FORMAL MODELING AND VERIFICATION OF MULTI-AGENTS SYSTEM USING WELL-FORMED NETS

Meriem Taibi[1] and Malika Ioualalen[1]

[1]LSI - USTHB - BP 32, El-Alia, Bab-Ezzouar, 16111 - Alger, Algerie
taibi,ioualalen@lsi-usthb.dz

ABSTRACT

Multi-agent systems are asynchronous and distributed computer systems. These characteristics make them also a discrete-event dynamic system. It is, therefore, important to analyze the behavior of such systems to ensure that they terminate correctly and satisfy other important properties. This paper presents a formal modeling and analysis of MAS, based on Well-formed Nets, in order to ensure the absence of any undesired or unexpected behavior. To validate our contribution, we consider the timetable problem, which is a multi-agent resource allocation problem.

KEYWORDS

Multi-agent system, Well-formed Nets, Model Checking.

1. INTRODUCTION

The paradigm of multi-agent systems (MAS) [1] offers an original way of modeling complex system. Therefore, multi-agent systems have been used in several areas, such as telecommunications, finance, Internet, energy, health, embedded systems ... etc. When designing MAS, it is often hard to guarantee the system specifications that have been designed, actually fulfil the requirements, i.e., whether it satisfies the design requirements, especially for critical applications. Tests and simulations have contributed for a long time to validate such systems. However, these techniques allow to investigate only one part of the global behavior. Thus, they differ from the formal verification techniques, which ensure that a property is verified by all possible system executions [2]. Consequently, it becomes crucial to have rigorous methods of formal specification and verification to ensure the safe development of agent based systems. These systems can be critical with no risk of error for some properties, such as security, integrity and robustness. Model checking techniques are widely used in analyzing MASs due to their completeness and automaton [3].

We have already presented in previous works an e-commerce multi-agent system modeling using Colored Petri Nets [4] [5], where some general properties verification was performed using CPN Tools [6] .Unfortunately CPN Tools does not allow verification of specific properties. In addition it suffers from the so called state explosion problem: the number of states in the model grows exponentially. In this paper, we present an efficient formal approach for modeling and verifying multi-agent systems, based on Well-formed Nets (WN) and model checking verification using GreatSPN [7] and SPOT [8] tools. The main advantage of Well Formed Nets is the notion of symbolic reachability graph that is composed of symbolic states. A symbolic state is a state representing several concrete states in the state space of the system described by the Petri net. So, much larger state spaces can be represented. Indeed, we present the Agent by class of color and his actions by transition associated by one or by several conditions.

We study in particular the interaction protocols. Interaction protocols enable agents to reach a solution in a quicker way. The agents know the messages they can receive in a given state, the message they can send and the rules that guide their choice in case of non-determinism. The agents thus go faster towards solution. As case study, we take FIPA contract net protocol applied to timetable problem. The timetabling problem is a resource allocation problem. It aims at finding an appropriate timetable for a set of courses to be scheduled within limited resources such as professors, student groups and class time.

There are generally two types of constraints in timetabling: hard and soft constraints. Hard constraints are those that must be satisfied and cannot be violated. For example, a professor can't give two courses at the same time to two different groups. Soft constraints are those that are preferably satisfied, but may be relaxed if necessary in order to meet hard constraints. For reasons of simplification we are interested by the verification of the hard properties which are expressed by the temporal logic used in this work. We begin by defining some atomic propositions that will help us to translate timetabling properties into LTL formula.

This paper is organized as follows. Related work are presented in Section 2. Section 3 details the analysed system and the proposed models. Section 4 describes the verification of the desirable properties and experimental results. Finally, we conclude our paper by giving some perspectives in Section 5.

2. RELATED WORK

Petri Nets (PN) have been successfully used in several areas for the modeling and analysis of distributed systems [9]. Several studies have been proposed to model MAS with Petri Nets. Balague [10], proposed a model for a promotional game of viral marketing on the Internet. She used Stochastic Petri Nets for modeling a multi-agent wish list. Gazdare [11] used Colored Petri Nets (CPN) as a formal method to model a transport MAS with containers, then, simulated and solved the storage problem. Lyu [12] used a Stochastic Petri Net (SPN) model to assess survivability and fault tolerance of mobile agents systems. They use the model for design and evaluation of their proposed agent architecture through simulation.

