A Toolbox for the Analysis of Discrete Event Dynamic Systems

Peter Buchholz and Peter Kemper*

Informatik IV, Universität Dortmund, D-44221 Dortmund, Germany

Abstract. We present a collection of tools for functional and quantitative analysis of discrete event dynamic systems (DEDS). Models can be formulated as a set of automata with synchronous communication or as Petri nets. Analysis takes place with a strong emphasis on state based analysis methods using Kronecker representations and ordered natural decision diagrams. Independent tools provide access to orthogonal techniques from different fields including computation of bisimulation equivalences, modelchecking, numerical analysis of Markov chains, and simulation. Two file formats are defined to provide a simple exchange mechanism between independent tools which allows to build various combinations of tools.

1 Introduction

Tools for the specification and analysis of DEDS exist in a rich variety and show a certain combinatorial explosion from the set of modeling formalisms and the set of analysis techniques. Selection of the "best" modeling formalism is a highly emotional topic, but for selection of analysis techniques criteria boil down to availability of implementations and applicability for a given model. We observed severe difficulties in exchanging models between different tools, such that for our toolbox a strong emphasis is on a simple exchange of information between the independent tools it contains. Fig. 1 gives an overview: the tools are arranged around two file formats by which models can be specified as a set of automata with synchronous communication and as a hierarchical, colored Petri net. The Petri net formalism - named *abstract Petri net notation* (APNN) [2] - integrates several kinds of Petri nets, including place/transition nets, colored Petri nets, timed nets with a stochastic timing, hierarchical nets using place and/or transition refinement, superposed nets based on transition fusion.

Networks of automata with synchronous interaction are specified in a different format, using state transition matrices and synchronization via equal labels. This format directly corresponds to a Kronecker representation of the state transition matrix of the composed model. The representation is compositional and uses a Kronecker product to express synchronization and a Kronecker sum for independent state transitions. The state space of the composed model can be represented in a (usually) very space efficient manner by a directed acyclic graph,

* This research is supported by Deutsche Forschungsgemeinschaft, SFB 559.

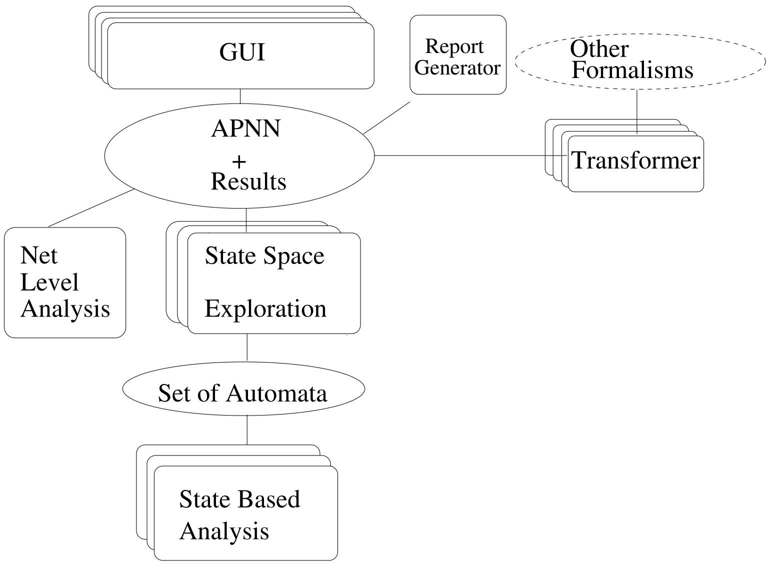

Fig. 1. Structural overview of the toolbox

a generalization of ordered binary decision diagrams (OBDDs). The latter allows exchange of state space descriptions among tools with low effort [8,14]. In the sequel we briefly sketch ways to use the APNN toolbox for modeling and analysis, for details we refer to [1].

2 Several Ways to Obtain a Model

Models can be either generated by a the grapical user interface (GUI) contained in the toolbox or translated from other modeling formalisms supported by other tools. A generation by hand, directly editing the textual description at APNN level or the matrix representation at the automata level is possible in principle but not recommended. APNNed [9] provides a GUI for the APNN format. It is a JAVA implementation of a Petri net editor supporting hierarchical nets (based on refinement of places and transitions) and colored nets with finite color sets. APNNed animates the dynamic behavior of a model by the token game in three ways: a) interactive, b) automatic selection of transitions to fire and c) trace-driven by importing a trace generated elsewhere. APNNed also provides functionality to start various analysis tools and to present their results. The analysis tools export results in a specific output format which can be used by the GUI or report generators for result presentation. The toolbox also provides transformers which translate other formats into APNN. These include a translator for generalized stochastic Petri nets (GSPNs) specified by GreatSPN to APNN, a translator for PEP low level nets (which are Place/Transition nets) to APNN and vice versa, and a transformer for Net/Condition Event systems (NCES) to APNN. The latter is a non-trivial mapping [10,13] since certain features of NCES have no direct correspondence in the Petri net formalism. A model can also be described at the level of networks of synchronized automata. However, so far no direct user interface is available at this level. A model for a network of automata is automatically generated from a Petri net model in APNN format by the state space exploration tool, cf. Fig. 1. This tool does not necessarily perform an exploration of the overall state space, it is also used to do an exploration by components which maps a set of submodels of a Petri net to a set of automata (provided the Petri net is appropriately structured). Nevertheless, the interface format can be also used to obtain suitable networks of automata from other compositional formalisms, e.g. from CCS-like process algebraic terms, if they are in the form of $(P_1|P_2|\ldots|P_n)\backslash L$ with processes/agents $P_1,\ldots,P_N$ and a set of synchronization labels L, however a corresponding tool is currently not available in our toolbox.

