Logic Beyond Formulas:
Designing Proof Systems on Graphs

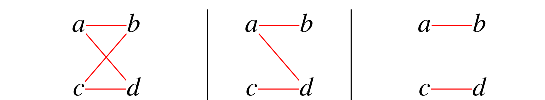
Matteo Acclavio[1]

Università Roma Tre, Roma, Italy

Keeping track of relations between objects or events is essential in modelling 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.

However, relations admitting no series-parallel decomposition [6], which are ubiquitous in distributed systems, cannot be directly treated by the current proof theoretical methods. In fact, the natural correspondence between graphs and formulas (see [6, 11, 8]) fails as soon as simple topological conditions on graphs are not met.

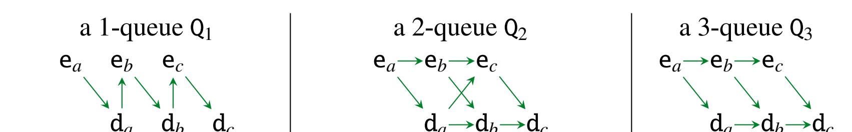
By means of example, consider four processes a, b, c and d where communication between some processes is forbidden because of certain conflicts of interest [5]. Thus, the following pairs cannot communicate: a and b, a and d, and c and d, as shown in the graph below in the centre.



$(a \vee c) \wedge (b \vee d)$	no formula	$(a \wedge b) \vee (c \wedge d)$

(1)

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 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).



$\mathsf{e}_a \triangleleft \mathsf{d}_a \triangleleft \mathsf{e}_b \triangleleft \mathsf{d}_b \triangleleft \mathsf{e}_c \triangleleft \mathsf{d}_c$	$\mathsf{e}_a \triangleleft (\mathsf{d}_a \parr \mathsf{e}_b) \triangleleft (\mathsf{d}_b \parr \mathsf{e}_c) \triangleleft \mathsf{d}_c$	no formula

(2)

The graphs Q_1 and Q_2 are series-parallel graphs and can be directly encoded as formulas. The graph Q_3, and more in general the causality patterns for n-queues with $n > 2$, cannot.

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

For this purpose, we use results on graph modular decomposition [10] to associate abstract syntax trees to graphs, allowing us to generalise the notions of connectives and subformulas to this new setting. We then define a linear implication $\multimap$ and we define proof systemsmeeting certain basic desiderata such as the derivability of the general identity ($G \multimap G$ is provable for any graph G), and the transitivity of implication (if $G \multimap H$ and $H \multimap K$ are provable, then $G \multimap K$ also is). Our proof systems on graphs are presented using the open deduction [9] proof formalism (see Figure 1) based on deep inference [4] since, as observed for the non-commutative logic BV [8], it is not possible to define an analytic sequent calculus for these logics.

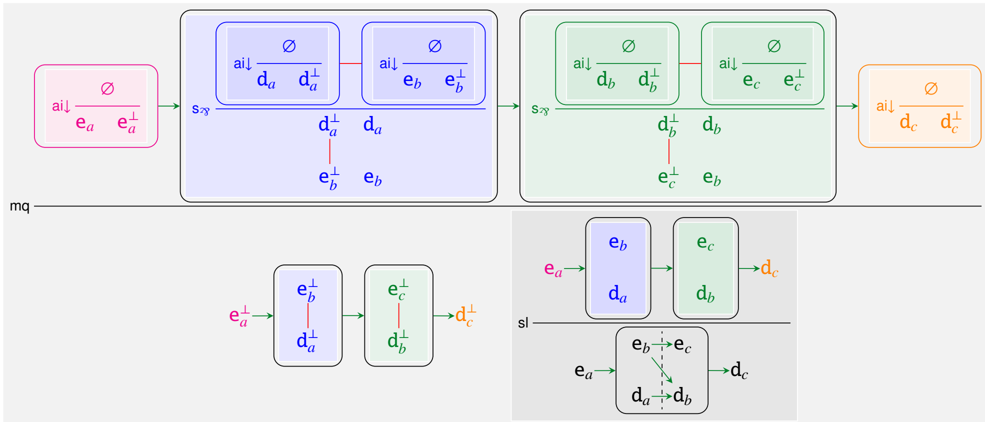


Figure 1: A proof of the graph $\mathsf{Q}_3 \multimap \mathsf{Q}_2$ in the system $\mathsf{GV}^{\mathsf{sl}}$ serving as proof that 3-queues can simulate behaviours of 2-queues. The rule sl slices a directed graph into a "before" and an "after" part by introducing additional directed edges. The rule mq merges the modules of two copies of the same directed graph.

We present the system GS handling undirected graphs as the ones in (1). and we prove that GS is a conservative extension of the *multiplicative linear logic with mix* [7]. Then we present the systems GV and $\mathsf{GV}^{\mathsf{sl}}$ handling graphs with both directed and undirected edges. These systems provide a conservative extension of both the graphical logic defined by GS and the non-commutative logic BV [8]. 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, 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] 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.

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

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

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

[9] 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.

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

[11] 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.