ITPN-PerfBound: A Performance Bound Tool for Interval Time Petri Nets

Elina Pacini Naumovich*, Simona Bernardi, and Marco Gribaudo

Dipartimento di Informatica, Università di Torino, Torino (Italy)
{pacini,bernardi,marcog}@di.unito.it

Abstract. The *ITPN-PerfBound* is a tool for the modeling and analysis of Interval Time Petri Nets (ITPN), that is Petri Nets in which firing time intervals, and possibly firing frequency intervals, are associated to transitions. The tool is particularly well-suited in the verification and validation activities of real-time systems, where the main goal is to give guarantees about the worst and best case system performance. The tool has been implemented within the *DrawNET* framework and supports the analysis of ITPN models based on the computation of upper and lower bounds of classical performance metrics, such as throughput and cycle time.

1 Introduction

In the verification and validation activities of real-time systems, one of the main goals is to give guarantees about the best and the worst system performance, e.g., best/worst execution time, before such systems are put into use. Interval Time Petri Nets (ITPNs) and related bound computation techniques can be used for this purpose [1]. ITPNs include Time Petri Nets (TPNs) [2] and TPNs with firing frequency intervals (TPNFs) [1]. TPNs are Petri Nets in which firing time intervals are associated to transitions. TPNFs reduce the non-determinism of free-choice conflicts in TPNs by assigning a firing frequency interval to each transition in free-choice conflict. ITPN performance bound computation techniques are efficient techniques that can be applied to compute bounds of transition throughput, cycle time, and place marking. The bounds are computed by solving a linear programming problem (LPP) derived from the structure of the net, the initial marking and the firing time/frequency interpretation.

Currently, there are many tools for the modeling and analysis of TPNs, while no tools for TPNF are available (e.g., see the Petri Net tool data-base for an updated list [3]). Several of them (e.g., [4,5,6]) provide support for the behavioral analysis of TPNs, based on the use of enumerative techniques. However, to the best of our knowledge, none of them have performance bound analysis capabilities.

* Corresponding author. Elina Pacini Naumovich has been supported by the World Wide Style project *"Sviluppo di metodi e tecniche per la validazione dell'affidabilità dei sistemi software critici"* of the University of Torino.

S. Kowalewski and A. Philippou (Eds.): TACAS 2009, LNCS 5505, pp. 50–53, 2009.

In this paper, we present *ITPN-PerfBound*, a graphical tool for the modeling and performance bound analysis of ITPNs. The tool has been implemented within the *DrawNET* framework [7], which provides facilities for the design and solution of models expressed in any graph-based formalism. In particular, *DrawNET* supports a two level structure. The *meta-level* defines the formalism being used, and the *level* describes the model belonging to the formalism. The models can be solved using other existing tools (i.e., GreatSPN [8]), interfaced to the *DrawNET* framework throughout *Solvers*. A Solver is a small interface that translates a model in a particular format (using an entity called *Filter*), and then takes care of invoking the tool that produces the actual results. It also extracts the particular measures and stores them back inside the corresponding model.

Finally, to support the inter-operability between PN tools, *ITPN-PerfBound* includes the possibility of exporting (1) TPN models, by using the standard PNML format [9] enriched with timing information which are compliant with the TINA tool [4], and (2) un-timed PN models toward the GreatSPN tool [8].

2 Overall Architecture and Main Functionalities

ITPN-PerfBound has been implemented within the *DrawNET* framework. We used the *DrawNET* XML-interchange format FDL (*Formalism Definition Language*) for defining the ITPN formalism. The *DrawNET* GUI reads the ITPN formalism, written in FDL and presents to the user a graphical user interface for designing models of that formalism. Figure 1 shows the overall architecture of the *ITPN-PerfBound* tool (part inside the grey rectangle) and its interaction with other Petri Net tools (i.e., TINA and GreatSPN). The components are depicted as rounded-cornered rectangles while the information flow between components is indicated by arrows.

The *DrawNET ITPN module* is the editor component that is used by the modeler to construct ITPN models. The *DrawNET ITPN module* and the *Model/ Result Filters* exchange ITPN models described with the DDL (*Data Definition Language*). The *Model Filter* produces a set of input files for the *ITPN bound solvers*, containing information on the ITPN model definition and on the metrics to be computed. In particular, the un-timed specification of the ITPN model is translated into the format of the GreatSPN tool (.net/.def files), then enabling the user to exploit the GreatSPN facilities for the structural analysis of the untimed version of such ITPN models. The *Model Filter* translates also the ITPN models into the standard PNML format [9], enriched with timing information which are compliant with the TINA tool.

The *ITPN bound solvers* include a set of solution components, written in C-language, that implement the ITPN performance bound techniques for ITPNs with generic topology and for special structural classes of ITPNs. The *LPP generator* components produce the LPP from the structure of the ITPN model and the initial marking (.net/.def files), the specification of the transition firing time/frequency (.itpn file), the user query about the metric of interest (.cmd file). Two LPP generators have been implemented that create a different LPP,

