Algebraic Petri Nets with Active Tokens: Effective Implementation in Maude

Lorenzo Capra[1], Michael Köhler-Bussmeier[2]

[1]Dipartimento di Informatica, Università degli Studi di Milano - Italy
[2]Hamburg University of Applied Sciences, Berliner Tor 7, D-20099 Hamburg, Germany

Implementing Algebraic Petri Nets

Coloured Petri nets (CPN) [1] are one of the major steps towards practical usage. In some sense, coloured nets are abstract in the sense that an external mechanism is assumed to generate bindings. A concrete formalism is obtained by importing a concrete annotation language and type system for arc expressions, like ML. Resig's Algebraic Petri nets [2] may represent one of the most robust and sophisticated high-level Petri net formalisms proposed to date, integrating clear mathematical semantics grounded in pure equational logic [3, 4] with intuitive operational semantics, high expressiveness, and analytical capabilities.

However, it appears that there are few rigorous implementations derived from this model based on an algebraic framework: notable among these is the OBJSA formalism [5] and the associated tool (based on OBJ [6]), both unfortunately no longer supported. Other implementations of pseudo-algebraic nets have utilised various established programming languages such as Java or Python, partially compromising the original model's soundness.

Translation of Algebraic Petri Nets into Conditional Rewrite Rules We present an effective implementation of algebraic Petri nets utilizing `Maude` [7], a purely declarative algebraic language characterized by sound rewriting logic semantics. Our definition follows the idea originally proposed in [8]. Despite `Maude` being proposed as a unifying framework for Petri Nets two decades ago and some recent works employing it to specify different classes of Petri Nets [9, 10, 11, 12, 13], we contend that our approach is the first systematic utilization of `Maude` as an effective rewriting engine for algebraic Petri Net models.

Additonally, we explore the formalisation of algebraic net inscriptions as terms designed to conform to Maude's pattern-matching (modulo axioms) strategy, which ensures efficiency through the ground coherence property.

Representation of Multisets In this contribution, we explore two potential approaches to cope with these modelling challenges arising from `Maude`'s pattern-matching methodology employed in its operational semantics. The first approach is based on a classical representation of multisets as a free monoid on a set and the second one is based on a more compact representation as weighted sums. A small set of base functional modules delineate the signature of algebraic Petri nets (accessible at https://github.com/lgcapra/rewpt/blob/main/algPT/ALG-PN.maude). The resulting module hierarchy is succinct, modular, and easily extendable For both methods we define a translation procedure that generates conditional rewrite rules from the inscriptions of arcs and transitions.

PNSE'25, International Workshop on Petri Nets and Software Engineering, 2025
capra@di.unimi.it (L. Capra); michael.koehler-bussmeier@haw-hamburg.de (M. Köhler-Bussmeier)
https://www.haw-hamburg.de/michael-koehler-bussmeier (M. Köhler-Bussmeier)
0000-0002-1029-1169 (L. Capra); 0000-0002-3074-4145 (M. Köhler-Bussmeier)

Active Tokens We further demonstrate the advantage of using rewritable terms as active tokens [14, 15, 16, 17, 18], a feature naturally supported by `Maude`. When places are linked to system modules, tokens act as "dynamic entities" that can be rewritten. To demonstrate the advantages of this capability – natural within the context of `Maude` and aligning perfectly with our definition of algebraic nets – we will present an example of meta-modelling. Instead of utilizing the meta-model modules available in the `Maude` system, which are cumbersome and reserved for expert users (and also impact rewrite efficiency), we will showcase an application directly at the object-level. This approach is akin to the nets-within-nets concept, but more generalized: We define an algebraic net where places contain nested algebraic nets, using the same methodology described in the previous sections.

Outlook

We presented an effective methodology for the comprehensive implementation of the algebraic nets schema in `Maude`, a purely declarative algebraic language characterized by sound rewriting logic semantics. The fundamental `Maude` tools for model-checking and formal verification are readily accessible; more sophisticated options (such as symbolic reachability and SMT) may be integrated at a later stage.

Our approach has the advantage that it allows rewritable terms for coloured tokens, i.e., we allow for active tokens. A special case of active case are nets-within-nets, where the tokens are Petri nets again. Our framework allows modifications to the structure and dynamic aspects can be straightforwardly integrated, such as the removal or addition of net places, changes in types linked to places, arcs that depend on parameters or markings, rewrite rules or transitions parametric in pre- and post-sets.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] K. Jensen, L. M. Kristensen, Coloured Petri Nets: Modelling and Validation of Concurrent Systems, Springer-Verlag, 2009. doi:10.1007/b95112.