EL Fallah-Seghrouchni [13], Boukredera [14] and Khosravifar [15] proposed to use the CPN formalism to model interaction protocols. These Petri Net-based approaches provided a MAS's specification to facilitate applications design and implementation. However, they did not address the verification problem of the proposed models. The advantages of having a Petri net model were not exploited. The work presented by Hsieh in [16] proposed a new model called a collaborative Petri net and addressed the question of deadlock and undesirable state avoidance under the contract net protocol. Other Petri Net extensions were proposed in more recent works. [17] defined nested predicate transition nets to analyze multi-agent system and a set of translation rules that translate the multi-agent model to an executable PROMELA model [18]. Marzougui in [19] proposed an Agent Petri Net to model interactions between agents. The transformation of the obtained model, in an ordinary Petri Net, is also required to analyze the behavioral properties of the system.

Recently, model checking techniques are widely used in analyzing MASs due to their completeness and automaton. So several model checker are proposed for modelling and verifying critical properties of MASs, e.g., MCMAS [20], MCK[21], SPIN [2] and NuSMV [20]. These approaches, however, still have some limitations. Specifically, MCMAS and MCK mainly focus on concurrent systems without stochastic behaviors, which limits their application in unreliable environments or agents with random behaviors. In our work, we use a formal model, based on Well formed Nets, a class of high level Petri Nets, allowing qualitative analysis together with performance evaluation. This special class of high level Petri nets, allows to express symmetrical behaviours, which generates more compact state space.

3. MODELING MULTI-AGENT INTERACTIONS USING WN

An agent is an active and autonomous entity, it perceives its environment and interacts with other agents to achieve its goal.

The communication between agents can be structured by the use of protocols, structured descriptions of possible interactions between two or more agents. Protocols are a formalization of processes, which allow the organization of recurring tasks.

Several protocols have been proposed (see for instance the proposal of FIPA[1]).

3.1 FIPA contract net protocol

The contract net protocol [22] is an elementary protocol that facilitates task allocation between a group of agents' roles. In this protocol, there are two different types of roles, an Initiator and a Participant. The finite automata in Fig.1 and Fig. 2 model the different states and transitions of these roles. The Interaction Protocol is composed of a sequence of four main steps, illustrated by the sequence diagrams shown in Fig. 3. The agents must go through the following loop of steps to negotiate each contract.

1. The Initiator sends a "call for proposal" (CFP).

2. Participants who receive the announcement can answer by either a *Proposal* or *reject*.

3. Initiator receives and evaluates proposals, it sends a *Contract* to participant agents, whose proposals are accepted, and *Refuse* to other agents.

4. At the end of interaction, the participant sends to the initiator agent, an *Inform message* to confirm the action achieving, or a *failure message* in a failure case.

