Space-Time Deterministic Graph Rewriting

Pablo Arrighi[1,*,†], Marin Costes[1,†], Gilles Dowek[1,†] and Luidnel Maignan[2,‡]

[1]Université Paris-Saclay, Inria, CNRS, LMF, 91190 Gif-sur-Yvette, France
[2]Univ Paris Est Creteil, LACL, 94000, Creteil, France

Abstract

We study non-terminating graph rewriting models, whose local rules are applied non-deterministically—and yet enjoy a strong form of determinism, namely space-time determinism. For terminating computation, it is well-known that the property of confluence may ensure a deterministic end result. In the context of distributed, non-terminating computation however, confluence alone is too weak a property. Here we provide sufficient conditions so that asynchronous local rule applications produce well-determined events in the space-time unfolding of the graph, regardless of their application orders. Our first two examples are asynchronous simulations of dynamical systems. Our third example features time dilation, in the spirit of general relativity, as a prime illustration of a phenomenon that is fundamentally asynchronous yet gives rise to a consistent space-time.

Keywords

Causal graph dynamics, Cellular automata, Covariance, Commutation, Strong confluence, Distributed computation, Task dependencies, DAG, Poset, Space-like cut, Foliation.

Dynamical systems and asynchronism. Distributed models of physical, biological, social or technological objects are often composed of interacting elements (particles, cells, agents, processes etc.) that evolve according to local rules. From this local evolution, the global evolution may be defined in various ways. Dynamical systems like cellular automata [1] and their graph-based extension causal graph dynamics [2] assume a global clock and synchronous updates, whereas rewriting systems typically rely on asynchronous, non-deterministic rule applications. Synchronism is often criticised for being costly and physically unrealistic, but asynchronism leads to an inherent lack of determinism and inconsistencies therein. We show that even in fully asynchronous graph rewriting, a strong form of determinism can be recovered. To this end, we introduce a graph rewriting formalism based on a DAG of dependencies, together with local conditions on rewrite rules ensuring that asynchronous executions yield well-determined events.

Confluence and non terminating computation. In rewriting systems, this non-determinism arises from the freedom in choosing where and when to apply local rules. A fundamental question is therefore whether different evaluation strategies ultimately agree. In the case of terminating computations, this question is classically addressed by the notion of confluence [3, 4, 5]. More generally, *confluence* of a set of rewrite rules is the property that if the evolutions $a \to^* b$ and $a \to^* c$ are possible, then there exists d such that $b \to^* d$ and $c \to^* d$. In the context of terminating computations, this ensures that the non-determinism arising from the order of rule applications does not affect the final result, yielding a unique normal form.

However, this guarantee breaks down in non-terminating settings. Consider the example of Fig. 1, which shows left and right-moving particles on a line, where left versus right is indicated by the ports along the edges. The local rule simply has them move, which means consuming the particle at vertex, to give it to another. A little thought shows that the system is confluent. Yet, depending upon the chosen order of applications, one finds that one particle propagates much faster than the other, yielding the collision event to occur at very different places. As a result, the underlying computational events are not well-defined: their position and timing depend on the evaluation strategy. More generally, asynchronous

Joint Proceedings of the STAF 2026 Workshops: AgileMDE, GCM, ICMM, LLM4SE, TTC. Rennes, France, June 29–July 3, 2026.
0000-0002-3535-1009 (P. Arrighi); 0000-0003-0915-9192 (M. Costes); 0000-0001-6253-935X (G. Dowek); 0009-0009-5297-5022 (L. Maignan)
© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-26

