Petri Net Modelling and Simulation in AnyLogic and MATLAB for ARGESIM Benchmark C4 'Dining Philosophers'

Ruth Leskovar[1,*], Johannes Tanzler[1], Martin Bicher[1,2]

[1] Inst. of Analysis and Scientific Computing, Vienna University of Technology, Wiedner Hauptstraße 8-10, 1040 Vienna, Austria; *ruth.leskovar@tuwien.ac.at

[2] dwh Simulation Services, Neustiftgasse 57-59, 1070 Vienna, Austria

Simulation Notes Europe SNE 24(1), 2014, 55 - 58
DOI: 10.11128/sne.24.bn04.10237
Received: June15. 2013; Revised December 2, 2014;
Accepted: January 4, 2014;

Abstract. To analyse the 'Dining Philosophers' comparison Petri Net models are used and implemented in two different libraries based on the simulation environments, MATLAB and AnyLogic are used. Different strategies are investigated to solve this logical problem. Therefore the visualisation in the used MATLAB library is very helpful but the easy application of the AnyLogic library allows a fast adaption and testing of different strategies.

Introduction

This benchmark 'Dining Philosophers' [1] describes the situation of five philosophers sitting around a table meditating. Like every human being also philosophers have the need to eat. Following from the definition of this comparison it is not possible that all philosophers eat simultaneously.

A task of this comparison is to investigate if every philosopher gets enough food and if this is not the case to create strategies in which this is ensured. The optimum strategy is found when every philosopher gets about the same amount of food.

According to the comparison description the method of Petri Nets has to be applied for analysing this situation. In this particular project two different libraries for Petri Nets are used, on the one hand a library created by Gaspar Music [3], an associate Professor of the University of Ljubljana, Slovenia, which is based on MATLAB, and on the other hand a library in AnyLogic [4].

1 Model Description

The initial situation is that five philosophers are sitting around a large round table and everyone has a bowl of food in front of him and between each of the bowls a chopstick is placed.

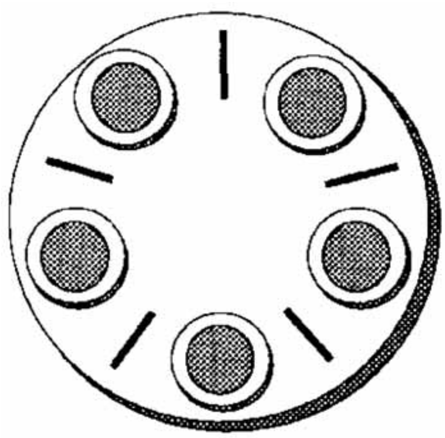

Figure 1: Philosopher's table [1].

All philosophers are either meditating or eating. If a philosopher wants to eat, both chopsticks beside his bowl have to be available. Each philosopher takes the chopstick to his left first and if available afterwards the one on his right. If a philosopher has accomplished getting both chopsticks he starts to eat and then he starts meditating again. Philosophers need lots of food because of their hard mental work and so they are hungry again immediately after they have started meditating.

It is obvious that if all philosophers are meditating at the same time, all of them grab the left chopstick simultaneously in the next step. But the problem is that then there are no chopsticks left and therefore no one has a chopstick available to his right.

For analysing and solving this problem Petri Nets [2] are used to describe this model. Petri Nets are used for a qualitative analysis of a system. Every Petri Net consists of places, transitions, arcs and tokens. Arcs always connect places with transitions. The tokens are located at the places and if predefined logical conditions concerning the transitions are fulfilled, tokens can change the places. Tokens can only change their positions along the direction of the arcs. Concerning the logical conditions a transition can only fire if all places which are connected by arcs with directions to the transition are taken with tokens. Further a transition, connected with places by arcs in direction to the places, adds a token to each of those places.

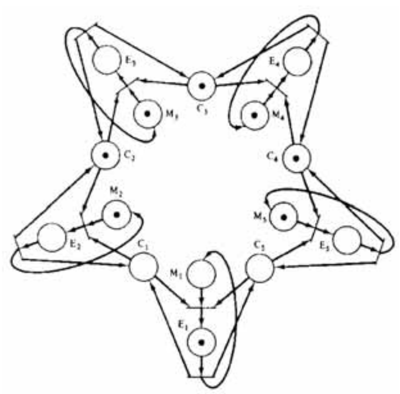

Figure 2: Philosopher's Petri Net [1].

