Journal of Engineering and Applied Sciences 5 (6): 413-419, 2010
ISSN: 1816-949X

Using Metaheuristics and SPC in the Analysis of State Spaces of Petri Nets

Eleazar Jimenez Serrano
Department of Automotive Science, Grad. School of Integrated Frontier Sciences,
Kyushu University, 744 Motooka, Nishi-Ku, 819-0395 Fukuoka City, Japan

Abstract: Reachability problems of state spaces derived from Petri nets are mainly tackled through structure analysis of the network and state space analysis of the behavior of the network. Both types of analysis have been combined in order to cope with their limitations but still the state space explosion in big networks keeps the margin of impracticability large. Here we use simulation, the third type of analysis technique and present four partial exploration metaheuristic methods intended to explore only certain evolutions of the state space and find the searched state in the fastest possible way (pathwise). The methods adopt some fundaments from statistical process control and six sigma used in the manufacturing industry and the example presented is precisely for a manufacturing system.

Key words: Petri nets, state space, reachability, stochastic optimization, impracticability, limitation

INTRODUCTION

In this study we focus on how to cope with one of the main limitations in the analysis of reachability problems in models of system behavior called Petri Nets (PNs). Two types of analysis are mainly conducted by researchers: structure and state space analysis. Structure analysis includes structural properties, reduction techniques, traps and siphons properties. State space analysis includes state matrix equation, integer linear programming, places and transitions invariants. Conclusive research have been presented for general PNs but all of them struggling against the same practical limitation; the state space explosion.

On the other hand, one more type of analysis which is not so popular among PNs theoreticians exists simulation analysis. It has been mainly used among practitioners for validation, verification, quantitative evaluation and visualization of the behavior of the network and scarcely used for solving reachability problems. The reason is because despite it uses the same state matrix equation as in state space analysis, different rules for solving conflicts in the network are implemented in order to obtain results on case by case basis, severely conditioning the generalization of those simulation is time-consuming and cannot cover all possible scenarios in the mathematical sense (Ciardo, 2004; Jensen and Kristensen, 2009). The reachability problem is described as: is the marking m_r reachable from the marking m_0 i.e., is there an occurrence sequence starting from m_0 which leads to the marking m_r?

Analysis of the reachability problem conducted by the state matrix equation requires the exhaustive exploration of the entire state space and then the construction of its reachability graph. A marking m_r is reachable from a marking m_0 if and only if there exists a path in the reachability graph from the node representing m_0 to the node m_r, assuming $m_0 \neq m_r$. However, working with reachability graphs is difficult if not impossible when we face the state space explosion.

This limitation has been tackled by means of conducting only a partial exploration of the state space using incomplete or partial graphs and estimated information in order to come to a conclusion. We classify partial exploration methods in two. The first methods still explore the entire state space but focus on how to avoid using a lot of memory by efficiently managing partial information, like coverability graphs, sweep line exploration and symbolic state space generation.

The second methods focus on stochastic optimization; they incorporate probabilistic elements in the problem data in the algorithm itself or in both. In particular, we will focus in a branch of these methods called metaheuristics (Luke, 2010). These last ones do not explore the entire state space, omit to visit all states, ignore which portion of the state space has been explored, produce unpredicted wrong results and have polynomial processing time but hold the possibility of not finding the solution even if it exists. The intention of this study is to present four partial exploration methods for the analysis of reachability problems based on metaheuristics. Implementing an implosive and sorted exploration

approach, we intent to visit only certain states and find the target marking if it exists by exploring fewer markings than the conventional searching algorithm or stopping the exploration at certain conditions when the probability of finding the target state is low. The methods are implosive because they takes advantage of the concurrent property of untimed PNs to avoid visiting some intermediate markings and sorted because the methods decide the direction of the exploration process based on some characteristics of the current marking, the set of enabled transitions and the target marking.

The main point about the metaheuristics is the assumption about the cardinality of all markings in the state space and the termination of the method based on Statistical Process Control (SPC) and 6σ. In it, we presume a normal distribution and use some of its parameters to skip certain exploration directions or stop the search. The methods have proved effectiveness in finding the target marking without exploring the entire state space (Serrano, 2010) but failed in selecting the shortest path. In this study, we show that the combination of two of the methods can reach the target marking and obtain a shorter path.

