Western Norway
University of
Applied Sciences

Heterogeneous behavioral model composition using graph grammars

NWPT 2021

Tim Kräuter
05. november 2021

Motivation: Model-driven engineering (MDE) I

Software systems and their development becomes more and more complex.

Separation of concerns

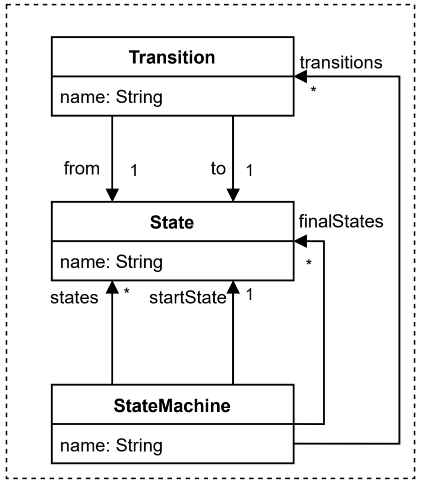
Abstraction of details

Set of **related** heterogeneous models

Motivation: Model-driven engineering (MDE) II

Structural models

Class diagram



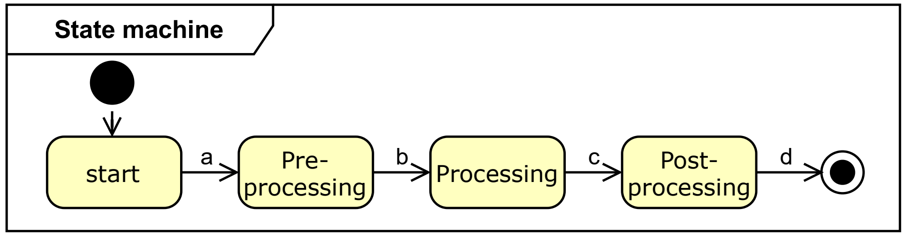
Entity relationship models

Object diagrams

…

Behavioral models

State machine



Activity diagrams

Process algebras

Petri-nets

BPMN models

…

Relations between structural models, their merge and consistency checking is actively researched [5, 6].

What about behavioral models and relations between them?

Related work

There is research on homogeneous model composition (state machines, petri nets, etc.)

There are ad-hoc solutions for specific combinations of behavioral models.

[5] proposes event structures as an underlying formalism but only uses it for homogeneous composition.

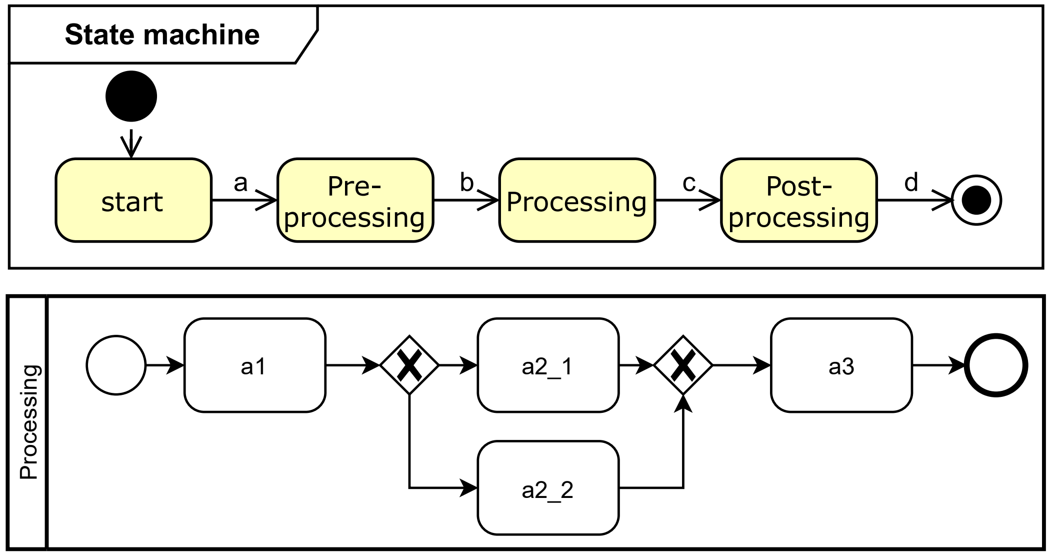
There are tool implementations for the simulation of heterogenous models [2, 4].
However, they do not generate a state space suitable for model checking.

