping track of relations between objects or events is essential in modeling processes and in verifying their security and privacy properties. For this purpose, relations are encoded by means of formulas in order to use proof theoretical results to design verification tools.

The "happens before" relation [12], providing a partial order between events to express when an event precede another, is crucial when studying distributed systems. Its restriction to series-parallel orders have received a special attention [16, 9, 5], giving rise to a family of non-commutative logics, including pomset logic [14] and BV [10].

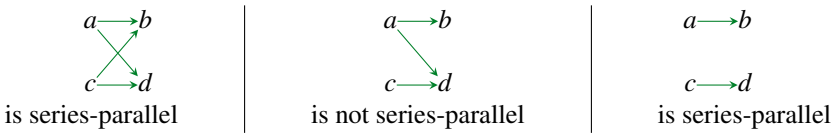
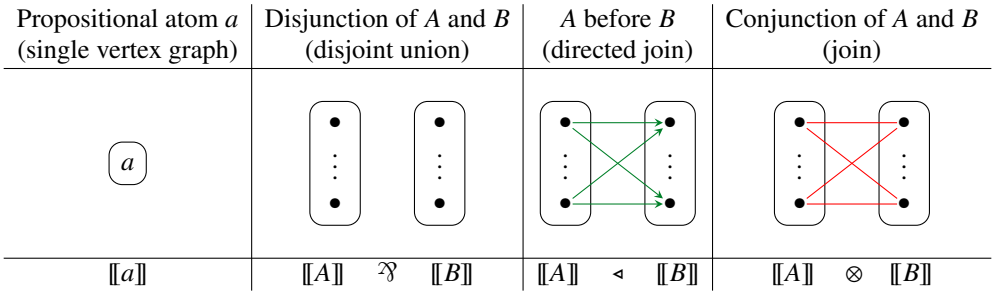


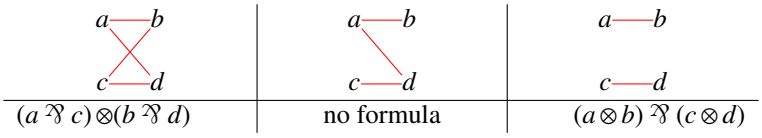
Figure 1: Three partial orders represented by their Hessse diagrams.

However, relations admitting no series-parallel decomposition [7] cannot be directly treated by the same proof theoretical methods since they require the use of encodings, which create a gap between meaning (semantics) and formal representation (syntax). In fact, the natural correspondence between graphs and formulas provided by the operations below [14, 10] fails as soon as simple topological conditions are not met in the graph representing a relation.



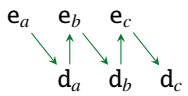
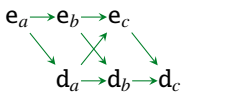
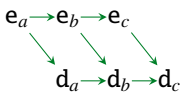
(1)

By means of example, consider four processes a, b, c and d where communication between some processes is forbidden because of certain conflicts of interest [6]. Thus, the following pairs cannot communicate: a and b, a and d, and c and d, as shown in the graph below in the center where the edges represent the impossibility of communication between processes.



(2)

Another example is given by the causality patterns for n-queues, where n is the bound on the number of elements that can be enqueued. These patterns can be represented by the graphs below, where nodes labelled by e_x and d_x respectively represent the enqueuing and dequeuing of the element x (we only represent the first three elements a, b, and c inserted into the queue), and edges represent the "happens before" relation. Among these graphs, only Q_1 and Q_2 are series-parallel graphs and can be directly encoded as formulas. In fact, the graph Q_3, and more in general the causality patterns for n-queues with $n > 2$, cannot.

a 1-queue Q_1	a 2-queue Q_2	a 3-queue Q_3
		
$\mathsf{e}_a \triangleleft \mathsf{d}_a \triangleleft \mathsf{e}_b \triangleleft \mathsf{d}_b \triangleleft \mathsf{e}_c \triangleleft \mathsf{d}_c$	$\mathsf{e}_a \triangleleft (\mathsf{d}_a \parr \mathsf{e}_b) \triangleleft (\mathsf{d}_b \parr \mathsf{e}_c) \triangleleft \mathsf{d}_c$	no formula

(3)

This contribution, based on joint works with Straßburger, Horne and Mauw [3, 2, 1], is an introduction on the proof theory of proof systems operating on graphs instead of formulas. This line of work aims at defining proof theoretical tools able to directly handle non series-parallel relations as primitive objects of a logic.

In order to design such systems, we use results on graph modular decomposition [13] allowing us to associate abstract syntax trees to graphs, and therefore to generalize the notions of connective and subformula which are fundamental to express desirable proof theoretical notions. After defining a (linear) implication $\multimap$, we define proof systems meeting certain basic desiderata such as the derivability of the general identity ($G \multimap G$ is provable for any graph G), the transitivity of implication (if $G \multimap H$ and $H \multimap K$ are provable, then $G \multimap K$ also is), and analyticity (if G is provable, then G admits a proof containing only its "subformulas" of G) . To this end, we use the open deduction [11] proof formalism (see Figure 2 for an example) based on deep inference [4] since, as observed for the non-commutative logic BV [10, 15], it is not possible to define an analytic sequent calculus for these logics.