PETRI NETS

Let, P and T be finite and nonempty disjoint sets and $F \subseteq (P\times T) \cup (T \times P)$. An ordinary Petri net Σ is a tuple (P, T, I^-, I^+, Q) a finite bipartite directed graph where $P = \{p_1, ..., p_a\}$ is a finite and non-empty set of a places, $T = \{t_1, ..., t_b\}$ is a finite and non-empty set of b transitions with $(P \cap T = \varnothing)$, $I^-: (P \times T) \rightarrow \{0,1\}$ and $I^+: (T \times P) \rightarrow \{0, 1\}$ are the backward and forward incidence functions and Q is a place capacity function $P \rightarrow Z^+ \cup \{\infty\}$. A marking in the net is a function $M: P \rightarrow Z^+ \cup \{\infty\}$. We say, there are k tokens in a place p if m(p) = k. The initial marking in the net is defined as m_0 and is usually s.t. card $(m_0)>0$, i.e., the cardinality function (the sum of all tokens in all place) at the initial marking is >0. And for convenience, the number of tokens in a place p of a marking m will be described using the bag-like notation e.g., m = (104) is to $m = 1p_1+4p_3$.

A transition t_e is enabled at a marking m if $\forall p \in {}^\bullet t_e$, $M(p)\geq I^-(p,t_e)$ and $\forall p\in t_e^\bullet$, $M(p) + I^+(t_e,p)\leq Q(p)$. The set E(m) contains all enabled transitions t_e at the marking m and the powerset $E^*(m)$, contains all subsets e of E(m) (all combinations of enabled transitions at a marking m). The occurrence (firing) of an enabled transition removes $I^-(p, t_e)$ tokens and produces new tokens s.t. $\forall p \in t_e^\bullet$, $M(p) = I^+(p, t_e)$. For any set $e \in E^*(m)$, the notation $m[e\rangle m'$ will mean that all enabled transition t_e in e may fire at m yielding m' and e^- is its representative firing vector. The set $R_e(m_0)$ contains all immediate reachable markings produced from each string $e \in E^*(m)$. The state matrix equation is such that $m' = m + e^-\cdot(I^+ - I^-)$. The vector e^- is a nonnegative integer solution of the equation if it yields to m' and then e is a valid string.

The union of valid strings e is called a firing sequence σ and γ is the sum of all firing vectors (Parikh vector). The set $R(m_0)$ contains all possible consecutive reachable markings from $m_0[\sigma\rangle$. A set $R(m_0)$ of markings is linear if any marking produced is equal to $c\cdot m_0$ with $c\geq 0$ and semilinear if it is the union of a finite number of linear sets.

Enabled transitions t_e, t_e' are parallel if $\exists m\in R(m_0)$ s.t. $\{t_e, t_e'\}\in E(m)$ and ${}^\bullet t_e \cap {}^\bullet t_e' = \varnothing$, in conflict if $\neq \varnothing$ and the string notation $[t_e, t_e']$ means their firing is simultaneous. A path is a sequence of elements of P and T starting from an initial marking (m_0), reaching a marking (m_r) through a series of firings, i.e., $m_0 \rightarrow \sigma_0 \rightarrow m_1 \rightarrow \ldots \sigma_{r-1} \rightarrow m_r$. A path between markings m_0 and m_r is the shortest path if the number of all markings in between is minimum. We point at Murata (1989) for other descriptions about Petri nets.

DISTRIBUTION OF TOKENS AND MARKINGS

Tokens are active entities in a Petri net system. The token game determines the number of tokens to put and eliminate in every place and their distribution give place to every marking. We intent to measure the tokens difference among any given marking m (including m_0) and a target marking m_t because we are interested in how to fulfill the tokens distribution and difference (tokens quota) to reach m_t. We start presenting the first main assumption in the analysis of reachability problems for a system (Σ, m_0) and a target marking m_t: the cardinalities of all markings in its state space appear like a probability density function with normal distribution, mean equal to the cardinality of the target marking and standard deviation equal to the absolute value of the difference between the cardinality of the target marking and the cardinality of the initial marking (Fig. 1).