Lack in the research of using behavioral models in MDE.

Consistency problem in MDE

Multiple behavioral models are not independent of each other.

Their behavior must be compatible.



How to check global properties?

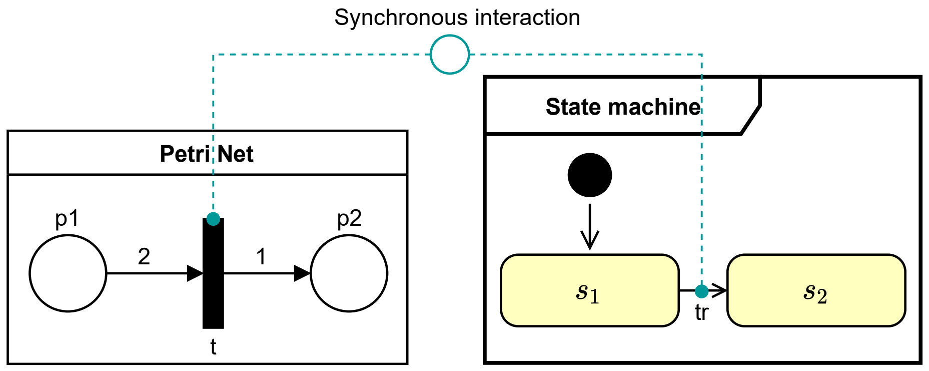
1. Define interactions between the models.

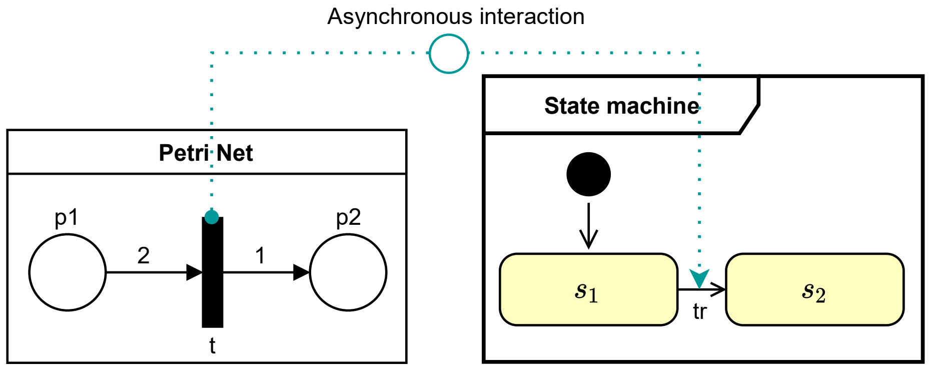
2. Generate a global model based on the models and their interactions.

Heterogeneous behavioral model composition I

Each behavioral model describes a component of the overall system running independently.