[2] W. Reisig, Petri nets and algebraic specifications, Theoretical Computer Science 80 (1991) 1–34.

[3] H. Ehrig, B. Mahr, Fundamentals of algebraic Specification, EATCS Monographs on TCS, Springer-Verlag, 1985.

[4] A. Bouhoula, J.-P. Jouannaud, J. Meseguer, Specification and proof in membership equational logic, Theoretical Computer Science 236 (2000) 35–132.

[5] E. Battiston, F. DeCindio, G. Mauri, OBJSA nets: a class of high-level nets having objects as domains, Springer-Verlag, Berlin, Heidelberg, 1988, p. 20–43.

[6] J. Goguen, G. Malcolm, Algebraic Semantics of Imperative Programs, MIT Press, 1996.

[7] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, C. L. Talcott (Eds.), All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic, volume 4350 of *Lecture Notes in Computer Science*, Springer-Verlag, 2007.

[8] R. Bruni, J. Meseguer, U. Montanari, V. Sassone, A comparison of Petri net semantics under the collective token philosophy, in: J. Hsiang, A. Ohori (Eds.), Advances in Computing Science ASIAN 98, Springer Berlin Heidelberg, Berlin, Heidelberg, 1998, pp. 225–244.

[9] J. Padberg, E. Mahr, L. Ribeiro, Algebraic high-level net transformation systems, Mathematical structures in computer science (1995) 217–256.

[10] O. Biberstein, D. Buchs, N. Guelfi, Object-oriented nets with algebraic specifications: The co-opn/2 formalism, in: G. A. Agha, F. De Cindio, G. Rozenberg (Eds.), Concurrent Object-Oriented Programming and Petri Nets: Advances in Petri Nets, Springer, Berlin, Heidelberg, 2001, pp. 73–130. URL: https://doi.org/10.1007/3-540-45397-0_3. doi:10.1007/3-540-45397-0_3.

[11] M.-O. Stehr, J. Meseguer, P. C. Ölveczky, Rewriting logic as a unifying framework for Petri nets, in: H. Ehrig, G. Juhas, J. Padberg, G. Rozenberg (Eds.), Unifying Petri Nets, Lecture Notes in Computer Science (Advances in Petri Nets), Springer-Verlag, 2001.

[12] K. Hoffmann, H. Ehrig, T. Mossakowski, High-level nets with nets and rules as tokens, in: Application and Theory of Petri Nets and Other Models of Concurrency, volume 3536 of *Lecture Notes in Computer Science*, Springer-Verlag, 2005, pp. 268 – 288.

[13] J. Padberg, L. Kahloul, Overview of reconfigurable Petri nets, in: R. Heckel, G. Taentzer (Eds.), Graph Transformation, Specifications, and Nets, volume 10800, Springer-Verlag, 2018, pp. 201–222.

[14] R. Valk, Object Petri nets: Using the nets-within-nets paradigm, in: J. Desel, W. Reisig, G. Rozenberg (Eds.), Advanced Course on Petri Nets 2003, volume 3098 of *Lecture Notes in Computer Science*, Springer-Verlag, 2003, pp. 819–848.

[15] K. Hoffmann, T. Mossakowski, Algebraic higher-order nets: Graphs and petri nets as tokens, in: Recent Trends in Algebraic Development Techniques, 16th International Workshop, WADT 2002, volume 2755 of *Lecture Notes in Computer Science*, Springer-Verlag, 2003, pp. 253–267.

[16] M. Köhler-Bußmeier, A survey on decidability results for elementary object systems, Fundamenta Informaticae 130 (2014) 99–123. doi:10.5555/2608435.2608440.

[17] M. Köhler-Bußmeier, Hornets: Nets within nets combined with net algebra, in: K. Wolf, G. Franceschinis (Eds.), Application and Theory of Petri Nets (ICATPN'2009), volume 5606 of *Lecture Notes in Computer Science*, Springer-Verlag, 2009, pp. 243–262. doi:10.1007/978-3-642-02424-5_15.

[18] L. Capra, M. Köhler-Bußmeier, Modular rewritable Petri nets: an efficient model for dynamic distributed systems, Theoretical Computer Science 990 (2024) 114397. doi:10.1016/j.tcs.2024.114397.