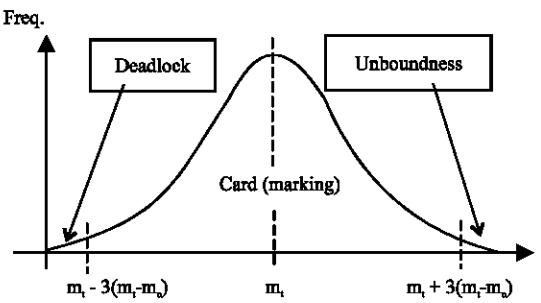

Fig. 1: Distribution of the cardinalities of all marking

When the difference between both cardinalities is zero, the default value is one. And for the case when card $(m_t) - 3\|\text{card}(m_t) - \text{card}(m_0)\| < 1$, then the cardinalities of all markings in a state space appear like a probability density function with normal distribution and left-side trimmed at one which limits having zero and negative markings (Fig. 2).

Binary distance: To establish a point of convergence for the state progression going from any marking (including m_0) towards the target marking m_t, we define the Binary Distance (BD) between any two given markings m and m_t as:

$$BD(m, m_t) = \sum_{i=1..a} \|m(p_i) - m_t(p_i)\| \quad (1)$$

The Polarized Binary Distance (PBD) is the sorting element which supports the proposal. It is obtained by removing the absolute value from the Eq. 1. Its interpretation is naively given as:

- Any positively PBD shows a surplus quota of tokens in m with respect to m_t
- Any negatively PBD shows a deficit quota of tokens in m with respect to m_t
- A PBD = 0 and BD ≠ 0 means tokens in m ≡ m_t (they are not aligned). We refer to this as quota sigma
- A PBD = 0 and BD = 0 means m = m_t. We refer to this as quota zero

Understanding the tokens quota between m and m_t allow us to select among all subsets of $E^*(m)$ the firing vectors which could produce the quota sigma or even better the quota zero. Consider the system in Fig. 3 with card $(m_0) = 2$, card $(m_r) = 3$ and the set $E(m) = \{t_1, t_2, t_3, t_4\}$. From the set E*(m), the subsets $\{t_1, t_4\}$ and $\{t_2\}$ evaluate BD and PBD equal to zero.

Concurrency in untimed Petri nets: Untimed PNs is a general description for any type of PNs without the dependency of time. The lack of this association permits the theoretical analysis of any unrestricted behavior: sequential, concurrent, distributed, synchronic, etc. In this study, we intent to use it for reducing the number of explored states in reachability problems. Consider the system in Fig. 4 with card $(m_0) = 2$, $m_t = 2p_5$ and the set $E(m) = \{t_1, t_2, t_3\}$. From the set $E^*(m)$ only the subset $\{t_2, t_3\}$ gives a negative integer solution. Three different firing sequences are observed in order to reach m_t: $t_1t_3t_3$, $t_3t_1t_3$, $[t_1t_3]t_3$.

The subsets $\{t_1\}$, $\{t_3\}$ and $\{t_1, t_3\}$ give different nonnegative integer solution with marking's cardinality of two but conditioned to explore at least one more marking before the target marking. The sequence $m_0 \rightarrow t_1 \rightarrow m_1 \rightarrow t_3 \rightarrow m_2 \rightarrow t_3 \rightarrow m_t$ visits two intermediate markings as well as the sequence $m_0 \rightarrow t_3 \rightarrow m_3 \rightarrow t_1 \rightarrow m_2 \rightarrow t_3 \rightarrow m_t$. Since, we want to reduce as many as possible the number of explored markings only the last subset visits one marking before reaching m_t, i.e., $m_0 \rightarrow [t_1t_3]\ m_2 \rightarrow t_3 \rightarrow m_t$.

This indicates that sorting of firing vectors alone is not enough, also the proper selection based on the firing result is important. And one more final though is that the subset $\{t_1, t_2, t_3\}$ which gives a nonnegative integer solution is another example on how to take advantage of concurrency. Let us consider the case where $m_r = 3$. We are looking for firing sequence in any of these forms: $t_1t_2t_3$,