1. Define interactions between the heterogeneous models.



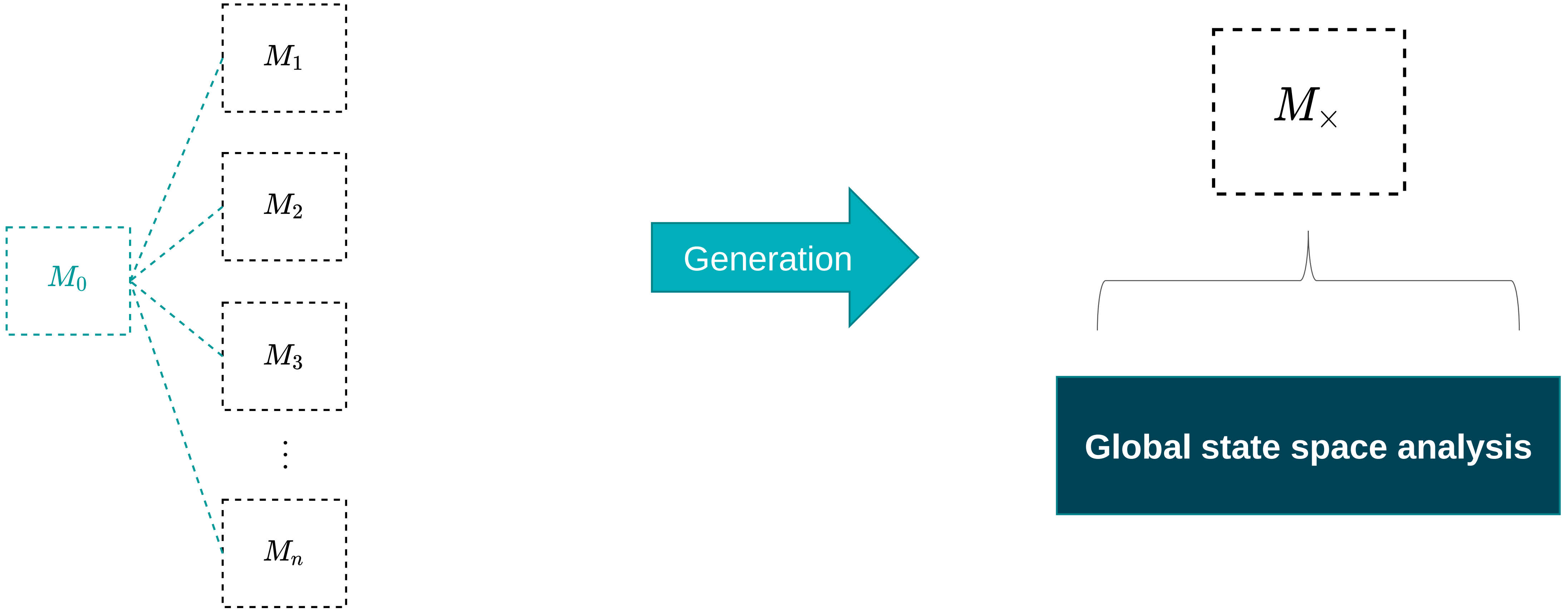


Synchronous interaction

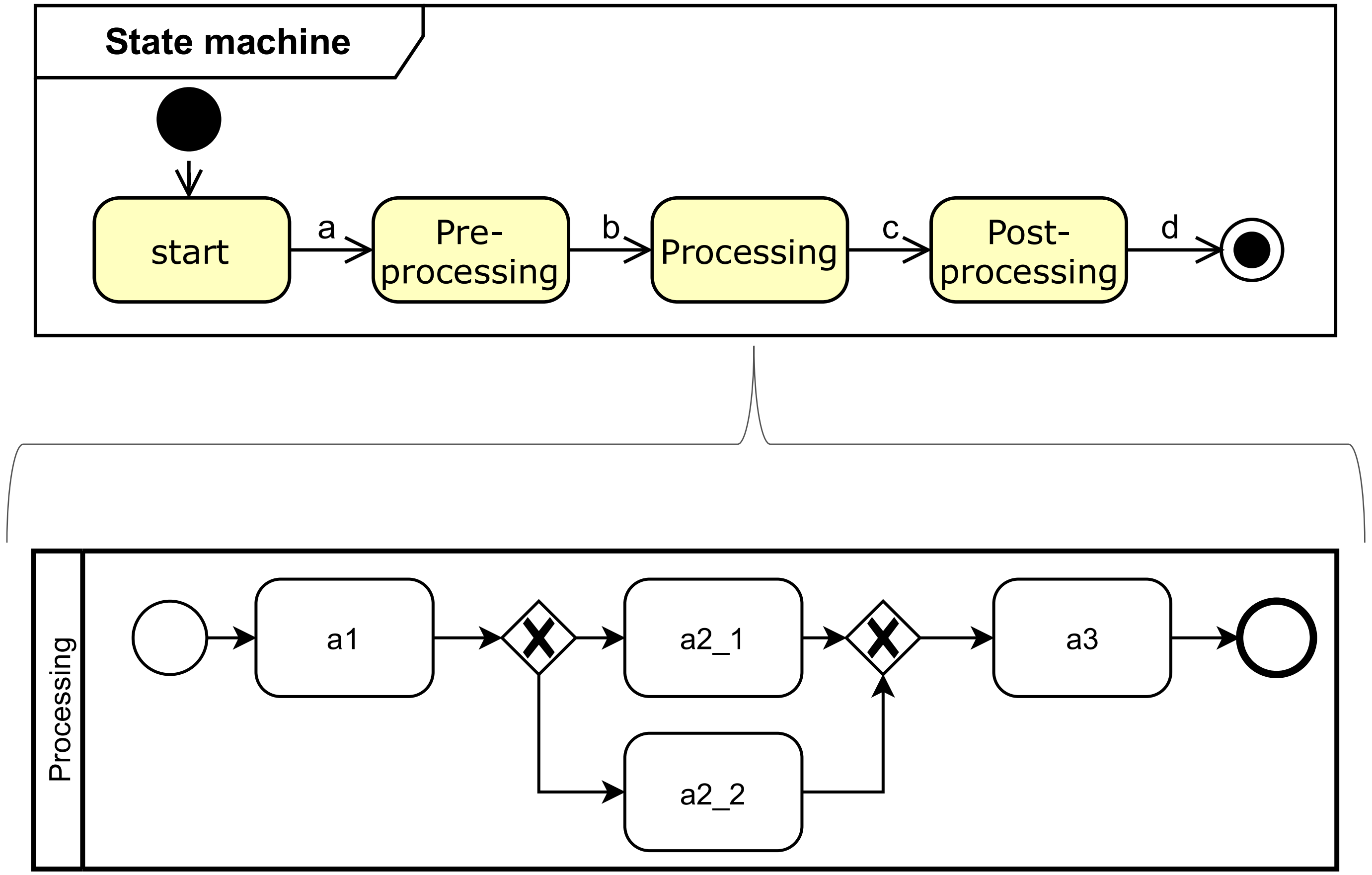
Asynchronous interaction

Heterogeneous behavioral model composition II

2. Generate a global model in a chosen formalism.



Implementation using graph grammars: Running example



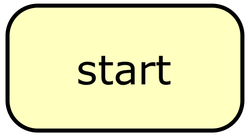
Hierarchical behavioral model composition

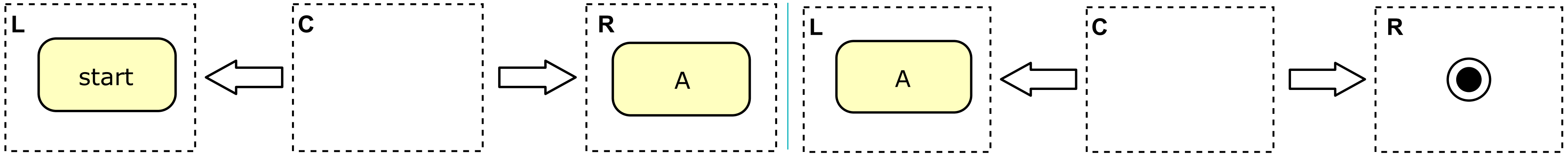
What is a graph grammar (GG)?