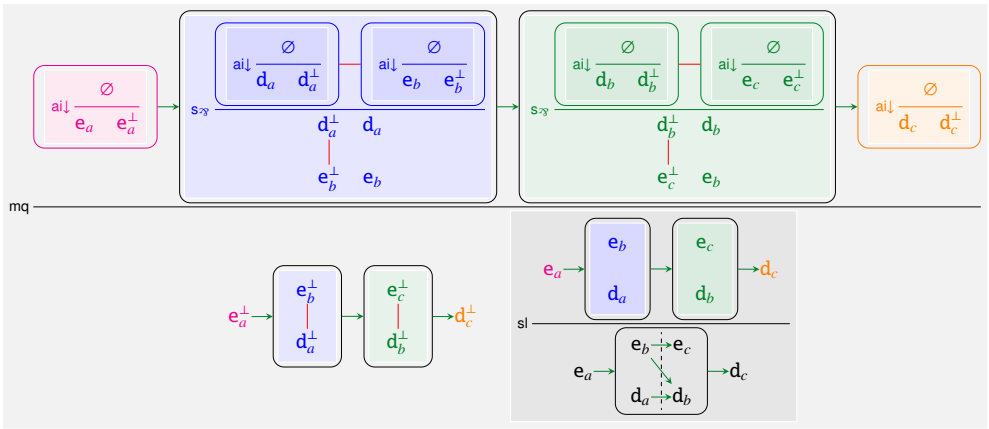


Figure 2: A proof of the graph $\mathsf{Q}_3 \multimap \mathsf{Q}_2$ in the system $\mathsf{GV}^{\mathsf{sl}}$ serving as proof that 3-queues can simulate behaviours of 2-queues. The rule sl slices a directed graph into a "before" and an "after" part by introducing additional directed edges. The rule mq merges the modules of two copies of the same directed graph.

We present the system GS, handling undirected graphs as the ones in Equation 2, and the systems GV and $\mathsf{GV}^{\mathsf{sl}}$ handling graphs with both directed and undirected edges. The system GS defines a conser-

vative extension of the *multiplicative linear logic with mix* [8], while the systems GV and GV$^{\mathsf{sl}}$ defines conservative extensions of both the graphical logic defined by GS and the non-commutative logic BV. We present the technique developed to prove these results, including the challenges we encountered in proving the analogous of cut-elimination for deep inference systems in the graphical setting. We conclude by recalling related results in proof theory and concurrency theory, their possible applications to verification thanks to their more expressive power, and giving an overview on the the ongoing researches on the topic.

References

[1] Matteo Acclavio, Ross Horne, Sjouke Mauw, and Lutz Straßburger. A graphical proof theory of logical time. In *FSCD 2022*, volume 228. LIPIcs, 2022.

[2] Matteo Acclavio, Ross Horne, and Lutz Straßburger. An analytic propositional proof system on graphs. 2020.

[3] Matteo Acclavio, Ross Horne, and Lutz Straßburger. Logic beyond formulas: A proof system on graphs. LICS '20, page 38–52, New York, NY, USA, 2020. Association for Computing Machinery.

[4] Andrea Aler Tubella and Lutz Straßburger. Introduction to deep inference. Lecture, August 2019.

[5] Denis Bechet, Philippe de Groote, and Christian Retoré. A complete axiomatisation for the inclusion of series-parallel partial orders. In *International Conference on Rewriting Techniques and Applications*, pages 230–240. Springer, 1997.

[6] David FC Brewer and Michael J Nash. The Chinese Wall security policy. In *IEEE symposium on security and privacy*, volume 1989, page 206. Oakland, 1989.

[7] R.J Duffin. Topology of series-parallel networks. *Journal of Mathematical Analysis and Applications*, 10(2):303 – 318, 1965.

[8] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50:1–102, 1987.

[9] Jay L Gischer. The equational theory of pomsets. *Theoretical Computer Science*, 61(2-3):199–224, 1988.

[10] Alessio Guglielmi. A system of interaction and structure. *ACM Trans. Comput. Logic*, 8(1):1–es, jan 2007.

[11] Alessio Guglielmi, Tom Gundersen, and Michel Parigot. A proof calculus which reduces syntactic bureaucracy. In Christopher Lynch, editor, *Proceedings of the 21st International Conference on Rewriting Techniques and Applications*, volume 6 of *LIPIcs*, pages 135–150, Dagstuhl, Germany, 2010. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[12] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM*, 21:558–565, 1978.

[13] László Lovász and Michael D Plummer. *Matching theory*, volume 367. American Mathematical Soc., 2009.

[14] Christian Retoré. Pomset logic: A non-commutative extension of classical linear logic. In Philippe de Groote and J. Roger Hindley, editors, *Typed Lambda Calculi and Applications*, pages 300–318, Berlin, Heidelberg, 1997. Springer.

[15] Alwen Tiu. A system of interaction and structure ii: The need for deep inference. *Log. Methods Comput. Sci.*, 2, 2006.

[16] Jacobo Valdes, Robert E Tarjan, and Eugene L Lawler. The recognition of series parallel digraphs. In *Proceedings of the eleventh annual ACM symposium on Theory of computing*, pages 1–12, 1979.