Petri Net-Based Optimization of Production Systems

Sz. Gyapay, A. Pataricza
Dept. Measurement and Inf. Systems
Budapest Univ. of Technology and Economics
H-1117 Budapest, Magyar tudósok körútja 2 Hungary
pataric@mit.bme.hu

J. Sziray
Department of Informatics
Széchenyi István University
H-9026 Győr, Hédervári út 3.
Hungary
sziray@sze.hu

F. Friedler
Department of Computer Science
University of Veszprém
H-8200 Veszprém, Egyetem u. 10.
Hungary
friedler@dcs.vein.hu

Abstract **– Petri nets are widely used as an underlying framework for formalizing UML-based system models. Based on their precise semantics and easy-to-understand graphical representation, they are appropriate to model production systems together with their quantitative properties.**
The production of desired materials can be modeled as a reachability problem of its Petri net model, which can be analyzed by linear algebraic techniques (solving linear inequality systems) providing a qualitative solution structure. However, traditional reachability analysis techniques can result in a state space explosion, while numerical methods for invariant computations give either sufficient or necessary conditions. The main advantage of Petri nets is the ability to describe complex and even adaptive control structures.
Another mathematical paradigm, Process Network Synthesis (PNS) algorithms are widely used in chemical engineering to estimate optimal resource allocation and scheduling in order to produce desired products from given raw materials.
In the current paper it will be shown that PNS algorithms that exploit the specific combinatorial features of PNS problems, can be applied to Petri nets in order to give more efficient mathematical methods for their analysis.
This combination of Petri nets and PNS algorithms can unify the efficiency of the optimization algorithms in the PNS fields with the power of Petri nets in modeling complex processes.
The current paper presents efficient semi-decision and optimization methods for the reachability problem based on PNS algorithms: Solution Structure Generation (SSG) and Accelerated Branch and Bound (ABB) algorithms. We show that the ABB algorithm can be used to solve scheduling problems efficiently and can be extended for other Petri net analysis. Obviously, the combined methodology can be used for a variety of application fields like IT systems, workflow optimization, etc., including production processes as a special case.

I. INTRODUCTION

Due to the increasing complexity of IT systems, the formal verification and validation of system models are required. Petri nets provide one of the most efficient tools to model parallel processes and high-level abstraction. Petri nets as specification formalism are widely used to model systems, especially concurrent and distributed systems: by means of places and tokens within the places nets can specify concurrency, sequential composition, mutual exclusion and non-determinism.

Since system models could be very large the traditional Petri nets analysis techniques such as reachability analysis based on the state graph can result in state space explosion.

Frequently, the objective of the reachability analysis is only the proof of the non-existence of an unwanted situation, for instance, a dangerous combination of the states. In such cases semidecisions try to prove the soundness of the system in an indirect way by showing that an unwanted situation would not fulfill a necessary condition at the system level. Obviously, if this proof is successful then it is an evidence of the non-existence of the unwanted case but if the proof is unsuccessful further analysis is needed. The main advantage of the semidecision-based techniques is that the necessary conditions are frequently easy to analyze abstractions of the original system behavior.

Recently, a number of semidecision algorithms using the results of linear algebra and integer programming have been developed [1,2,3]. On the one hand, these numerical methods are much more efficient (often with polynomial runtime) than traditional reachability methods. On the other hand, sacrificing the preciness of analysis they provide only either sufficient or necessary conditions for the reachability problem [4].

In the related topic production optimization, efficient algorithms were elaborated for Process Network Synthesis problems based on the reformulation of the problem to produce desired products from given raw materials by resource allocation.

Recently, first results of the process graph based algorithms have been exploited and applied in several fields of computer science (e.g., to the syndrome-decoding problem of diagnostics of multiprocessor systems [5]).

Based on the state equation, a necessary condition for the reachability of a particular state in the state space of the Petri net is provided by linear algebraic techniques: a chord connecting the start and the end points of the required trajectory is computed without any details about the real trajectory. Thus the non-reachability of a system state *could* be decided:

- if the state equation has no solution (there is no chord) then the state is not reachable
- otherwise the state *could* be reachable.

Formalizing the reachability problem of Petri nets as a mixed integer linear programming problem together with an appropriate target function (e.g., time or cost function), we propose a semidecision method based on Process Network Synthesis (PNS) algorithms.