CEUR-WS.org/Vol-4338/GCM-invited3.pdf

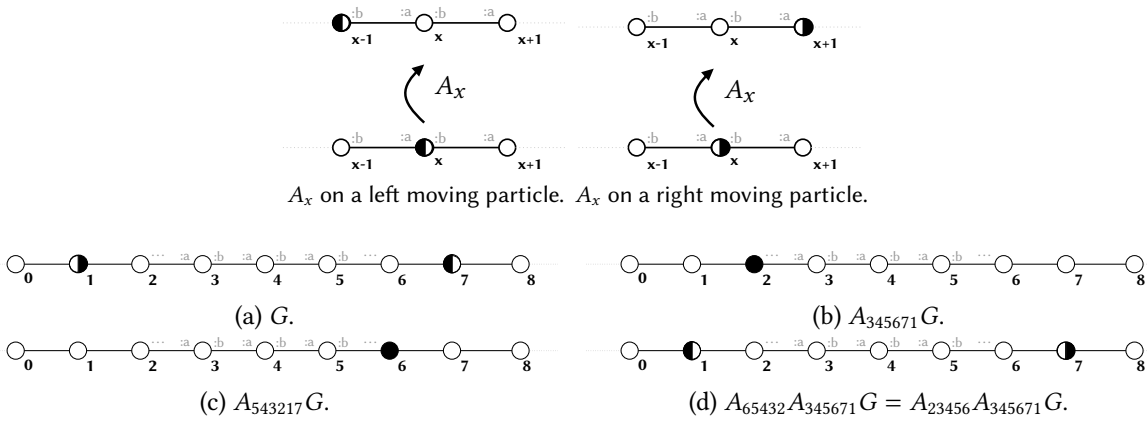


Figure 1: *Confluence yet inconsistencies.* The local rule transports left-moving particles to the left, and right-moving particles to the right, where left/right are distinguished thanks to the a/b ports. (a) We start with a left-moving particle in 7 and a right-moving particle in 1. (b)&(c) The point of collision between the two particles is not well defined, it depends on the evaluation strategy. Here $A_{71}G$ is short for $A_7(A_1G)$. (d). Still, the system is confluent, as the divergent configurations can be both evolved into the last.

rewriting may lead to inconsistent behaviours, nonphysical effects, or pathological dynamics. This is a major limitation of confluence when applied to dynamical systems.

Space-time determinism In order to fix this, we introduce the notion *space-time determinism*. The idea is to move beyond the notion of confluence, which traditionally compares configurations globally while ignoring their spatial and local structures, and introduce a spatially and locally refined notion instead. It will ensure the existence of well-defined, interrelated events constitutive of a global "space-time diagram" (in the sense of cellular automata or Physics). Space-time determinism, considers all events and asks that two snapshots of the evolving graph that have consumed the same information in some region, be consistent upon the (partial) results in this region—regardless of the evaluation strategies used to compute each snapshot. Our main theorems show that some natural, local conditions upon the local rule, entail (weak or full) space-time determinism [6].

Whilst mainly of theoretical nature, this work does suggest efficient parallel schemes for the implementation of dynamical system and distributed systems—doing away with any expensive clock synchronisation mechanisms and replacing it with just a cheap DAG. Another strand of applications lies in Physics, namely seeking for mathematically sound, constructive frameworks for discrete models of general relativity [7]. From a general philosophy of science point of view, we find it compelling to reconcile asynchronism and determinism.

Declaration on Generative AI

During the preparation of this work, the authors used ChatGPT in order to: Paraphrase and reword, Improve writing style. After using these services, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] J. von Neumann, Theory of Self-Reproducing Automata, University of Illinois Press, IL, USA, 1966.
[2] P. Arrighi, G. Dowek, Causal graph dynamics, in: Proceedings of ICALP 2012, 2012, pp. 54–66.
[3] L. Lambers, H. Ehrig, F. Orejas, Efficient conflict detection in graph transformation systems by essential critical pairs, Electronic Notes in Theoretical Computer Science 211 (2008) 17–26.
[4] I. Hristakiev, D. Plump, Checking graph programs for confluence, in: Software Technologies: Applications and Foundations: STAF 2017 Collocated Workshops, Springer, 2018, pp. 92–108.

[5] F. Bonchi, F. Gadducci, A. Kissinger, P. Sobociński, F. Zanasi, Confluence of graph rewriting with interfaces, in: Programming Languages and Systems, 2017, pp. 141–169.
[6] G. D. Pablo Arrighi, Marin Costes, L. Maignan, Space-time deterministic graph rewriting, in: Rewriting Logic and its Applications 16th International Workshop, WRLA 2026, Informal Proceedings., 2026. URL: https://wrla2026.github.io/preproceedings/preproceedings.pdf.
[7] R. Sorkin, Time-evolution problem in Regge calculus, Phys. Rev. D. 12 (1975) 385–396.