Acta et Coomentationes, Exact and Natural Sciences, nr. 2(8), 2019 ISSN 2537-6284
p. 103-110 E-ISSN 2587-3644

CZU: 004.02 DOI: 10.36120/2587-3644.v8i2.103-110

APPLICATION OF PETRI NETS IN LOGICAL PROBLEMS SOLVING

Inga ȚIȚCHIEV, associate professor
Tiraspol State University
Vladimir Andrunchievici Institute of Mathematics and Computer Science

Abstract. The main goal of the article is to perform theoretical research with practical applicability that will contribute to solving some mathematical logical problems by applying the formalism of Petri nets. The facilities offers a new approach that allows the solution to be determined by a method different from the existing ones, but which is quite suggestive and easy to understand.

Keywords: logical problem, modeling, Petri nets, reachability graph.

APLICAREA REȚELELOR PETRI LA REZOLVAREA PROBLEMELOR DE LOGICĂ

Rezumat. Scopul principal al articolului este axat pe efectuarea cercetărilor teoretice cu aplicabilitate practică care vor contribui la soluționarea unor probleme de logică matematică prin aplicarea formalismului Rețelelor Petri. Facilitățile acestora oferă o nouă abordare care permite determinarea soluției printr-o metodă diferită de cele existente, dar care este sugestivă și ușor de înțeles.

Cuvinte cheie: problemă de logică, modelare, rețele Petri, graf de acoperire.

Introduction

The development of information technologies, formal methods and means of communication have contributed to the development in a very fast time of the tools that can be applied to complex processing and which lead to the occurrence of new approaches in problems solving.

In this research, the formalism of Petri nets was proposed for solving some logical problems. The solution is possible by applying the verifying methods of their properties, as for example the well-known logical problem with wolf, goat and cabbage is used (LCV).

The formalism of the Petri nets [6] allows representing graphically the modelled systems and also offers the possibility of analyzing their dynamic properties. One of the most important things is that a model of a system to function properly, qualitative (or behavioural) properties must be determined mainly. Another important aspect is the fulfilment of certain associated performance characteristics (or quantitative properties).

Petri nets

Petri nets were introduced by Carl Adam Petri in the 1960s, at that time the mathematical models used to model the distributed systems were the state-action transitional systems (finite automata). Starting from these models C.A. Petri introduced the idea of modelling the distributed systems, dividing the system into certain elements that would characterize the local states of the modelled system and characterizing the evolution of the system through a concurrent execution of some local actions. They formalize the description of competition, conflict and synchronization in distributed systems in an inductive way.

Definition 1. *A Petri net* is a structure $\Sigma = (P, T, F, K, W, M_0)$, where

- P is a finite set of elements called *places*.
- T is a finite set of elements called *transitions* ($P \cap T = \varnothing$;).
- $F \subseteq (P \times T) \cup (T \times P)$, is called the *flow relations* (the set of arcs).
- K is a function $P \to N \cup \{\infty\}$, called the *capacity function* (*K(p)* it is called *the capacity* of the element $p \in P$. If $K(p) = \infty$, then we will say that p has *infinite* or *unlimited capacity*).
- W is a function $F \to N$, called the *weight function* (*W(f)* it is called *the weight of the element* $f \in F$).
- $M : P \to N$ is the *marking function* (M_0 is the *initial marking*, or the state from which the system study begins).

The modelling of the systems distributed through the Petri nets is performed at the state level: it determines which actions occur in the system, which states precede these actions and in which states the system will be after the actions have been produced. Simulating the model of states through Petri nets gives a description of the behaviour of the system.

Petri nets currently have many applications and are used in various fields as engineering or business process modelling, operating systems, formal languages, logic programming, industrial control systems, parallel programming, multiprocessor systems, neural networks, decision models because they have a very accessible graphical representation and have a well-defined semantics that allows a formal analysis of the behaviour and properties of the modelled systems.

Petri nets [1] allow the verification of properties at the design stage, by applying methods, already considered as classical for them, such as *reachability graph*. This method will be applied to determine the solution of the LCV problem in a safe and easy to understand manner.