After a short introduction to Petri nets (Section 2) and Process Network Synthesis (Section3), the theoretical relationship between Petri nets and P-graphs is discussed in Section 4. In Section 4.2, differences between Petri nets and PNS are described while Section 5 provides a Petri net example to demonstrate the adaptation of the Accelerated Branch and Bound algorithm. Finally, further research directions are discussed, i.e., the adaptability of PNS algorithms to other Petri net analysis.

II. PETRI NETS AND THE REACHABILITY PROBLEM

Before the introduction to the reachability problem, we give a short overview about Place/Transition nets.

A. *Place/Transition nets*

A P/T net (see Fig. 1.) is a 4-tuple $N = \langle P, T, E, W \rangle$ where P and T are disjoint sets of places and transitions, respectively. $E \subseteq (P \times T) \cup (T \times P)$ defines the edges from places to transitions and conversely, while $W : E \to naturals$ is a weight function on edges. Places can contain tokens and transitions can consume from and also produce tokens to places according to the weight function.

The input and output places or transitions of an element $a \in P \cup T$ are denoted by $\bullet a = \{b \mid (b,a) \in E\}$ and $a\bullet = \{b \mid (a,b) \in E\}$, respectively.

In other words, a *P/T* net is a directed bipartite graph, where the two disjoint node types are places and transitions, respectively, connected by directed and weighted arcs.

A M marking is a $|P|$ dimensioned vector over naturals composed of the numbers of tokens in the corresponding places. A given marking represents the state of the net which can be changed by firing transitions. A transition t can fire (is enabled) if each of its input places $\bullet t$ contains at least as many tokens as is specified by the weight function, i.e., $\forall p \in \bullet t : M(p) \geq W(p,t)$. The firing of a t transition removes the determined amount $W(p,t)$ of tokens from the input places, and the $W(t,p)$ tokens produced for the output places.

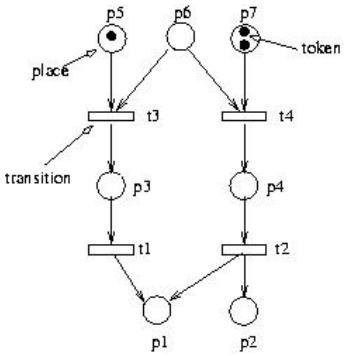

Fig. 1. An example P/T net

B. *Reachability Problem*

A sequence of firings describes the changes in the state of the Petri net along a particular trajectory. The *reachability problem* is to decide whether from an initial marking a given marking as a starting point there exists a legal trajectory to a target marking, i.e., whether there exists a firing sequence $t_1 \ldots t_n$ from the initial marking M_0 to the final marking M denoted by $M_0 \xrightarrow{t_1 \ldots t_n} M$.

The *transition vector* of a firing sequence is a $|T|$ dimensioned vector over naturals composed of the number of the occurrences of each transition in the sequence.

In case of a huge system model, the decision of the existence of such a trajectory by the construction of the state graph can result in state space explosion. The methods avoiding the state space explosion provide the transition vector describing the chord between the initial marking and the final marking.

Thus these methods produce only semidecision methods because they deliver either the answer ‘no’or ‘I do not know’ instead of a surely fireable trajectory. This way semidecisions are effective in excluding unwanted solutions.

The main difference between the firing sequence completely describing trajectory and the transition vector generated by the semidecision algorithm is that the latter one describes only the number of occurrences of the individual transitions and neglects their order.

The fireability of the resulted transition vectors is satisfied only in case of some specific Petri net subclasses, e.g., state machines. In case of general Petri nets, the output of the semidecision algorithms requires an additional filtering of the transition vectors.

III. PROCESS NETWORK SYNTHESIS

In chemical engineering, PNS algorithms are used to determine an optimal resource allocation and scheduling for the production of desired products from given raw materials.

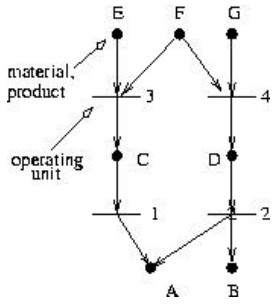

Fig. 2. An example P-graph representing a PNS problem

A. *Problem definition*

A PNS problem is represented by a so-called Process graph (shortly P-graph), which is a directed bipartite graph consisting of material containers and operating units. Operating units consume their input materials and produce their output materials while the directed arcs between materials and operating units symbolize the flow of materials in the network.

In addition, costs are assigned to raw materials and operating units and balance equations for material production. *The objective of a PNS problem is to find a solution structure of a minimal cost, i.e., a sub P-graph, which represents how final products can be efficiently synthesized from raw materials.* In the P-graph of Fig. 2., products A and B are to be synthesized from raw materials F and G by the four operating units.