[1]FIPA: The Foundation for Intelligent Physical Agents

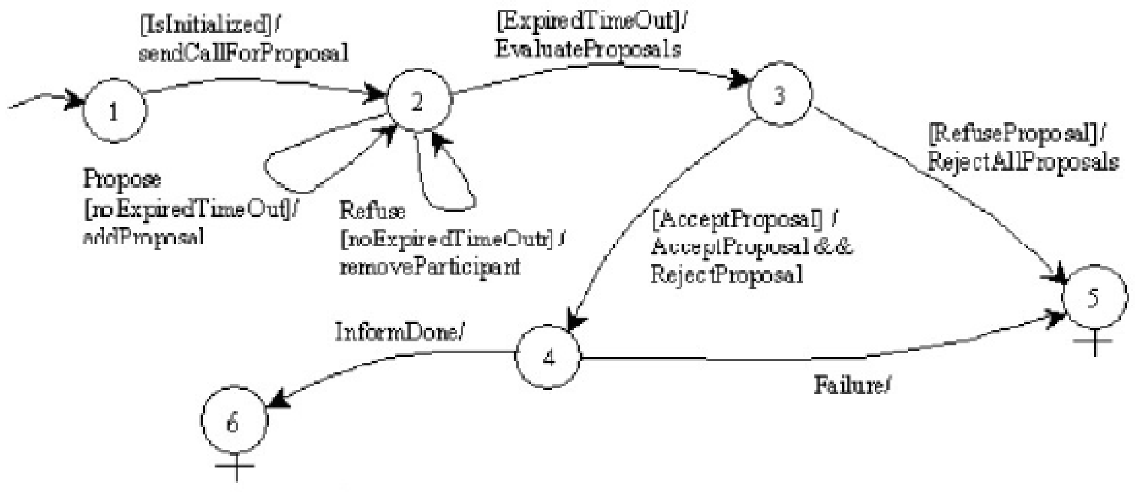


Fig. 1. Initiator automaton

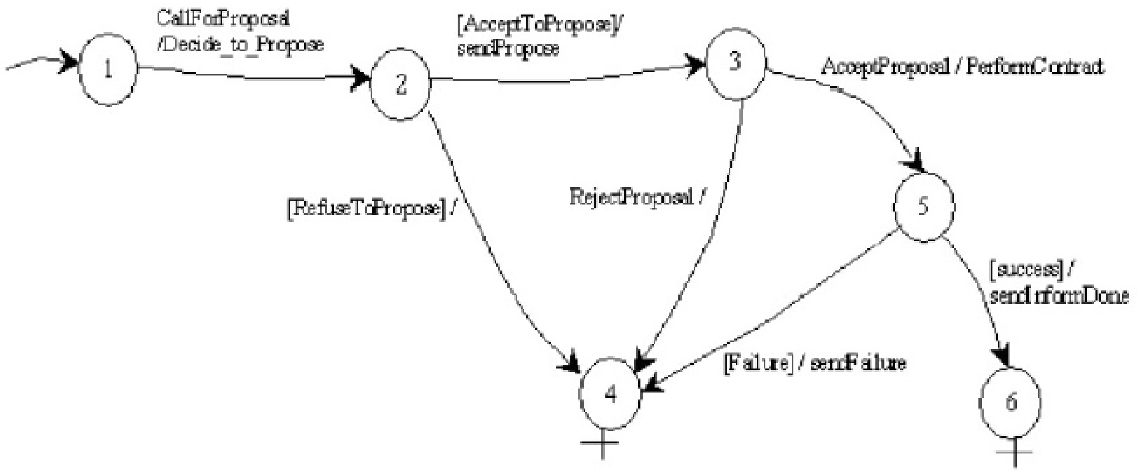


Fig. 2. Participant automaton

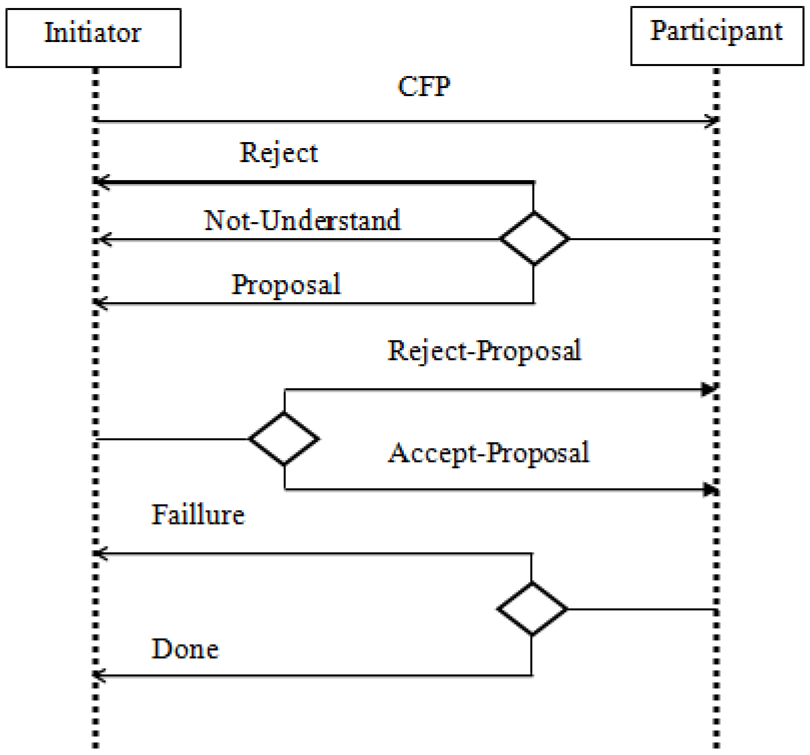


Fig. 3. FIPA Contract-Net Sequence Diagram

3.2 The Well formed Petri Nets (WN)

As mentioned in the introduction, our method is based on the Well-Formed Petri nets. Petri Nets are state based models which are well known for being able to model complex systems with concurrency and conflicts, even in the stochastic context. Moreover, WN model can also take advantage of behaviourial symmetries of systems' entities, if there are such symmetries. Finally, WNs are a well studied class of high level colored Petri nets and benefit from a large set of analysis algorithms and tools

A Well-formed Net [23] is a colored Petri net, where places and transitions are provided with a structured type of tokens. In this model, tokens are grouped into basic classes called color classes. These classes are brought together to form a color domain, which is associated to places and transitions. Colors of a place label its tokens, whereas colors of a transition define possible firings of the transition. Thus, an initial marking of a place is defined as a multiset (bag) of colored tokens. A color function is attached to each arc: its role is to define for, a given color of the associated transition, the number of colored tokens to add or to remove from the attached place.

A color domain is a Cartesian product of color classes. A total order, expressed by a successor function, can be defined on a color class. The Cartesian product defining a color domain can be empty (for example, in the case of a place containing neutral tokens). It can also contain repetition of a class (modelling internal synchronization of this class). A color class, grouping colors of same nature (eg. processes, resources), can be divided into static sub-classes, where a sub-class contains colors with identical behaviours, even in terms of performance.