Figure 2 shows a Petri Net of the philosophers sitting around the table. The places E_i represent the eating state and the M_i represent the meditating state of each philosopher. The places C_i stand for the chopsticks. If a token is placed in a place, it means that this state is active respectively that the chopstick is available. Figure 2 does not include the theory that every philosopher first takes the chopstick to his left and in this case just one philosopher is eating and the others are meditating.

Like mentioned above, if the preference of the left hand is considered, it is possible that at some point no one can get a second chopstick and eat. The first strategy to solve the problem is to add a hungry state. This means that after meditating the philosopher gets hungry and then takes the left chopstick and if then the chopstick to his right is available, he eats.

Also in this case a deadlock can occur and so the strategy 'gentleman' is added. This means that one philosopher is very gentle but also very fast in his actions. If this particular philosopher gets hungry, he does not take the left chopstick first.

2 Implementation

2.1 AnyLogic

For the implementation in AnyLogic a programming example of the AnyLogic Help is used. This example provides a possibility to work with Petri Nets. It has places, transitions and arcs which just have to be connected properly and in this case a little bit adapted. The places and transitions are implemented as agents. As an additional opportunity it is possible to use this as well for timed Petri Nets. But in this model the time is set to 1 and so it is equivalent to a not timed Petri Net.

2.2 MATLAB

For the implementation in MATLAB a Petri Net library from Gaspar Music is used. First of all the initial marking of the tokens and the number of places have to be defined. Then the transitions have to be defined. This means how many transitions are needed and in which direction they are connected to the places. Simultaneously in this step the arcs are defined automatically. For a proper graphical output the positions of the arcs and the transitions have to be set. For this model the library has to be adapted slightly.

3 Results

First of all the Petri Net was implemented with an initial marking as shown in Figure 2. Figure 3 shows in which state the philosophers are at the end of the simulation run.

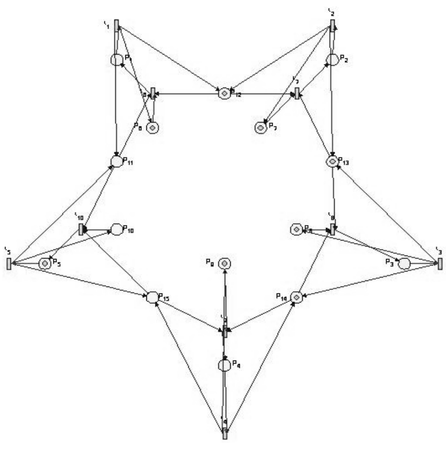

Figure 3: Petri Net – MATLAB.

Obviously one philosopher is eating while the others are meditating. In this scenario, with this initial marking, it is not possible that more philosophers are eating at the same time because the chopsticks are not available immediately after eating as a result of passing the Transitions. In Figure 4 the places `Eating` and `Meditating` of each philosopher are presented.

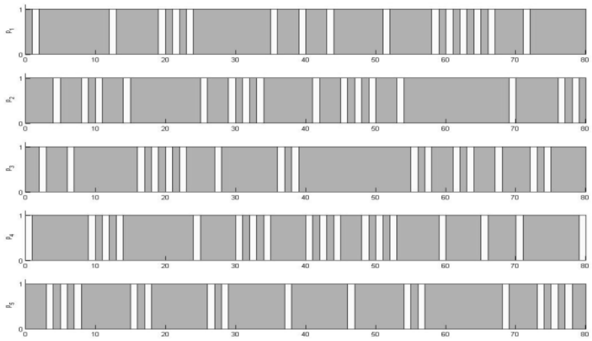

Figure 4: Places Eating and Meditating – one Philosopher eating – MATLAB.

The black bars mark the places `Meditating` and the yellow bars represent the places `Eating`. Clearly each philosopher stays in one of these two places. No deadlock can appear under those circumstances.