Graph grammar [3]

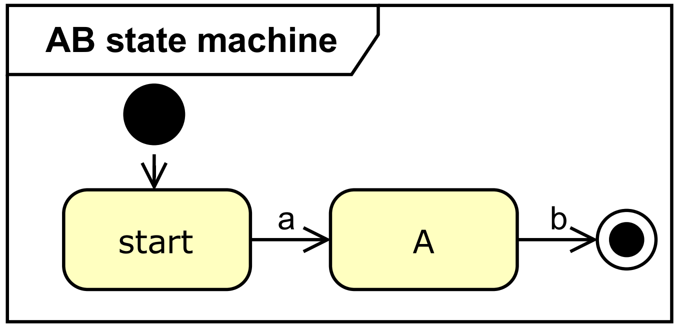
Start graph

Set of production rules

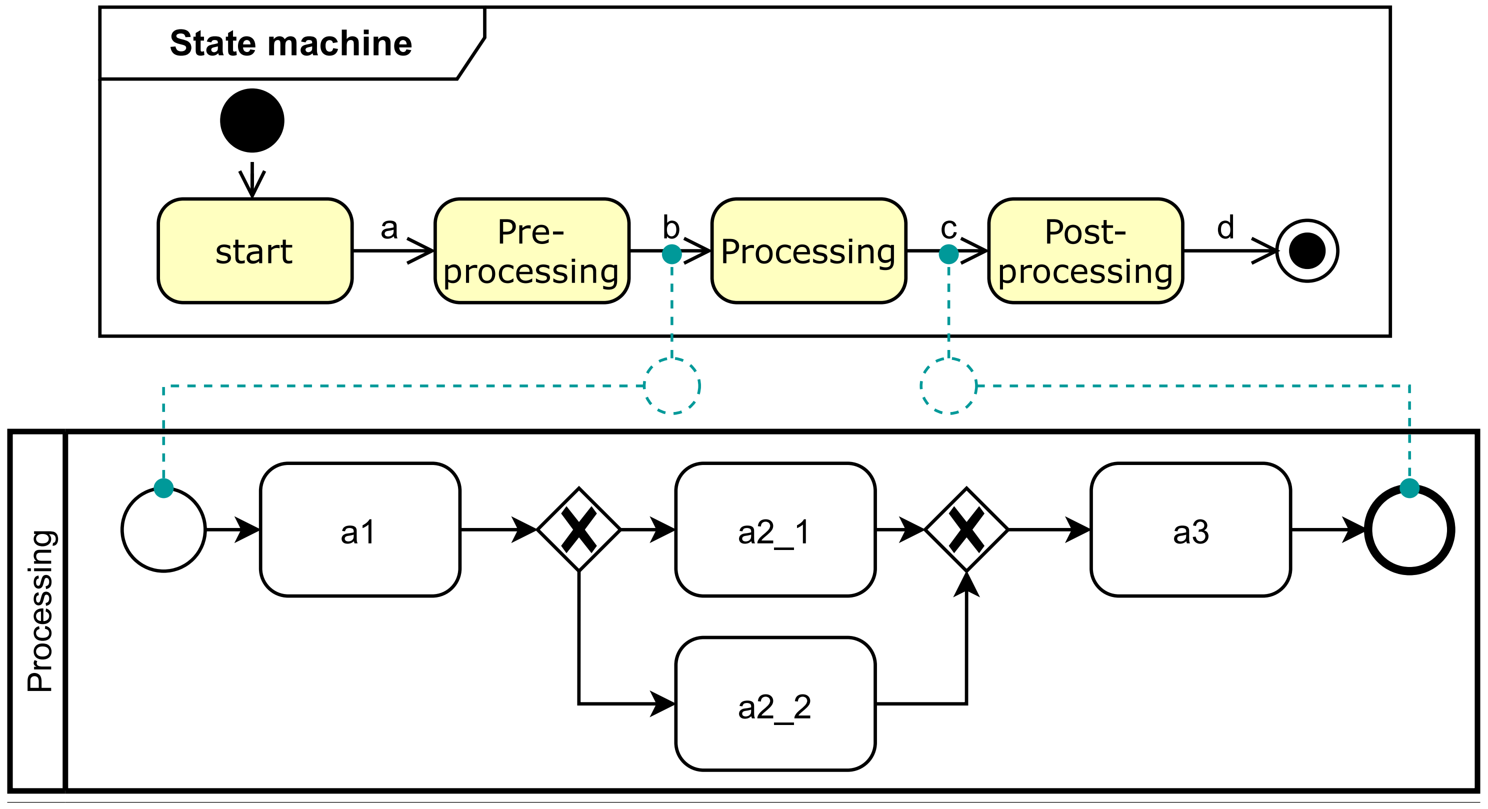




State machine example

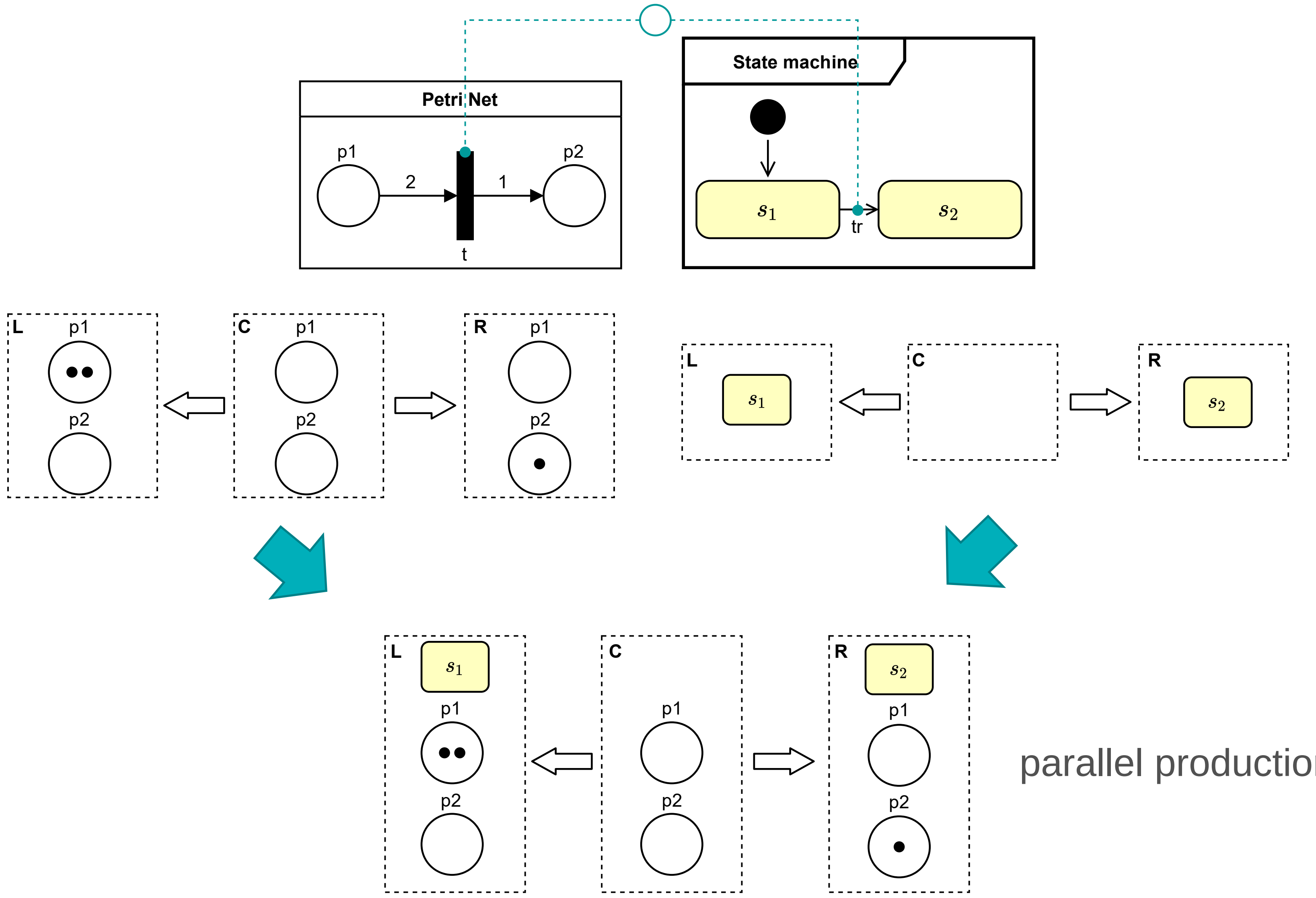


Interactions in the example