B. PNS algorithms

By means of PNS algorithms, sufficient **and** necessary conditions for feasible solution structures are determined defining the entirely solution space. These requirements are given by five axioms [6] discussed shortly in the following describing the specific combinatorial properties required to have in order to be a feasible solution structure.

- if the state equation has no solution (there is no chord) then the state is not reachable
- raw materials must not be produced
- and all products have to be synthesized.

The efficient algorithms for PNS problems [7] exploit these combinatorial properties.

The solution space is reduced by the polynomial time *Maximal Structure Generation (MSG)* algorithm removing the redundant operating units and materials surely not playing any role in a feasible solution.

The *Solution Structure Generation (SSG)* algorithm generates all of the feasible shop recipes. Obviously, a combination of two shop recipes is a feasible solution as well.

The subsequent *Accelerated Branch and Bound algorithm (ABB)* estimating the optimal solution can deal with combinations of complex shop recipes instead of individual operations thus drastically reducing the search space. This reduction can reach 5-10 orders of magnitude even in the case of small scaled problems.

IV. UNIFYING CONCEPTS

Obviously, the combination of the Petri net based paradigm of modeling complex systems with the efficiency of the PNS approach in resource allocation and scheduling optimization promises an important synergetic effect for a wide class of problems.

Based on their semantic resemblance, a unified treatment can be defined for Petri nets and PNS problems. At first, the structural, then the semantic analogies are defined.

A. Joint features

As it is shown in the previous summary, the structural similarity between Petri nets and P-graphs is straightforward (TABLE I).

In the Petri net terminology, the five axioms of the PNS problem define a partial reachability problem where the initial state corresponding to the marked states representing raw materials and the designated end state has to cover all and only the places corresponding to products.

The restriction 'partial reachability' means that the final marking of all places except these specific output places does not care. Please note that in another interpretation if we use the notion of information and information processing steps instead of materials and physical operations the PNS problem can be applied for a large spectrum of partially observable IT systems.

TABLE I

THE ASSIGNMENT OF PNS AND PETRI NET ELEMENTS

P-graph of a PNS problem	Petri Nets
material containers	places
operating units	transitions
raw materials	corresponding places are marked at the initial marking
final products	corresponding places are marked at a final marking
cost of materials and products	cost of places (time)
cost of operating units	e.g., firing time of transitions
rate for operating units	weight of incoming and outcoming edges of the corresponding transition
material quantities	number of tokens

Our objective in merging Petri nets and PNS are the following ones:

- On the one hand, if traditional Petri nets are enriched by numerical cost functions and their objective function to be optimalized then we can use this methodology for searching the optimal solution of an arbitrary weighted reachability of the special structure described in Section 3.2.

Please note, that for a wide class of Petri net related quantitative problems the solution complexity is a major limiting factor to build faithful models. For instance, stochastic Petri nets can be currently solved in state spaces roughly of the size of $10^7 - 10^8$ only while PNS can cope with essentially larger problems.

- On the other hand, the extension of the traditional PNS approach by the feasible modeling power of Petri nets allows to model and solve highly complex production problems like those in logistics or in feasible manufacturing cells.

The PNS algorithms can be used for Petri net reachability analysis in three steps. First, the non-existence of the maximal structure means that there exists no trajectory from the initial marking, i.e., the final marking is not reachable. Second, algorithm SSG provides every feasible solution structure as a set of their contained operating units, i.e., the results are represented as characteristic vectors of the contained transitions of the trajectories in Petri nets.

Finally, algorithm ABB gives the optimal solution structure in form of a vector denoting the number of the use of the individual operating units, i.e., it provides the transition vector of a possible optimal trajectory of the corresponding Petri net.

B. Essential differences

Thereupon its chemical use, PNS problems can contain some structures which are numerically untreatable in case

of Petri nets. In the chemical production, a specific kind of materials is used: the catalyst.

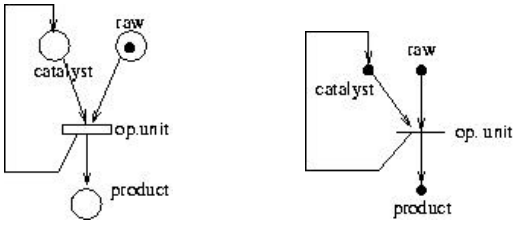

Fig. 3. Catalyst in Petri net (left) and in PNS (right)

In Fig. 3., a P-graph and a Petri net of a catalyst are shown not carrying the same meaning. For instance, whenever the production of a compound requires a catalyst it can be produced and taken for the production where the cost of the catalyst container can be the cost of the catalyst production. Meanwhile, a token in the graphically identical Petri net can not be produced if it represents a state variable which is unmarked in the initial state.