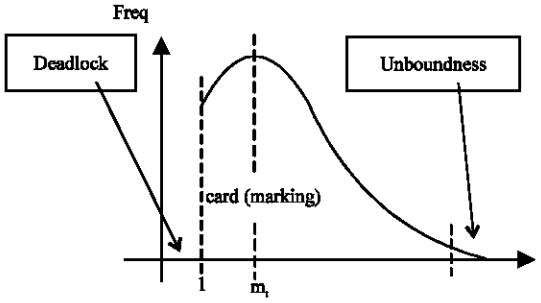

Fig. 2: Left-side trimmed distribution

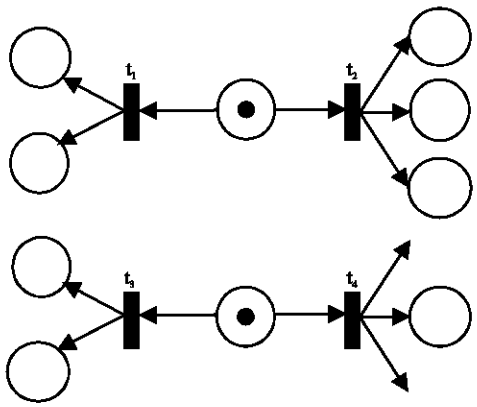

Fig. 3: PN system with four enable transitions

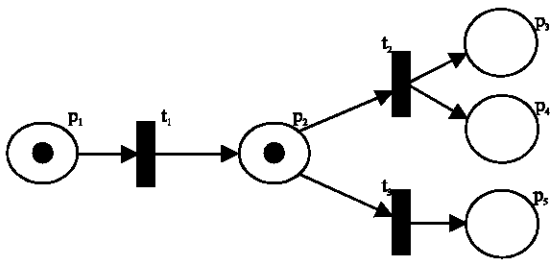

Fig. 4: PN system with three enable transitions

$t_1t_3t_2$, $t_1[t_2t_3]$, $t_2t_1t_3$, $t_3t_1t_2$. From the set $E^*(m)$, the subset $\{t_1, t_2, t_3\}$ not only gives the desired solution but also is the one that does not visit any other intermediate markings.

METAHEURISTICS

Summarizing the purpose of this study, we want to minimize the number of explored states which are not in the shortest path of states between m_0 and m_t (avoid total exploration) i.e.,

$$\text{Min } Z = \phi + \gamma + \tau$$

Where:

ϕ = The number of states in the shortest path
γ = The number of states not in the shortest path
τ = The number of states visited more than once

For this problem, we observe that not choosing firing vectors that produce markings with the same cardinality of the target marking is critical and that linearity in markings also plays an important role in rapidly reaching the target marking.

And when m_t is not a linear marking of m_0, stochastic methods are an alternative worth to explore. Metaheuristics is a combinatorial optimization method intended to stochastically optimize problems. It employs some degree of randomness to find optimal (or as optimal as possible) solutions to hard problems by iteratively trying to improve a candidate solution with regard to a given metric (Ciardo, 2004).

These methods make few or no assumptions about the problem being optimized and can explore large state spaces of candidate solutions. However, metaheuristics do not guarantee an optimal solution is ever found. Metaheuristics will be used here to optimize the search of a target marking m_t if it exists, starting from an initial marking m_0.

Sorting

Problem 1: For a marking m and its set of all enabled transitions $E^*(m)$, find a valid string e such that $m[e\rangle m_r$ where $m_r = m_t$ or at least $m_r = m_t$. For this problem, we define the following generalized sorting pseudo-code:

- Generate the set $E^*(m)$
- Obtain the set $R_e(m) = \{m_r \in R(m_0) \mid m[e\rangle m_r \; \forall e \in E^*(m)\}$
- Delete from $R_e(m)$ any marking $m_r < 0$
- Let set $C_p(m) = \{x \in \mathbf{Z}^+ \mid x = \text{card}(m_r) \; \forall m_r \in R_e(m)\}$
- Rest $\text{card}(m_t)$ to every $x \in C_p(m)$
- Create List : $E^* \rightarrow \|C_p\|$

The first metaheuristic is that if there is a valid string e which produces $m_r = m_t$ or at least $m_r = m_t$, it will be the in list such that list (e) = 0.

Selecting

Problem 2: For a marking m, a sorted list of valid strings $e \in E^*(m)$ and a subset $S \subseteq E^*(m)$ with $m[s\rangle m_r$, $s \in S$ and $m_r = m_t$, find s such that the number of intermediate markings is minimum. Let us start with the description of the diamond rule: if $m_0 \rightarrow t_1t_2 \rightarrow m_r$ and $m_0 \rightarrow t_2t_1 \rightarrow m_r$ then $m_r = m_r$. There is a sorted sublist list containing only the relations of $S \rightarrow \|C_p\|$. Based on the diamond list and the basic addition of firing vectors, the following instruction must be performed to list:

- Let set $C_t(S) = \{y \in \mathbf{Z}^+ \mid y = \text{card}(s) \; \forall s \in S\}$
- Create list' : $S \rightarrow \|C_t\|$
- Sort list' in descending order (starting from the highest)

The valid string s with firing vector having the highest cardinality is the second metaheuristic approximation to the reachability problem. However, another possible perspective to the problem 2 is given as follows.