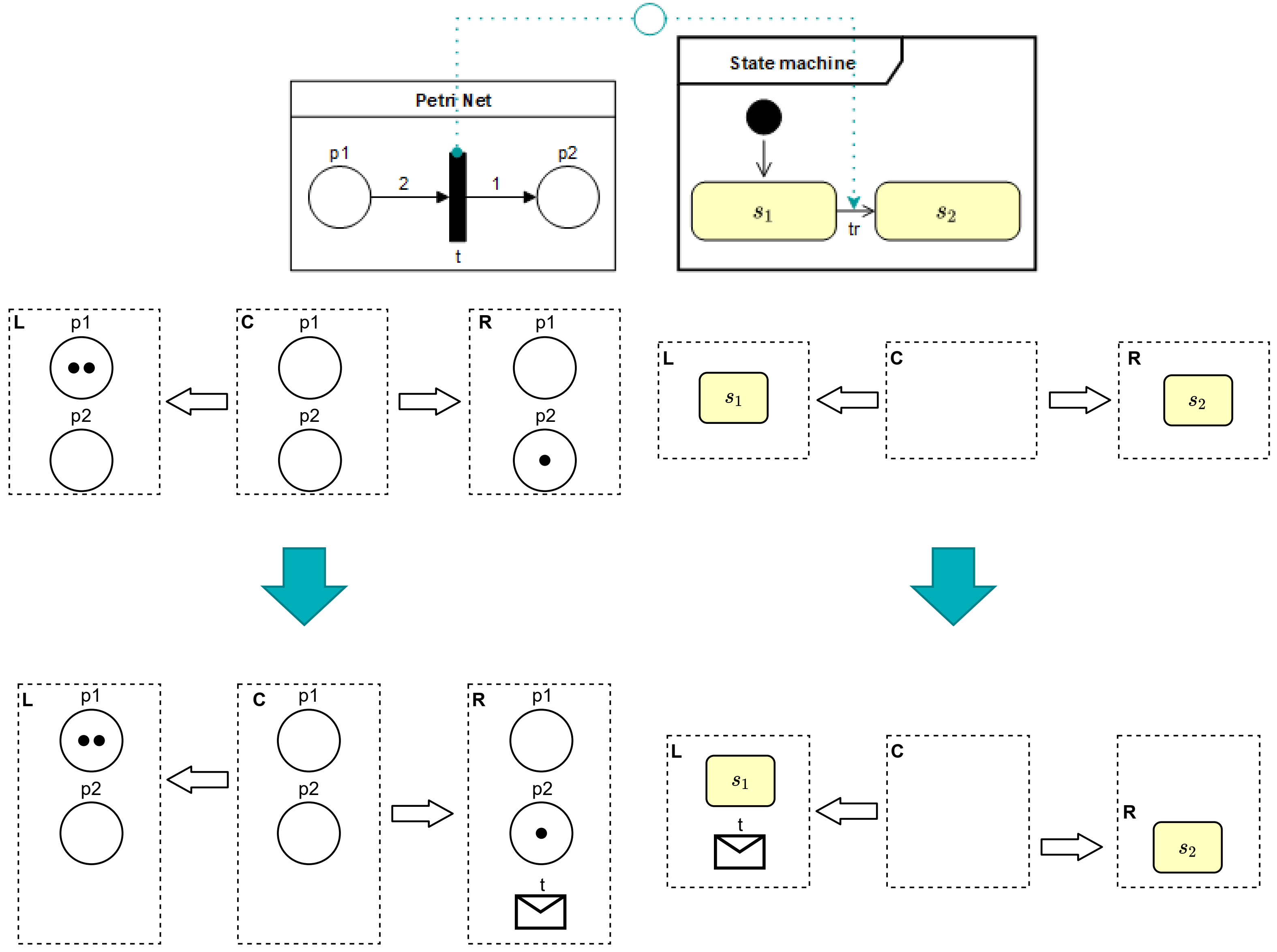
Synchronous interactions between transitions and start and end event.