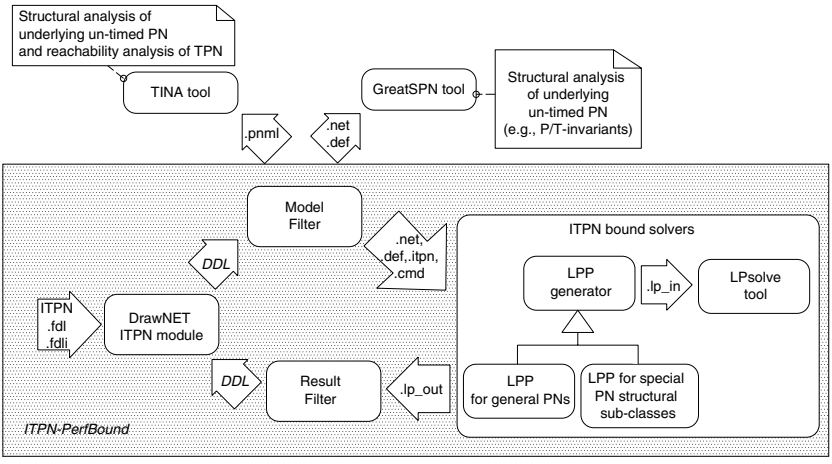

Fig. 1. Components interaction and information flow

according to the type of performance bound technique chosen, and return the LPP written in the input format (.lp_in file) of the free-source *LPsolve tool* [10]. The latter is used to compute the LPP solution (.lp_out file). The results of the performance bound analysis are fed back to the original ITPN model by the *Result Filter* and displayed by *DrawNET ITPN module*.

From the user point of view, the tool provides the following functionalities:

- Create ITPN models by loading the ITPN module from the *DrawNET* GUI and save them in the *DrawNET* XML-interchange format (i.e., mdl).
- Compute performance upper/lower bounds of transition throughput, cycle time and place marking for ITPN model with generic topology. The result of the analysis (i.e., the optimal value of the LPP objective function) is displayed in the *Properties* tab associated to the corresponding net object (transition/place) in the *DrawNET ITPN module*.
- Compute best/worst case performance transition cycle time and throughput for special structural classes of ITPNs (i.e., Marked Graphs and 1-consistent monoT-semiflows). The presentation of the results (i.e., the LPP optimal solution and the corresponding value of the objective function) is both textual and graphical: the bound values are displayed in the *Properties* tab associated to the net model in the *DrawNET ITPN module* and the slowest sub-net of the ITPN model (i.e., the sub-net with the highest cycle time) is highlighted in red.
- Export the ITPN models toward the GreatSPN tool [8] and the standard PNML format, enriched with timing information which are compliant with the TINA tool [4].

3 Applications and Conclusion

The *ITPN-PerfBound* tool has been applied to several case studies and examples from the literature. Among them, we can mention: a flexible manufacturing system [11] that produces short-stroke cylinders; the alternating bit protocol, modeled with an un-timed PN in [12]; and a computer-assisted braking system for vehicles [13]. The integration of the ITPN bound solvers within the *DrawNET* environment, well as the implementation of the filters towards GreatSPN and TINA, has been considerably reduced the time devoted to the V&V activities of the considered case studies. New GUI facilities are going to be implemented that allows the analyst to visualize the net labels, parameters, timing specifications and results in the *DrawNET* model job window and to export the net to the SVG format. Possible future developments concern the implementation of new filters that enable to import, in the *ITPN-PerfBound* tool, the ITPN models specified either in the PNML standard format or in the GreatSPN format.

The tool is available at the following URL:

http://www.draw-net.com/ITPN-PerfBound.

References

1. Bernardi, S., Campos, J.: On Performance Bounds for Interval Time Petri Nets. In: Proceedings of the 1st International Conference on Quantitative Evaluation of Systems (QEST'04), Enschede, The Netherlands, pp. 50–59. IEEE Computer Society Press, Los Alamitos (2004)
2. Merlin, P., Faber, D.: Recoverability of communication protocols. IEEE Trans Commun. COM-24(9) (1976)
3. PetriNetsWorld: Petri nets tool database, `http://www.informatik.uni-hamburg.de/TGI/PetriNets/tools/db.html`
4. Berthomieu, B.: The TIme petri Net Analyzer (TINA) toolbox, `http://www.laas.fr/tina/`
5. Roux, O., et al.: The Roméo tool, `http://romeo.rts-software.org`
6. Vicario, E., et al.: The ORIS tool, `http://www.stlab.dsi.unifi.it/oris/index.html`
7. Gribaudo, M., Codetta Raiteri, D.G.F.: The DrawNET Modelling System: a framework for the design and the solution of single-formalism and multi-formalis models. Technical Report TR-INF-2006-01-UNIPMN (January 2006)
8. PerfGroup: The GreatSPN tool, `http://www.di.unito.it/~greatspn`
9. Billington, J., et al.: The Petri Net Markup Language. Standard ISO/IEC-15909-2
10. Berkelaar, M., et al.: LP_SOLVE: library for solving linear (integer) programming problems, `http://lpsolve.sourceforge.net/5.5/`
11. Paoli, A., Sartini, M., Tilli, A.: Rapid prototyping of logic control in industrial automation exploiting the generalized actuator approach. In: Proc. of 13th IEEE International Conference on Emerging Technologies and Factory Automation, Hamburg, Germany. IEEE, Los Alamitos (2008)
12. Diaz, M., Azema, P.: Petri net based models for the specification and validation of protocols. In: Rozenberg, G. (ed.) APN 1984. LNCS, vol. 188, pp. 101–121. Springer, Heidelberg (1985)
13. Bernardi, S., Campos, J., Merseguer, J.: Timing-failure risk assessment based on Petri net bounding techniques. Technical report, University of Torino, Italy (June 2008)