Using an initial marking where two philosophers are eating it is easy to see that then alternating two philosophers are eating and none is eating. Nevertheless with both initial markings the philosophers get the same amount of food. In Figure 5 the result of the simulation with two philosophers eating is presented.

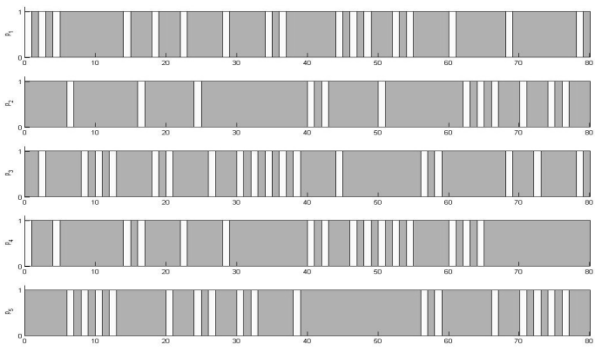

Figure 5: Places Eating and Meditating – two Philosophers eating – MATLAB.

In the definition of the project is mentioned that the Philosopher do not take both chopsticks at once but take first the left. So in a further model an additional place `Left Hand` is implemented for each Philosopher. In this scenario each simulation leads to a deadlock i.e. all philosophers stop in the place `Left Hand`. In Figure 6 for each Philosopher the places `Left Hand` and Eating are plotted, the black bars mark the `Left Hand` and the yellow bars mark the `Eating Places`.

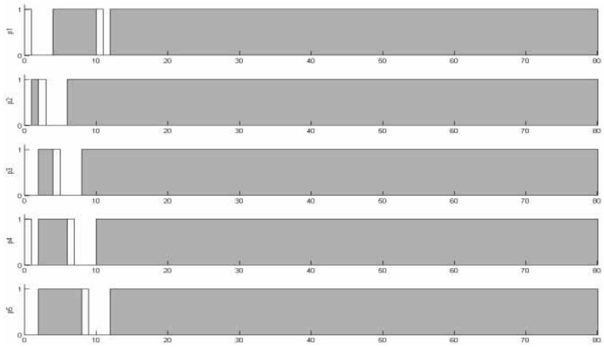

Figure 6: Places Eating and Left Hand – MATLAB.

To handle this problem an extra place `Hungry` is added for each philosopher. This place is scheduled between the places `Meditating` and `Left Hand` and therefore the chopstick can only be taken if `Hungry` is active. This strategy does not prevent the possibility of a deadlock.

On the one hand this depends on the initial marking on the other hand it depends on which transition is randomly chosen. For analysing this scenario an initial marking with two philosophers eating is chosen because all the possible outcomes if just one philosopher is eating are included and investigated. There are three possible outcomes for this model. One possibility is a deadlock this means in this case that all philosophers have their left chopstick in their left hand.

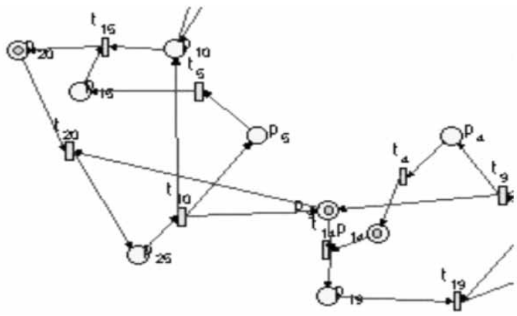

Figure 7: Transition Conflict – MATLAB.

In Figure 7 key marking is shown on which the outcome of the simulation depends. One philosopher holds his left chopstick in his left hand (`Place20`) and his right neighbour is hungry at this moment (`Place14`). Between them one chopstick is available (`Place9`) now it depends if the `Transition14` or the `Transition20` will fire. This situation is called a transition conflict when the firing conditions are fulfilled for two different transitions with the same Token. In MATLAB as well as in AnyLogic it is randomly chosen which Transition will fire.

If the `Transition14` fires the token jumps to the `Place19` which represents the `left hand`. To get from place `Left Hand` in the Eating state the right chopstick is needed. If the right chopstick is not available and no other philosopher is eating a deadlock occurs.