A color function is built from standard operations (linear combination, composition, etc) of basic functions. The projection (denoted by X or X_i^j in figures) selects an element of a tuple; it is represented by a typed variable or by X if no confusion is possible. The synchronization/diffusion (denoted by Si or Si, k) returns the set of all colors of a class (Si) or a sub-class (Si, k). The successor function is defined for ordered classes only and returns the color following a given color.

A transition or an arc function can be guarded by an expression which is a linear combination of atomic predicates. An atomic predicate expresses the equality of two variables, or restricts the color domain of a variable to a static subclass. A predicate is evaluated on colors of a transition firing.

The structured definition of a WN allows us to exploit automatically system symmetries, by compacting its reachability graph, leading to a Symbolic Reachability Graph (SRG). An SRG is composed of symbolic markings, where each symbolic marking represents a set of ordinary (colored) markings having equivalent behaviours. Several qualitative properties can be checked on the SRG (reachability of a marking, deadlock freeness, etc.)

Formally, a well-formed Petri Net N is a tuple
$(P, T, C, cd, Pre, Post, Inh, Guard, Pri, M_0)$ with [23]:

- P,T: the finite sets of places and transitions,
- $C = C_i / i \in I = 1, , n$: the set of basic color classes; C_i is possibly partitioned into n_i static sub-classes: $C_i = \bigcup_{j=1}^{n_i}$,
- $cd : P \bigcup T \rightarrow Bag(I).cd(r) = C_1^{e_1} \times C_2^{e_2} \times \cdots \times C_n^{e_n}$ is the color domain of a node r; $e_i \in IN$ is the number of occurrences of C_i in the color domain of r, where $Bag(I)$ is the set of multisets (bags) on I.
- $Pre, Post, Inh$: the input, output and inhibition standard color functions from $C(t)$ to $Bag(C(p))$.
- $Guard(t) : C(t) \rightarrow true, false$ is a standard predicate associated with the transition t. By default, $Guard(t)$ is the constant function of value $True$.
- $Pri : T \rightarrow IN$ the priority function. By default, we assume $\forall t \in T, Pri(t) = 0$;
- $M0 : M0(p) \in Bag(C(p))$ is the initial marking of p.