Reachability and coverability graphs

Reachability graphs are structures obtained from the accessible markings of the Petri net from which additional information about some properties of the modeled systems can be extracted. These, in some cases, can be infinite and then it is practically impossible to verify there properties, in this case from the reachability graph is obtained the coverability graph, from which also useful information about the properties of the modeled system can be extracted, but which are finite already and in which the information about some legacies or repetitions that are determined is not lost.

Definition 2. Let be $\Sigma = (P, T, F, W, M_0)$ a marked Petri net. *The coverability graph* is called any graph $\Im_\Sigma = (V, E, l_V, l_E)$, which satisfies the following properties:

(i) $l_V : V \to M$; $l_E : E \to T$.

(ii) The root v_0 of the $\Im_\Sigma$ it is labeled with M_0: $l_V(v_0) = M_0$;

(iii) for any node v with label M ($l_V(v) = M$) it takes place:

a. $|v+| = 0$ (v is a leaf node), if it does not exist $t \in T$ such that $M[t\rangle$ or exist $v' \in d(v_0, v)$, $v \neq v'$ with label M' such that $M = M'$;

b. $|v+| = |\{t \in T | M[t\rangle\}|$, otherwise;

(iv) for any node $v \in V$ with $|v+| > 0$, $l_V(v) = M$ and any $t \in T$: $M[t\rangle$, $v' \in V$ exist, $l_V(v') = M'$ such that:

a. $(v, v') \in E$;

b. $l_E(v, v') = t$;

c. Either $M[t\rangle M'''$. For any $p \in P$ it take place:

i. $M'(p) = \omega$, if exist $v'' \in d_\Im(v_0, v)$ such that $l_V(v'') = M''$, $M'' \leq M'''$ și $M''(p) < M'''(p)$;

ii. $M'(p) = M'''(p)$, otherwise.

Using the coverability graphs such properties as: boundedness, pseudo-viability, viability, reversibility, blockage, ***accessibility*** can be checked.

Wolf, goat and cabbage problem

In [1] the folowing statement of the problem was proposed:

A man carrying a wolf, a goat and a cabbage comes to a river, which he must cross with a narrow otter. How will he do it, knowing that:

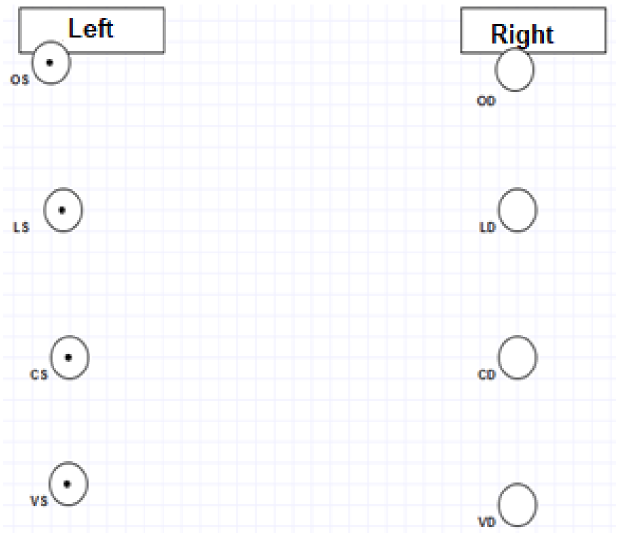
- *the wolf can eats the goat and the goat can eats the cabbage;*
- *man cannot pass them all once or in pairs;*
- *the number of crossings must be minimal.*

Figure 1 illustrate the problem [2].

Figure 1. Illustration of LCV problem

In order to solve the LCV problem, several preparatory modeling stages of all processes that may occur are required.