The second possibility occurs if the `Transition20` fires the token then jumps in the `Place25` which means that the philosopher eats and the right neighbour stays hungry. If no other philosopher is eating then the other four philosophers hold their left chopsticks in their hands and so the neighbour to his left can start to eat by taking the right chopstick. This eating loop then goes on counter clockwise.

The third possibility is similar to the second possibility. In this case the situation which is explained above occurs several times at different steps, which leads to an additional loop and so the eating frequency is increased like illustrated in Figure 8.

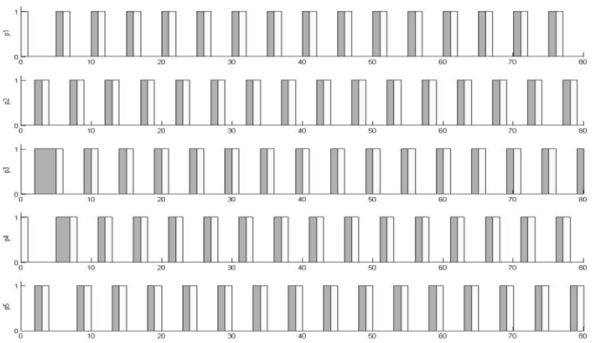

Figure 8: Places Eating and Left Hand – two Philosophers eating – MATLAB.

To prevent the remaining deadlocks another strategy is implemented. In this case one of the philosophers, called the 'gentleman', takes both chopsticks at a time and does not take the left chopstick first. Therefore if not the second philosopher to his left or he himself eats at least the one to his left can eat and so again a counter clockwise loop occurs.

4 Conclusion

To summarise, the advantages and disadvantages of both libraries are described. As usual the drag and drop method of AnyLogic does not require much previous knowledge and is easy to expand. Additionally the visualisation of the simulation is already included, in contrast to the MATLAB library where additional programming is necessary for proper visualisation. But for the better understanding of the Petri Net the MATLAB library is easier to follow because the Tokens are visualised in the places and not in the transitions like in AnyLogic. Additionally it should be mentioned that it is not possible to exclude deadlocks in this Petri Net model by using these implementations but needs further analysing.

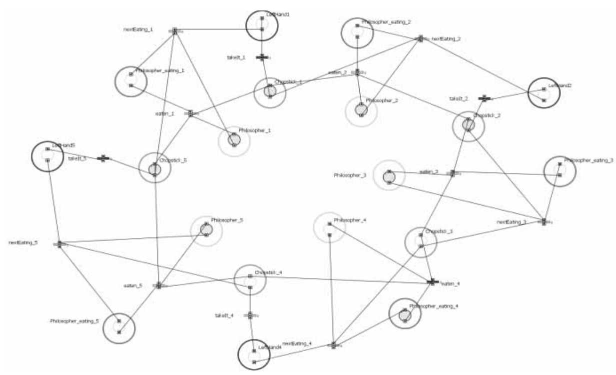

Figure 9: Gentleman Petri Net – AnyLogic.

In Figure 9 the Petri net in AnyLogic for this strategy is shown where the `Philosopher3` is the 'gentleman'.

Model sources

In both simulators AnyLogic and MATLAB special modules for Petri net modelling are used: in AnyLogic a template for Petri nets is used (part of AnyLogic), in MATLAB Music's external toolbox was used. Model parametrization files for both simulation systems, a short file description and a link to the MATLAB Petri net toolbox are can be downloaded (zip format) by EUROSIM socitetіes' members from SNE website, or are availably from the author.

References

[1] SNE Editors. Comparison 4: Dining Philosophers Problem. EUROSIM - Simulation News Europe. 1991; 3(1), 28 p.

[2] Priese L, Wimmel H. Petri-Netze. 2th ed. Berlin: Springer; 2008. 374 p.

[3] Music G, Hafner I, Winkler S, Skrjanc I. A MATLAB based Petri net Tool for E-learning: Examples for timed simulation and scheduling. In: Breitenecker F, Troch I. Full Paper Preprint Volume of the 7th Vienna International Conference on Mathematical Modelling. 7th Mathmod; 2012 Feb; Vienna. Vienna: ARGESIM and ASIM.

[4] The AnyLogic Company, 2014, AnyLogic Example model. Petri Nets. AnyLogic 7 University 7.0.2.