3.3 Case study: Timetabling management benchmark

The Timetabling management helps users (teachers and student groups) to set their course sessions. Each user has an agent assistant who manages his schedule. The problem is described by:

– Teacher agent: The goal of such agent is to ensure teacher load, by fixing hours and courses he must have with all classes. The teacher agent has to instantiate and activate the initiator role of FIPA Contract Net, to determine every hour of its timetable, and hence his behavior can be described by the following rules:

  - If the teaching load is not empty, then we activate the Initiator role.
  - If the teaching load is empty, we suspend the agent teacher.

– Student-group agent: The student-group agents need to define their sessions with teachers. For their part, the student-groups must answer the teachers requests, so they instantiate and activate the participant role in the FIPA Contract Net protocol.

Our modeling, using WN, is essentially based on the description given in the section 3.1. Well formed Nets offer powerful expression as Colored Petri Net and also reduction characteristics to construct symbolic state space. The model is composed of transitions, places and arcs, indeed we present an agent by a token and its actions by transitions:

Structural representation

– Places represent agents states (before and after sending or receiving operations). Places contain also the different exchanged messages.
– Transitions allow to express sending and receiving messages or some other processing actions.
– Tokens express the different roles and the various exchanged messages.
– Incoming arcs labels specify data required for firing the associated transition.
– Outgoing arcs labels specify data produced by a firing transitions.

Token coloration To differentiate tokens, we use four color classes:

– The color class Te defines the set of teachers agents,
– The color class Cl defines the set of classes or student-groups,
– The class Hr defines the different time-slots (hours). To simplify the management, we consider that the time-slots have the same duration (2 hours for example).
– The color class Cr allows to definite the different courses, offered by teachers or asked by students.
  We associate to each place the color domain hat can mark it, and with each transition the color domain for which it is fired.

Description of models The global MAS model is shown in Fig. 4. Initially classes and teachers agents are inactive in *Classes* and *CFP* places respectively. Each teacher agent sends a call for proposal message to all classes agents by the *Broadcast* transition. One call is composed of sender (x), receiver (y) and proposed course (a).

Once the call is received by class agent, two cases can be distinguished:

- The proposed course is different from the requested one, in this case the offer of the teacher agent is rejected via the transition *Send-Reject*.
- The proposed course is asked by agent class, in this case, a *Proposal* is sent to teacher through the transition *Send-Prop*.

the proposal is treated according to two other cases:

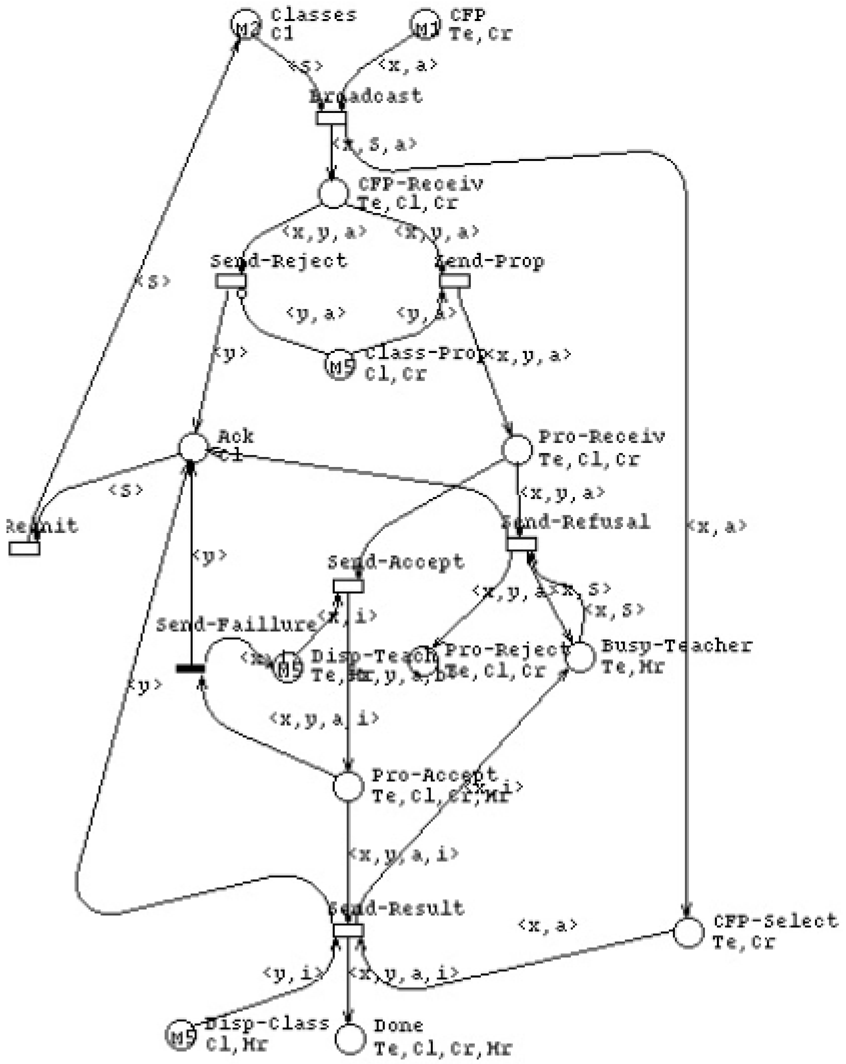


Fig. 4. The MAS model

- Teacher has no availability, in this case a *Refusal* is sent (*Send-Refusal* transition).
- The teacher answers by an acceptance and proposes a time-slot (i) to class agent ((*Send-Accept* transition)). If the student-group has the same availability, the interaction will end by a success and an affectation is created in the place (*Done*). Otherwise the interaction ends by a failure (*Send-Failure* transition).

4. Experimental Results

In this section, we check first some properties and then we analyse the obtained numerical results.

4.1 Properties verification

As discussed earlier, we use SPOT model checker to verify some properties, We begin by defining some atomic propositions that will help us to translate timetabling properties into LTL.

- P_1: the teacher t_1 is assigned to the group g_1 for the time slot h.
- P_2: the teacher t_1 is assigned to the group g_2 for the time slot$h$.
- P_3: the teacher t_2 is assigned to the group g_1 for the time slot h.
- P_4: the teacher t_1 receives more then proposal for his call for the course a.
- P_5: the teacher t_1 accepts one proposal for the course a.

Then the properties can be written down as follows:
Absence of conflict A situation of conflict is detected when the same group is assigned to two different teachers in the same time-slot or when the same teacher is assigned to two different groups in the same hour. These two properties are expressed respectively by the following expression, F_1 and F_2:

$$F_1 : G(!(P_1 \wedge P_3))$$
$$F_2 : G(!(P_1 \wedge P_2))$$

This means that the place *Done* is never marked with two tokens as (t_1, g_1, c_1, h) and (t_2, g_1, c_2, h) (resp. (t_1, g_1, c_1, h) and (t_1, g_2, c_2, h)) in the same time.

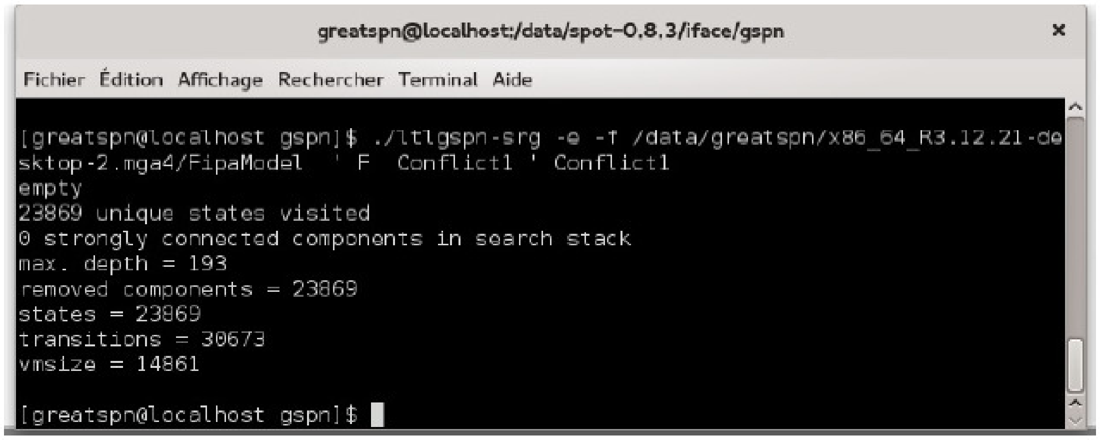
Absence of deadlock: A deadlock is detected when there are some offers for the same CFP and none of them will be accepted. The absence of deadlock is expressed as follows:

$$F_3 : G((P_4 \Rightarrow P_5))$$

This property may simply be expressed as: if the place $Pro - Receiv$ is marked by at least two tokens, then the place $Pro - Accept$ will also be marked.