Obviously, cycles may frequentally contain siphons in Petri nets, i.e., such subnets, which remain unmarked if they do not contain a proper marking in the initial state. Therefore it is not assured that the optimal results determined by algorithm ABB are realizable, i.e., fireable from the initial marking.

Due to the above facts, for an arbitrary Petri net the satisfaction of the five axioms is necessary but not sufficient to represent a feasible solution net for the (partial) reachability problem. For instance, a net containing a catalyst place that is not reachable from the initial marking can be contained by the result net computed by PNS algorithms.

In the literature, some specific Petri net subclasses exist that always provide feasible solution nets in case of the satisfaction of the five axioms. Some of them are *state machines* and *sound workflow nets* [8].

In addition, a slight difference holds in the optimal solution computations. Algorithm ABB computes a solution and also together with the determination that the contained operating units how many materials produce that numbers can be even rational. In contrary, transition vectors have to contain integers, i.e., the numbers of the firings.

In the following, algorithm ABB is adapted to a scheduling problem represented by a Petri net.

V. AN EXAMPLE: ADAPTATION OF ABB ALGORITHM

Based on their similarity, algorithm ABB is adapted for an example Petri net. The model depicted in Fig. 4. shows the repair of computer types A and B. A mechanic can repair either one A or one B typed computer and even both of them AB at the same time. The task assignment and their repair take 1, 1, 2 and 2, 1, 3 time units, respectively.

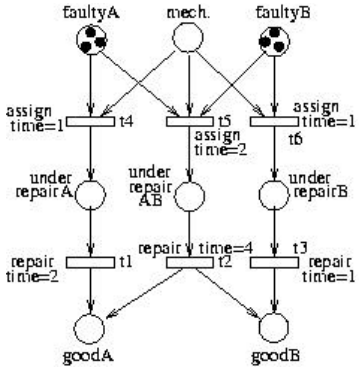

Fig. 4. An allocation problem: initial marking

In the initial state there are three-three faulty A and B typed computers, respectively. The objective is to find the optimal task allocation to the mechanics. thanks to a degree of freedom in the composition of the model we can introduce a constraint on the number of mechanics available to perform the repair job in a variety of forms.

- In the traditional Petri net style of modeling we can represent the number of mechanics available by a proper initial marking of the net and estimate the set of solutions by changing this initial marking. Obviously, this approach suffers from the problem originating in the pure scalability of the model. This way if the number of mechanics is changed then the reachability graph has to be repeatedly generated. (Its complexity can be slightly reduced only by identifying parts covered in the previous analysis. For instance, if we increase the number of the mechanics, all the solutions estimated in the previous case of a smaller number of mechanics remain feasible but the entire reachability graph has to be traversed in order to identify the new branches.)
- In the pure Petri net styled modeling we can associate a cost to hiring a new mechanic. In this case, this associated cost will held to keep the number of mechanics involved low but their number will be still influenced by the cost factors assigned to the other parts operating on the production flow.
- Finally, we can include a numeric constraint to the number of mechanics. This will confine the solution space in the optimization phase, moreover, it can be used to make cuts in the reachability graph. If these cuts are locked then the generation of the reachability tree for an increased number of mechanics can be done in an incremental way by extending the tree starting from the point where the cuts were made.

VI. PNS ALGORITHMS IN OTHER PETRI NET ANALYSIS TOOLS

Based on their resemblance, PNS algorithms are appropriate not only for reachability analysis but can be also extended for other Petri net analysis problems.

T-invariant In Petri net analysis, *T-invariants* play an important role. A T-invariant is a set of transitions (a transition vector) such that their firing do not change the

state of the net. The invariants are determined by the solution of a linear equation system. The lack of such a method for T-invariant computing is that it does not assure the fireability of the result transition vector while the use of algorithm ABB can provide a fireable transition vector if the net is appropriate.

Another point where the unified approach promises interesting results is the case when the production process underlies temporal constraints, i.e., if some production process has to precede another one. In Petri net domain, such problems occur frequently during validation and verification where the objective is to check the validity of a temporal constraint typically given in the form of a temporal logic expression.