3 Several Ways to Analyze a Model

The APNN toolbox provides tools for functional and quantitative analysis which apply either at net level or automata level. Only a minority is devoted to Petri nets at the APNN level including computation of invariants and a simulator for quantitative analysis of timed nets. A state space exploration tool transforms an APNN description into the format of the automata level with optional exploration of the overal state space of the composed model as in [12]. A strong emphasis is on tools which exploit the Kronecker structure implicitly given at the automata level. At this level a tool for computation of several equivalences of bisimulation type and subsequent aggregation of automata is available. Especially a weak backword bisimulation preserving reachability is useful in combination with state space exploration of composed automata since a disaggregation module can finally retranslate the resulting state space of an aggregated system into the state space of the original system [7,8]. A generalization of ordered binary decision diagrams (OBDDs) where nodes are allowed to have a variable number of outgoing arcs (ONDDs) has been successfully applied to represent extremely large states space in space efficient way [8,7,14], such that a corresponding file format allows to communicate state spaces between tools. Furthermore a model checker for computational tree logic (CTL) is available at this level [14] implementing classical model checking algorithms adapted to Kronecker algebra. An additional specialized model checker is exclusively devoted to check the liveness property in terms of Petri net theory. In case of (partial) deadlocks it generates a trace of transition firings which can be animated by APNNed. The APNN toolbox also contains a variety of tools for quantitative analysis of stochastic models based on the numerical solution of Markov chains which again profit from the Kronecker structure available at the automata level. These tools can be further distinguished according to hierarchical, block-structured Kronecker representations or modular Kronecker representations, see e.g. [5,3,4,6,11] for corresponding algorithms. A compositional representation based on Kronecker algebra is a key advantage of the analysis tools in our toolbox, since this gives a very space efficient data structure for large state transition systems with possibly millions of states.

References

1. F. Bause, P. Buchholz, and P. Kemper. A toolbox for functional and quantitative analysis of DEDS. Forschungsbericht 680, Fachbereich Informatik, Universität Dortmund (Germany), 1998.
2. F. Bause, P. Kemper, and P. Kritzinger. Abstract Petri net notation. *Petri Net Newsletters*, 49:9–27, Oct 1995.
3. P. Buchholz. Numerical solution methods based on structured descriptions of Markovian models. In G. Balbo and G. Serazzi, editors, *Computer Performance Evaluation - Modelling Techniques and Tools*, pages 251–267. Elsevier, 1992.
4. P. Buchholz. Hierarchical structuring of superposed GSPNs. In *Proc. 7th Int. Workshop Petri Nets and Performance Models (PNPM'97), St-Malo (France), June 1997*, pages 81–90. IEEE CS Press, 1997.
5. P. Buchholz. An adaptive aggregation/disaggregation algorithm for hierarchical Markovian models. *European Journal of Operational Research*, 116(3):85–104, 1999.
6. P. Buchholz, G. Ciardo, S. Donatelli, and P. Kemper. Complexity of Kronecker operations on sparse matrices with applications to the solution of Markov models. Technical report, ICASE Report No. 97-66 NASA/CR-97-206274, 1997. submitted for publication.
7. P. Buchholz and P Kemper. Efficient computation and representation of large reachability sets for composed automata. Forschungsbericht 705, Fachbereich Informatik, Universität Dortmund (Germany), 1999.
8. P Buchholz and P. Kemper. Modular state level analysis of distributed systems - techniques and tool suppport. In *accepted for 5th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS '99)*, 1999.
9. P. Buchholz, P. Kemper, and the APNNed group. APNNed - a net editor and debugger within the APNN toolbox. In J. Desel, P. Kemper, E. Kindler, and A. Oberweis, editors, *Proc. 5. Workshop Algorithmen und Werkzeuge für Petrinetze*, pages 19–24. Forschungsbericht Nr. 694, FB Informatik, Universität Dortmund, Germany, 1998.
10. H. Hanisch, P. Kemper, and A. Lüder. A modular and compositional approach to modeling and controller verification of manufacturing systems. In *accepted for 14th IFAC Word Congress, July 5-9, 1999, Beijing, China*, 1999.
11. P. Kemper. Numerical analysis of superposed GSPNs. *IEEE Trans. on Software Engineering*, 22(9), Sep 1996.
12. P. Kemper. Reachability analysis based on structured representations. In *Application and Theory of Petri Nets*, LNCS 1091. Springer, 1996.
13. P Kemper. A mapping of autonomous net condition event systems to GSPNs. submitted for publication, 1999.
14. P. Kemper and R. Lübeck. Model checking based on kronecker algebra. Forschungsbericht 669, Fachbereich Informatik, Universität Dortmund (Germany), 1998.