After expressing properties, their verification is split as follows:
(a) Computation of the state space,
(b) Translation of the negated formula (the properties are expressed in LTL),
(c) Synchronized product of the former two objects,
(d) Emptiness check of the resulting product (This step allows to calculate a counter example when the property is not verified). The print screen in Fig.5 displays the empty set obtained after verification.

Moreover, we declare a set of initial states. The system starts at the waiting state for protocol initialization.


greatspn@localhost:/data/spot-0.8.3/iface/gspn
Fichier Édition Affichage Rechercher Terminal Aide

```
[greatspn@localhost gspn]$ ./ltlgspn-srg -e -f /data/greatspn/x86_64_R3.12.21-de
sktop-2.mga4/FipaModel  ' F  Conflict1 ' Conflict1
empty
23869 unique states visited
0 strongly connected components in search stack
max. depth = 193
removed components = 23869
states = 23869
transitions = 30673
vmsize = 14861

[greatspn@localhost gspn]$
```

Fig. 5. Property verification

4.2 State Space Analysis Results

Contrary to the former works, in our modeling Agents are represented by tokens, WN allowed not only the use of colors classes to differentiate agents' roles but also to regroup similar behavior. The main advantage of Well Formed Nets is the notion of symbolic reachability graph that is composed of symbolic states. A symbolic state is a state representing several ordinary states in the state space of the system described by the Petri net. So, much larger state spaces can be represented (the gain factor can be up to exponential).

The experimental results of our example are presented in Table 1. The table shows the number of ordinary and symbolic states according to agents number . The gain factor was able to exceed 10000 states grouped in one symbolic state (the gain factor is calculated as: $(Ord\_number - Sym\_number) \div Ord\_number$). A considerable improvement is also noticed comparing these results with thus obtained in [5], where the same protocol was modeled with hierarchical Colored Petri net and the state space generation became very slow from 5 agents, as shown in Table 2. .

5. Conclusion and Future Works

Multi-agent systems, are a topic of research in such diverse areas like marketing, e-commerce, artificial intelligence and operational research. The high complexity of these systems forces designers to use formal methodologies associated with automated tools to analyze their behavior. In this paper, we presented a Well-formed Net model for time tabling problem, using FIPA Contract Net Protocol. This type of problem is modeled for the first time by Petri Nets. We used GreatSPN connected to SPOT model checker nets

Nb agent (Teachers,Classes)	Sym. states number	Ord. states number	Gain fact. (%)	Time (s)
2 (1,1)	19	51	62,75	0
4 (2,2)	167	1815	3231,37	0
6 (3,3)	658	36655	70582,35	5
8 (4,4)	1825	5,55E+05	1,08E+06	15
10 (5,5)	3784	7,22E+06	1,41E+07	30
16 (8,8)	23869	1,044E+10	2,05E+10	171
32 (16,16)	346485	3,08E+179	6,04E+179	2658

Table 1. Experimental results with WN

Nb agent	State Space Nodes	Time (s)
3	99	0
4	5954	5
5	39618	422

Table 2. Experimental results with CPN

to verify some properties. Furthermore, the state space analysis showed the efficiency of WN and symbolic states generation to model such systems and highlighted the scalability open issue.

We are interested in our future work by analyzing a larger-scale multi-agent systems. We will extend our modeling and analysis by introducing the temporal dimension in order to perform a quantitative analysis and compute system performances, such as average waiting time, average affectation number, etc.