The typical approach discussed below is to construct such a product *Büchi net* [9] automaton, which has as a solution space firing sequences satisfying both the logical constraints from the original structure and the temporal one.

Büchi net In more details, Büchi net is a product automaton of a Petri net of the modeled system (it is required to be a *1-safe* Petri net, i.e., for every place and reachable marking $M : M(p) \leq 1$ and the Büchi automaton of the negation of an LTL formula to be checked for the system. If the constructed net accepts no words then the formula holds for the system. Since the automaton is a net itself, having an initial state and accepting states it can be extended with source and sink transitions. If there is no T-invariant the automaton is empty.

T-invariant computation has a traditional solution algorithm introduced by Martinez and Silva [3]. The algorithm generates the basis containing the minimal T-invariants, i.e., other T-invariant can be easily computing from the basis elements. Since algorithm SSG generates *every* feasible solution network it is a redundant method. Based on the original method, algorithm SSG can be improved to provide the minimal feasible basis.

VII. UML-BASED MODEL GENERATION

One main technical disadvantage of both the PNS approach and Petri nets is that they require the construction of large or even extremely large models. This model construction is a main bottleneck both in the efficiency of the analysis and in the quality of the model. In order to avoid troubles induced by the manual creation of large scale models, we plan to automatically generate these models from UML models [10,11].

Our current analysis indicates that the standard UML profile intended to describe schedulability can be used without modifications for our problem field as well [12].

VIII. CONCLUSION

The discussed combination of the efficient PNS algorithms with the modeling power of Petri nets provides an efficient tool to improve the analysis of UML models. Based on their graphical and semantical similarity, PNS algorithms can be adapted to Petri net analysis and evaluation. The current paper examined their unified treatment: joint and different features and further application possibilities of PNS methods to Petri net analysis problems.

IX. ACKNOWLEDGMENT

This work was supported by projects FKFP 0193/1999 and OTKA T038027.

REFERENCES

[1] S. Melzer and J. Esparza, "Checking system properties via integer programming", in *European Symposium on Programming*, pp. 250-264.

[2] J. Desel, "Petrinetze, Lineare Algebra und lineare Programmierung", vol. 26. of Teubner-Texte zur Informatik, B. G. Teubner Stuttgart-Leipzig, 1998.

[3] M. Silva, E. Teurel and J. M. Colom, "Linear Algebraic and Linear Programming Techniques for the Analysis of Place/Transition Net Systems", *Lectures on Petri Nets I:Basic Models*, vol. 1791, Springer Verlag, 1998, pp. 309-373.

[4] A. Pataricza, "Semi-decisions in the Validation of Dependable Systems", in *Proceedings of IEEE DSN'01: The IEEE International Conference on Dependable Systems and Networks*, 2001, pp. 114-115.

[5] B. Polgár, Sz. Nováki, A. Pataricza and F.Friedler, "A Process-Graph Based Formulation of the Syndrome Decoding Problem", *IEEE DDECS*, 2001, pp. 267-272.

[6] F. Friedler, K. Tarjan, Y. W. Huang and L.T. Fan, "Graph-Theoretic Approach to Process Synthesis: Axioms and Theorems", *Chemical Engineering Science*, vol. 47(8), 1992, pp. 1973-1988.

[7] F. Friedler, J. B. Vajda, and L.T. Fan, "Algorithmic Approach to the Integration of Total Flowsheet Synthesis and Waste Minimization", in M. M. El-Halwagi and D. P. Petrides, editors, *Pollution Prevention via Process and Product Modifications*, vol. 90(303) of AIChE Symposium Series Of, 1995, pp. 86-87.

[8] W. van der Aalst, "Structural Characterizations of Sound Workflow Nets", *Computing Science Reports*, vol. 23, Eindhoven University of Technology, Eindhoven, 1996.

[9] J. Esparza and S. Melzer, "Model Checking {LTL} using Constraint Programming", in *Proceedings of Application and Theory of Petri Nets*, 1997.

[10] A. Pataricza, Gy. Csertán, Gy. Román, V. Keresztély and J. Sziray, "UML based control of industrial processes", *IEEE International Conference on Intelligent Engineering Systems*, 2000, pp. 49-52.

[11] A. Pataricza, Gy. Csertán, O. Dobán, A. Gábor and J. Sziray, "Process Modeling and Optimization in UML", IEEE International Conference on Intelligent Engineering Systems, 2001.

[12] "Response to the OMG RFP for Schedulability, Performance, and Time", Draft version 0.14. *Real-Time UML Submission Consortium*,1996.