Synchronous interaction with GGs



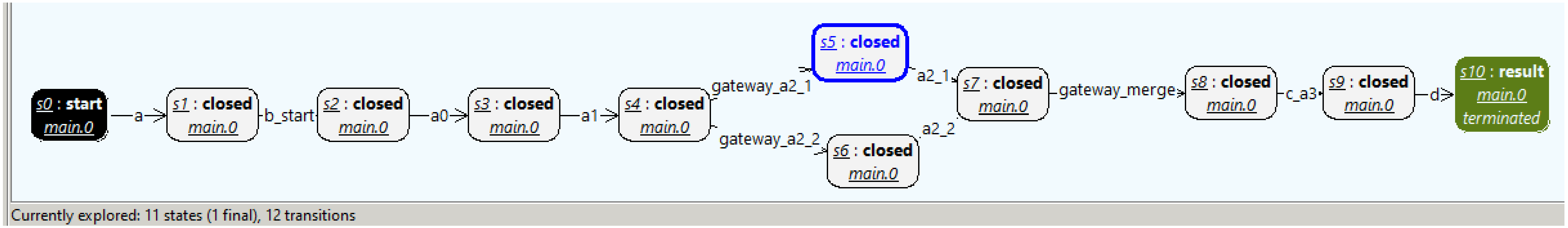
parallel production rule [1]

Asynchronous interaction with GGs



State space for the example using GROOVE

GROOVE can be used to simulate graph grammars.



Currently explored: 11 states (1 final), 12 transitions

Global state space can be generated and analyzed.

For example: Check termination using LTL in groove.

Conclusion

Approach for heterogeneous behavioral composition

Coordination language for heterogeneous models

Semantic model composition respecting interactions

The approach is independent of the chosen underlying formalism.

Graph grammars are proposed as a possible underlying formalism.

Checking of global behavioral properties (for example for behavioral consistency in MDE).

The end

References I

[1]: Paolo Baldan, Andrea Corradini, Ugo Montanari, Francesca Rossi, Hartmut Ehrig, and Michael Löwe. Concurrent semantics of algebraic graph transformations. In Handbook of Graph Grammars and Computing by Graph Transformation, volume 3,pages 107–188. World Scientific, August 1999.
[2]: Julien Deantoni. Modeling the Behavioral Semantics of Heterogeneous Languages and their Coordination. In2016 Architecture-Centric Virtual Integration (ACVI),pages 12–18, Venice, Italy, April 2016. IEEE.
[3]: Hartmut Ehrig and Julia Padberg. Graph Grammars and Petri Net Transformations. In Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Dough Tygar, Moshe Y. Vardi, Gerhard Weikum, Jörg Desel, Wolfgang Reisig, and Grzegorz Rozenberg, editors, Lectures on Concurrency and Petri Nets, volume 3098, pages 496–536. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.
[4]: J. Eker, J.W. Janneck, E.A. Lee, Jie Liu, Xiaojun Liu, J. Ludvig, S. Neuendorffer, S. Sachs, and Yuhong Xiong. Taming heterogeneity - the Ptolemy approach. Proceedings of the IEEE, 91(1):127–144, January 2003

References II

[5]: Jörg Kienzle, Gunter Mussbacher, Benoit Combemale, and Julien Deantoni. A unifying framework for homogeneous model composition. Software & Systems Modeling,18(5):3005–3023, October 2019.
[6]: Patrick Stünkel, Harald König, Yngve Lamo, and Adrian Rutle. Comprehensive Systems: A formal foundation for Multi-Model Consistency Management. Formal Aspects of Computing, July 2021.