Symbolic Model Checking in the Modular State Space using Binary Decision Diagrams

Lukas Zech[0009−0003−7022−6021](✉)

University of Rostock, Schwaansche Str. 2, 18055 Rostock, Germany
lukas.zech@uni-rostock.de

Abstract. A *modular Petri net* is composed of multiple individual Petri nets, the modules. Modules are composed by fusing their *interface* transitions. The behavior of *internal* transitions is unrelated to other modules and is recorded in *local reachability graphs* for each module.
Fusion vectors describe which interface transitions of which modules need to synchronize for their firing. This behavior is recorded in the *synchronization graph*, linking the local reachability graphs together. For this, internal behavior between two firings of interface transitions is abstracted to form *segments*. A vertex in the synchronization graph collects segments for each module and records the changes from synchronized interface transitions. The *modular state space* [3] consists of the local reachability graphs and the synchronization graph.
Model Checking describes an exhaustive search through the state space to verify a property. For Symbolic Model Checking, the state space is encoded to combat the state explosion problem [5]. In this paper, we encode the local reachability graphs using Binary Decision Diagrams (BDD) [2,4]. To encode the reachability set of a module using BDDs according to [4], 1-safe modules are assumed. For each module, boolean variables are introduced for each place. A BDD encoding a reachability set then describes the characteristic function of the set as a directed acyclic graph in the form of a compressed decision tree. Similarly, BDDs can be used to encode segments in the local reachability graphs. Since BDDs encode *sets* and segments abstract away internal behavior, leaving only sets of markings, the encoding does not suffer great loss of information.
We study the reachability of markings in the composed Petri net that satisfy *state predicates*. A state predicate is a conjunction of *atomic* state predicates. An atomic state predicate is an inequality over places, i.e. $\sum_i \lambda_i p_i \leq \lambda$, with $\lambda_i, \lambda \in \mathbb{Z}$. We aim to verify the reachability of such markings using the modular state space, without constructing the state space of the composed Petri net. A method to solve this is presented in [3]. There, the problem can be divided into two parts: Atomic state predicates where all occurring places belong to the same module (1) and atomic state predicates where occurring places spread across multiple modules (2).
In the first case, reachability can be verified using just the local reachability graph of the corresponding module. The atomic state predicate can be encoded as a BDD that characterizes the set of markings satisfying the predicate. In [1], a method for this is presented assuming $\lambda_i \in \mathbb{N}$. By

joining that BDD with one that encodes a segment, the intersection of the marking sets can be checked for satisfying markings.
For the second case, according to [3], an atomic proposition can be split according to the module affiliation of places, resulting in partial sums for every module. We present a way to calculate these sums for segments encoded as BDDs by analyzing their structure. This way, these atomic state predicates can be verified for vertices in the synchronization graph. Combining both methods, we can verify such state predicates using the modular state space.

References

1. Abío, I., Nieuwenhuis, R., Oliveras, A., Rodríguez-Carbonell, E., Mayer-Eichberger, V.: A New Look at BDDs for Pseudo-Boolean Constraints. Journal of Artificial Intelligence Research **45**, 443–480 (Nov 2012). https://doi.org/10.1613/jair.3653
2. Burch, J.R., Clarke, E.M., McMillan, K.L., Dill, D.L., Hwang, L.J.: Symbolic model checking: 1020 States and beyond. Information and Computation **98**(2), 142–170 (Jun 1992). https://doi.org/10.1016/0890-5401(92)90017-A
3. Gaede, J., Wallner, S., Wolf, K.: Modular State Spaces - A New Perspective. In: Kristensen, L.M., van der Werf, J.M. (eds.) Application and Theory of Petri Nets and Concurrency. pp. 312–332. Springer Nature Switzerland, Cham (2024). https://doi.org/10.1007/978-3-031-61433-0_15
4. Pastor, E., Cortadella, J., Roig, O.: Symbolic analysis of bounded Petri nets. IEEE Transactions on Computers **50**(5), 432–448 (May 2001). https://doi.org/10.1109/12.926158, conference Name: IEEE Transactions on Computers
5. Valmari, A.: The state explosion problem. In: Reisig, W., Rozenberg, G. (eds.) Lectures on Petri Nets I: Basic Models: Advances in Petri Nets, pp. 429–528. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (1998). https://doi.org/10.1007/3-540-65306-6_21, https://doi.org/10.1007/3-540-65306-6_21