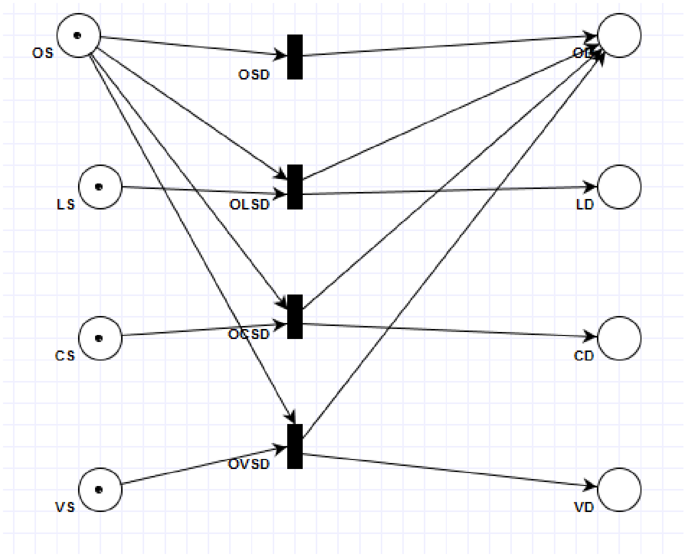
First stage. Modeling of the initial state (when the man, the wolf, the goat and the cabbage are on the left side) and the final state (when the man, the wolf, the goat and the cabbage are on the right side), which would allow the problem to be successfully solved. For this we will use eight places that have the meaning shown next to Figure 1.



OS – the man is on the left side of the river;
LS – the wolf is on the left side of the river;
CS - the goat is on the left side of the river;
VS - the cabbage is on the left side of the river;
OD – the man is on the right side of the river;
LD – the wolf is on the right side of the river;
CD - the goat is on the right side of the river;
VD - the cabbage is on the right side of the river.

Figure 2. Initial and final state of the LCV problem modelled by Petri nets

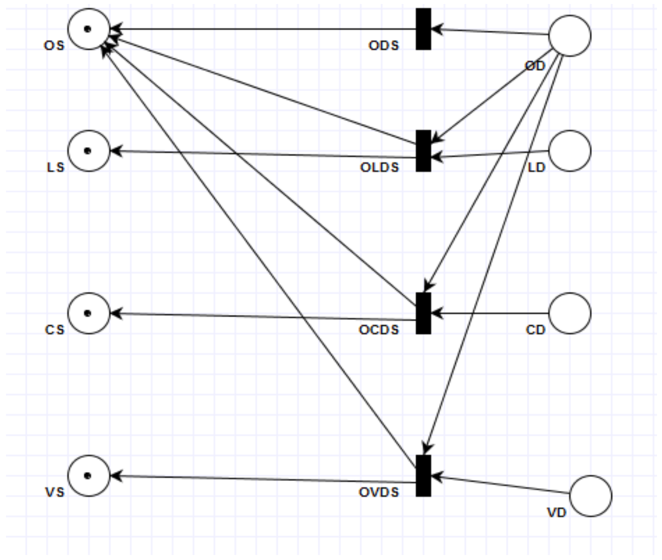
Second stage. The second stage in solving the problem involves modelling the actions that allow crossing from one side to the other one, but only by two actors involved in the crossing process. For this, the actions in Figure 2 are added, the meaning of which is given along with Figure 3 and Figure 4.



OSD – the man crosses from the left to the right side of the river;
OLSD – the man and the wolf cross from the left side to the right side of the river;
OCSD – the man and the goat cross from the left side to the right side of the river;
OVSD – the man and cabbage cross from the left side to the right side of the river.

Figure 3. Actions modelling that allow to cross the river from the left side to the right side

The situation when crossing the river from the right side to the left side is modelled by the below Petri net:



ODS – the man crosses from the right to the left side of the river;
OLDS – the man and the wolf cross from the right side to the left side of the river;
OCDS – the man and the goat cross from the right side to the left side of the river;
OVDS – man and cabbage cross from the right side to the left side of the river.

Figure 4. Actions modelling that allow to cross the river from the right side to the left side

The third stage. The next stage in problem solving involves modelling of the critical situations in which the wolf eats the goat as long as they are on the left side of the river and the man is on the right side and vice versa, or the goat eats the cabbage as long as they are on the left side of the river and the man is on the right side and vice versa, they are modeled by four transitions whose meaning is given in the below Petri net:

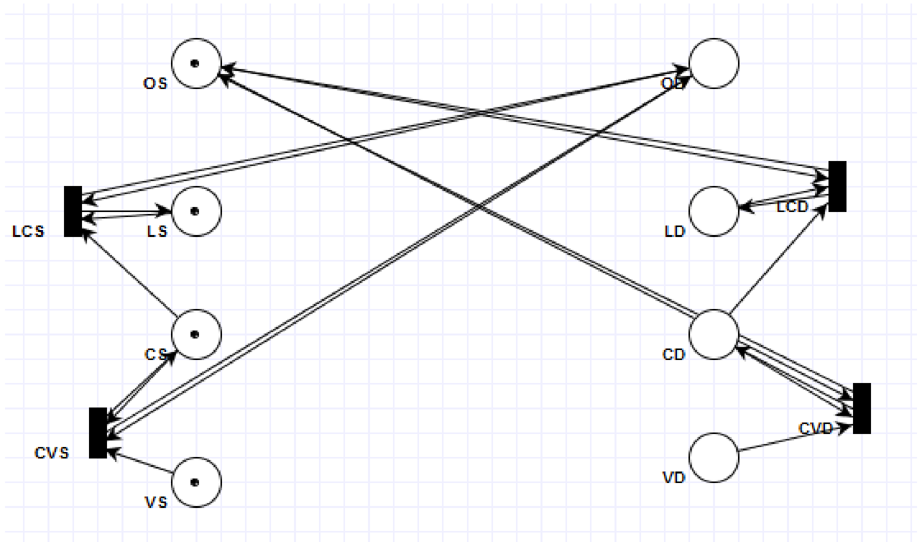


Figure 5. Actions modelling that represents critical situations

LCS - the wolf eats the goat as long as they are on the left side of the river, and the man is on the right side;
CVS - the goat eats cabbage as long as they are on the left side of the river, and the man is on the right side;
LCD - the wolf eats the goat as long as they are on the right side of the river, and the man is on the left side;
CVD - the goat eats cabbage as long as they are on the right side of the river, and the man is on the left side.

The fourth stage. The last stage in the problem modelling consists in combining the previous steps and thus we obtain the final Petri net which represents the LCV problem:

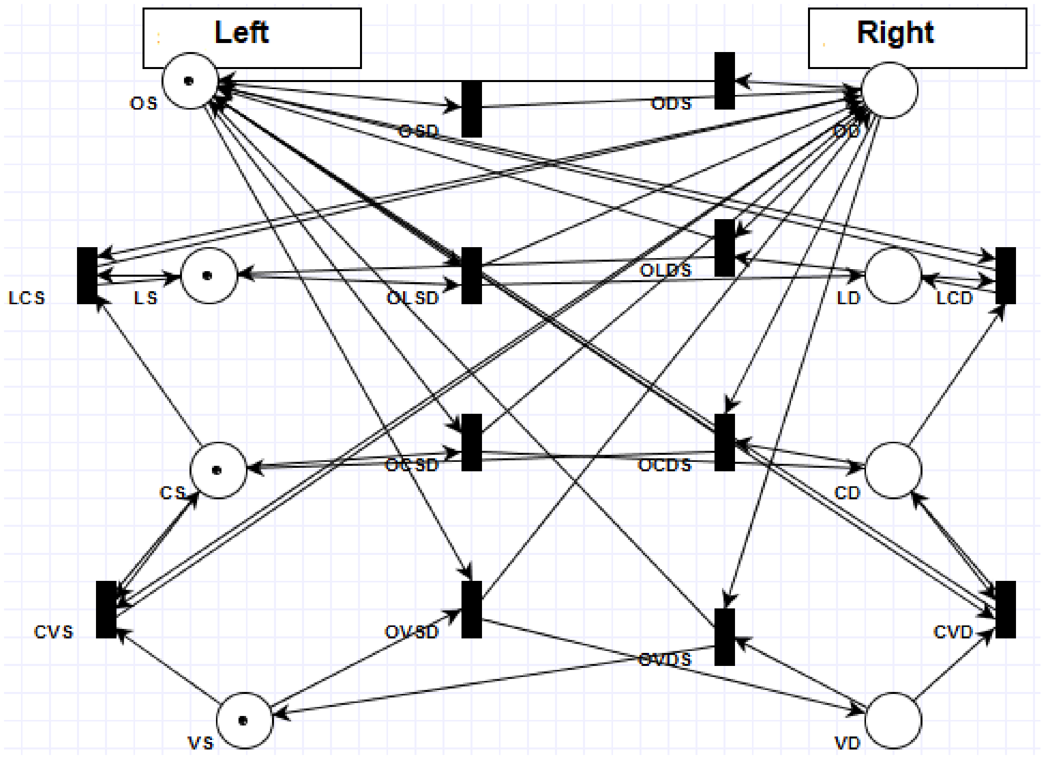


Figure 6. The LCV problem modelled by Petri net

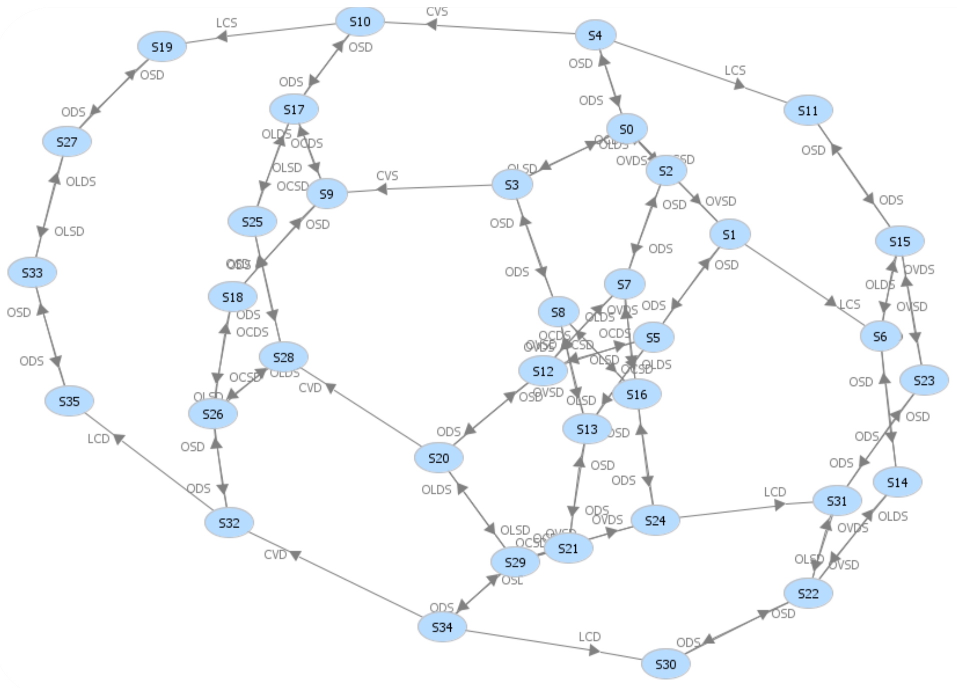


Figure 7. The coverability graph of the LCV problem

After obtaining the final Petri net we can apply the methods of the properties verifications to find the solution, so in this case we will apply the coverability graph method that allows us to determine all the possible transitions and from them to identify for the given problem a path that lead from the initial state of the system (in which all the involved actors, i.e. the man, the wolf, the goat and the cabbage are on the left side of the river), to a final state which represents the situation when all the involved actors of the problem have successfully crossed from the left side of the river to the right one. At the same time, we will follow as on this road the critical actions that lead to the unsuccessfully solution of the problem will not appeared. For this we will use the pipe simulator, which allows us to obtain the coverability graph and extract the necessary information from it. After simulation the coverability graph from Figure 7 was obtained.

From the obtained coverability graph it is needed to extract the information that would solve the LCV problem, this is possible if in the given graph we will exclude the critical actions that are on the way from the initial state $S_0=(1,1,1,1,0,0,0,0)$ to the final state $S_{29}=(0,0,0,0,1,1,1,1)$, i.e. LCD, LCS, CVD, CVS should not be appear on this road. Thus we obtain the following results:

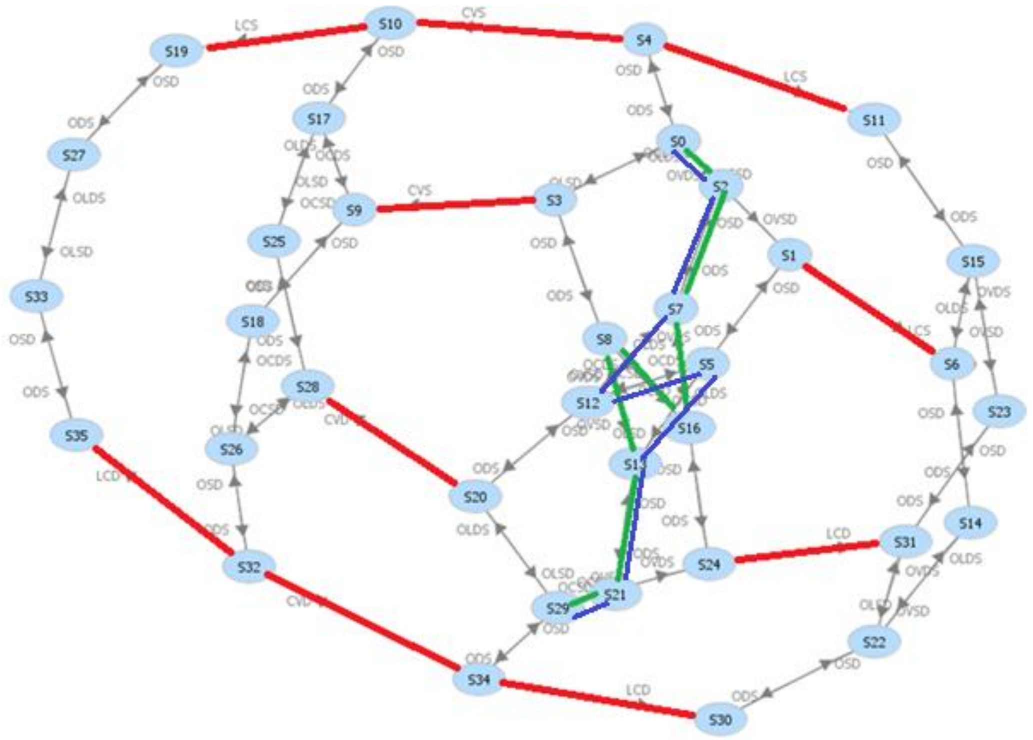


Figure 8. The coverability graph that models the LCV problem with the indication of the determined roads

In Figure 8 was indicated, by thickening the actions that was excluded (with red) and respectively the obtained roads:

OCSD-ODS-OLSD-OCDS-OVSD-ODS-OCSD,

OCSD-ODS-OVSD-OCDS-OLSD-ODS-OCSD,

which also represent the solutions of the problem, consisting of 7 steps:

1. OCSD – the man and the goat cross from the left side to the right side of the river;
2. ODS – the man crosses from the right side to the left side of the river;
3. OLSD – the man and the wolf cross from the left side to the right side of the river;

4. OCSD – the man and the goat cross from the left side to the right side of the river;
5. OVSD – man and cabbage cross from the left side to the right side of the river;
6. ODS – the man crosses from the right side to the left side of the river;
7. OCSD – the man and the goat cross from the left side to the right side of the river.

Thus, we obtained the successful crossing from the left side of the river S0=(1,1,1,1,0,0,0,0) to the right side of the river S29=(0,0,0,0,1,1,1,1) of the four involved actors man, wolf, goat and cabbage and thus the proposed problem was solved.

Conclusions

For the logical problem with the wolf, the goat and the cabbage were proposed the formalism of the Petri nets that allows to obtain its solutions by applying the method of the coverability graph from which we extract it. This approach is suggestive, easy to understand and can be applied to other logical problems from the same classes.

Bibliography

1. The statement of the problem - http://www.logicus.ro/index.php/numarare-si-distribuire/214-lupul-capra-si-varza, vizited 05.07.2019.
2. Wuf.ro: Wolf, Goat and Cabbage online game (in romanian). https://www.youtube.com/watch?v=7UHLvYKVt7g, vizited 10.07.2019.
3. Peterson J. L. Petri Net Theory and the Modelling of Systems. Prentice Hall, 1981.
4. Akharware N. Pipe2: Platform Independent Petri Net Editor. [Online], 2005. Available:http: //pipe2.sourceforge.net/documents/PIPE2-Report-20050814.pdf.
5. Camerzan I. Reachability graph for Petri nets (in romanian). In: Analele Universităţii de Stat din Tiraspol. 2002, v. III, pp. 93- 100.
6. Camerzan I. Verification of system nets. In: International Conference on „Microelectronics and Computer Science”, october 1-3, Chisinau, 2009. pp. 280-282.