Problem 3: Finding s such that the number of places marked in m_r which are also marked in m_t is maximum. This problem involves a more detailed vision on a place-by-place basis. The following instructions must be performed to list:

- Let set $C_s(m) = \{z \in \mathbf{Z}^+ \mid \text{numel}(m_r(p_i)\text{-}m_t(p_i) = 0), I = 1 \ldots a\}$
- Create list' : $S \rightarrow \|C_s\|$
- Sort list' in ascending order (starting from the lowest)

This second valid string s with the firing vector producing a marking m_r with tokens in as many places as possible as in the marking m_t is the third metaheuristic approximation to the reachability problem.

SPC AND 6σ

We make use of the already known Statistical Process Control (SPC) methods used in the manufacture industry for the control of the searching method. A vast literature exists about SPC methods therefore, we will briefly introduce only the concepts used in this research; the reader can check (Stapenhurst, 2005) or any other available literature on SPC and 6σ for a deeper understanding.

Exploring: For the partial exploration, in this sstudt we discuss the following generalized pseudo-code for state space exploration having the depth-first search algorithm as the pillar of the pseudo-code.

```
  let stack.n(1) = 1, stack.m(1) = m0, stack.l(1) = 1;
  let I = 1, j = 1;
let c_node = stack.n(i), c_mrk = stack.m(i), c_e = stack.l(j);
       while c_node>0 do {
         if c_e>0 then {
            select list'(c_e);
            calculate new_mrk;
            if new_mrk is in stack.m then
               put new_mrk in stack.m as "not_new";
   else { put new_mrk in stack.m as "new";
        let c_mrk = new_mrk; }}
  else {        // case c_e = 0
     if prev_node>0 then {
        if exist stack.n = prev_node with stack.n>0 then
           let c_mrk = found_mark;
         else let c_node = 0; }
       else let c_node = 0; }
   }endwhile.
```

In addition to the new methods with sorted lists of firing vectors, there are particular features in the algorithm which are important to recall.

Enabling of transitions: Petri nets are used to model systems in various number of application areas and levels of abstraction. How likely, it is that transitions occur as singletons or concurrently cannot be quantified without considering the application area and the level of abstraction on which a Petri net is built. In the exhaustive exploration of state space of Petri nets, it is folk knowledge to use the single-transition firing rule. The reason is because every marking that can be reached by all simultaneously enabled transitions can as well be reached through a sequence of single transition occurrences (Roch and Schmidt, 2006). And while some researchers propose to forbid structures that lead to markings only reachable through concurrent firings only the modeler can judge whether these markings belong to realistic scenarios (Roch and Schmidt, 2006). Nevertheless, despite both scenarios are realizable after the implementation of proper control extensions (Darondeau *et al*., 2008) (like controlled petri nets) for the exploration process we allow both types of firings.

Returning search point and interrupting the exploring: Another way to classify state space exploration methods is as event based (enabling and firing of transitions) and state based (marking in places). The method we are presenting in this study falls in the second category since, we focus on how to reach a tokens quota. For what is next, we will use the tokens count to determine when to return to a previous node to continue the exploration or to stop it. In order to avoid the total exploration of the state space, we determine the termination of the method given that it is improbable to find the target marking in the state space with the marking results we are obtaining.

Proposition 1: If $\forall e \in E^*(m)$ and $\forall m \in R(m_0)$ the firing of any vector e^- cannot produce as many tokens as the cardinality of the target marking then the probability of finding the target marking is zero.

Problem 4: Find the cardinality of a marking in the state space $R(m_0)$ for which the probability of finding a target marking m_t is lower than α. In statistics, the 68-95-99.7 rule states that for a normal distribution, nearly all values lie within 3 standard deviations of the mean. About 68.27% of the values lie within $\mu\pm\sigma$, 95.45% within $\mu\pm2\sigma$ and 99.73% within $\mu\pm3\sigma$. Taken from SPC, the thresholds at which any production process output is considered statistically unlikely are drawn typically at 3 standard deviations from a center line established from the process capability (in the case from the cardinality of the target marking). Then, the probability α of finding the target marking within $\mu\pm3\sigma$ is 99.73%. For the searching process, the calculation of the exploration limits is defined slightly different since, we only have the initial and target market information in the form of binary distance.