REFERENCES

[1] Ferber, J.: Les systèmes multi-agents vers une intelligence collective. InterEditions (1995)

[2] Thierry-Mieg, Y., Baarir, S., Duret-Lutz, A., Kordon, F.: Nouvelles techniques de model-checking pour la vérification de systèmes complexes. Revue Génie Logiciel (69) (2004) 17–23

[3] Song, S., Liu, Y., Zhang, J., Sun, J.: An extensive model checking framework for multi-agent systems. In: Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems. AAMAS '14, Richland, SC, International Foundation for Autonomous Agents and Multiagent Systems (2014) 1645–1646

[4] Taibi, M., Ioualalen, M., Salmi, N.: Analyzing e-commerce multi-agent systems using hierarchical colored petri nets. In: ICT Innovations 2012 Web Proceedings ISSN 1857-7288. (2012)

[5] Taibi, M., Ioualalen, M.: Analyzing negotiation in e-commerce multi-agent system based on fipa protocol. In: ICT Innovations 2013 Web Proceedings ISSN 1857-7288. (2013)

[6] : Cpn tools. http://cpntools.org

[7] Balbo, G., Conte, G., Donatelli, S., Franceschinis, G., Marsan, A.M.: Modelling with Generalized Stochastic Petri Nets. John Wiley & Sons Ltd (Import) (November 1995)

[8] Duret-Lutz, A., Poitrenaud, D.: Spot: an extensible model checking library using transition-based generalized buchi automata. In: IN PROC. OF MASCOTS04, IEEE Computer Society (2004) 76–83

[9] Celaya, J.R., Desrochers, A.A., Graves, R.J.: Modeling and Analysis of Multiagent Systems using Petri Nets. JCP (2009, pp. 981-996)

[10] Balague, C.: Les Systèmes multi-agents en marketing : Modélisation par les réseaux de Petri. PhD thesis, École des Hautes études Commerciales (2005)

[11] Kefi-Gazdare, M.: Optimisation Heuristique Distribuée du Problème de Stockage de Conteneurs dans un Port. PhD thesis, ECOLE CENTRALE DE LILLE (2008)

[12] Lyu, M.R., Chen, X., Wong, T.Y.: Design and evaluation of a fault-tolerant mobile-agent system. IEEE Intelligent Systems **19**(5) (Sept 2004) 32–38

[13] El Fallah Seghrouchni, A., Haddad, S., Mazouzi, H.: A formal study of interactions in multi-agent systems. International Journal of Computers and their Applications **8** (2001) 23–32

[14] Boukredera, D., Aknine, S., Maamri, R.: Modeling temporal aspects of contract net protocol using timed colored petri nets. In: STAIRS. (December 2012) 83–94

[15] Khosravifar, S.: Modeling multi agent communication activities with petri nets. International Journal of Information and Education Technology, Singapore **3**(3) (September 2013, pp. 310–3014)

[16] Hsieh, F.: Model and control holonic manufacturing systems based on fusion of contract nets and petri nets. Automatica **40**(1) (2004) 51–57

[17] Chang, L., He, X.: A methodology to analyze multi-agent systems modeled in high level petri nets. International Journal of Software Engineering and Knowledge Engineering **25**(7) (2015) 1199

[18] Holzmann, G.J.: The model checker spin. IEEE Trans. Softw. Eng. **23**(5) (May 1997) 279–295

[19] Marzougui, B., Barkaoui, K.: Interaction Protocols in Multi-Agent Systems based on Agent Petri Nets Model. International Journal of Advanced Computer Science and Applications (IJACSA) **4**(7) (2013) 166–173

[20] Lomuscio, A., Qu, H., Raimondi, F.: Mcmas: A model checker for the verification of multi-agent systems. In: Proceedings of the 21st International Conference on Computer Aided Verification. CAV '09, Berlin, Heidelberg, Springer-Verlag (2009) 682–688

[21] Gammie, P., van der Meyden, R.: Mck: Model checking the logic of knowledge. In: Computer Aided Verification, 16th International Conference, CAV 2004, Boston, MA, USA, July 13-17, 2004, Proceedings. (2004) 479–483

[22] : Fipa contract net interaction protocol. `http://www.fipa.org/specs/fipa00029`

[23] Chiola, G., Dutheillet, C., Franceschinis, G., Haddad, S.: Stochastic well-formed colored nets and symmetric modeling applications. IEEE Trans. Computers **42**(11) (1993) 1343–1360