Upper Limit (UL): We define this limit as the unboundness level, the point where the number of tokens might growth continually and uniformly. It is calculated as three times the binary distance (between the target and initial marking) over the target marking.

Lower Limit (LL): We call this limit as the deadlock level, the point where the number of tokens might turn into zero. It is calculated as three times the binary distance (between the target and initial marking) under the target marking. The default value of the lower limit is one when the calculation gives a value below one. Consider the following state space reachability problem with card $(m_0) = 10$ and card $(m_t) = 12$.

The cardinalities of all markings appear like a probability density function with normal distribution, mean $\mu = 12$ and standard deviation $\sigma = 2$. The upper and lower limits are 18 and 6 tokens, respectively. This apparent distribution is seen in the Fig. 5. For this problem, we define that the probability of finding the target marking when firing vectors produce markings with cardinality between 6 and 18 tokens is 99.73%. Now based on the previously explained limits, we stop the exploring process at a marking m and return to the immediate predecessor marking (if the stack is not

empty) when the next pulled firing vector e^- from list (m) reaches a marking m_r such that: LL<card(m_r)<UL.

Visualizing the exploration: We adopt run-chart (or control chart) from SPC to graphically observe the results of the exploration process which results very convenient for depth-first searches. Another purpose will be to visually recognize the behavior and results of the exploration process and the state space markings and to make hypothesis about the structural characteristics of the net.

Example: The following example is a Flexible Manufacturing System (FMS) taken to demonstrate the methods (Fig. 6). The initial marking is $m_0 = 2p_1+p_7+p_{11}+p_{13}+p_{19}$ with card (m_0) = 6. A target marking was fixed at $m_t = p_5+p_6+p_7+p_{11}+p_{17}+p_{19}$ with card (m_t) = 6. The apparent normal distribution of the cardinality of all markings has mean σ = 6 and a default standard deviation σ = 1. We are looking for a firing sequence achieving the shortest path to the target marking with only seven markings:

$$m_0 \rightarrow [t_1t_{12}] \rightarrow m_1 \rightarrow [t_1t_2t_{13}] \rightarrow m_2 \rightarrow [t_3t_{14}] \rightarrow m_3 \rightarrow [t_2t_4t_6] \rightarrow m_4 \rightarrow [t_3] \rightarrow m_5 \rightarrow [t_6] \rightarrow m_t$$

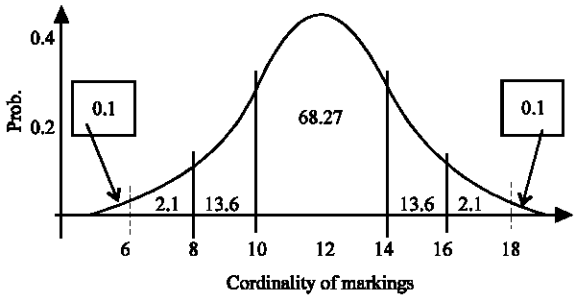

Fig. 5: Apparent distribution of markings' cardinality

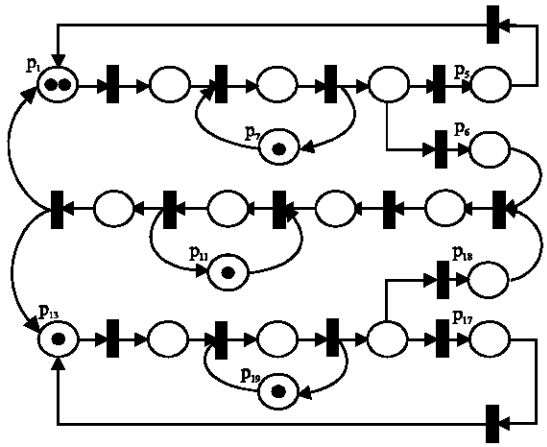

Fig. 6: Petri net system of a FMS

in order to compare its result with the methods. The second includes the basic sorting of the firing vectors.

First four different simulations were performed in order to reach the target marking. In the first one, we only perform a target search with the pure depth-first algorithm The next one is the selection of the vector according to the problem two. The third one includes the selection by the problem three. The results of the simulations are shown in Table 1.

From the simulation results we can observe that all the three methods exceed the results of the pure depth-first algorithm. The reason is their capability to engage concurrent firings to reach the target marking in a faster way.

However, the methods still fail in finding the shortest path for reaching the target marking; the obtained paths. In the run-chart of the simulation of pure depth-first algorithm (Fig. 7), we can observe several characteristics about the cardinality of the markings of the state space, like boundness (the model seems bounded to six tokens). In order to improve the exploration method, a particular

Table 1: Results of the simulations

Total exploration	Target search
Pure depth-first	
37.589119 sec	2.077347 sec
379 registers	72 registers
144 markings	55 markings
235 repeated	17 repeated
Sorted depth-first	
92.506132 sec	0.135850 sec
793 registers	13 registers
144 markings	13 markings
649 repeated	0 repeated
Selective-1 depth-first	
78.894180 sec	0.955007 sec
793 registers	24 registers
144 markings	21 markings
649 repeated	3 repeated
Selective-2 depth-first	
94.099431 sec	0.148233 sec
793 registers	13 registers
144 markings	13 markings
649 repeated	0 repeated

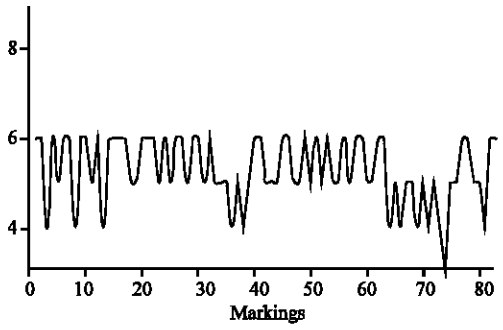

Fig. 7: Run-char of the FMS

Table 2: Results of the combined method

Total exploration	Target search
Combined depth-first	
92.152343 sec	0.133688 sec
793 registers	12 registers
144 markings	12 markings
649 repeated	0 repeated

combination of the three methods was prepared. Its result was slightly superior to any of the best results obtained with an individual method of the Table 2. Its superiority based on the detection of completed token quotas and weight of the firing vectors is sufficient to believe it can find shorter paths than any of the individual methods.

CONCLUSION

We presented four metaheuristics for the analysis of reachability problems in order to minimize the number of unnecessary explored markings. We observed that a combination of >1 metaheuritic is necessary in order to overcome difficulties when selecting the best firing vector. Due to the information about the required firing sequence is never known in advance, how to take advantage of concurrent behavior based on heuristic results will be studied in the future in order to obtain a better metaheuristic for the reachability analysis problem. The line of study will be in the feedback of the cardinality of found markings in the exploration process in order to adjust the specification of the upper and lower limit. The second and main line will focus on the combined metaheuristic method and its usability in more general Petri nets.

One more thing is that after seen the results from the different simulations, we observe that the proposals could work better only for the analysis or reachability problems and not for total exploration. And despite the main purpose of the methods was achieved, it is the reduction of the number of states explored and registered, the processing time seems to be compromised as the size of the models increase. Finally, we believe the searching method might work better for state spaces belonging to real-life systems, like a FMS and not for those belonging to random graphs. The reason is because state spaces which are not from random graphs have several typical properties which are specific to their structure, like normal distribution of the cardinality of all markings, making them more suitable for model checking algorithms (Roch and Schmidt, 2006).

REFERENCES

Ciardo, G., 2004. Reachability set generation for petri nets: Can brute force be smart. Proceedings of the 25th International Conference on Applications and Theory of Petri Nets, June 21-24, Bologna, Italy, pp: 17-34.

Darondeau, P., M. Koutny, M. Pietkiewicz-Koutny and A. Yakovlev, 2008. Synthesis of nets with step firing policies. Appl. Theory Petri Nets, 5062/2008: 112-131.

Jensen, K. and L.M. Kristensen, 2009. Coloured Petri Nets: Modeling and Validation of Concurrent Systems. Springer-Verlag, Berlin, Heidelberg,.

Luke, S., 2010. Essentials of Metaheuristics. Department of Computer Science, George Mason University, USA.

Murata, T., 1989. Petri nets: Properties, analysis and applications. Proc. IEEE, 77: 541-580.

Roch, S. and K. Schmidt, 2006. On the step explosion problem. Lecture Notes Comput. Sci., 4024/2006: 342-361.

Serrano, E.J., 2010. A probability-based state space analysis of petri nets. IEICE Tech. Rep., 110: 7-12.

Stapenhurst, T., 2005. Mastering Statistical Process Control. Elsevier Ltd., UK., pp: 460.