SCJ SOFT COMPUTING JOURNAL
ISSN: 2322-3707
University of Kashan

A Petri Net Based Flexible Model for Reasoning and Behavior Modeling of Event-Based Systems

Mina Chavoshi[1] and Seyed Morteza Babamir [2,* †]

[1,2]Department of Computer, University of Kashan, Kashan, Iran.

Abstract. In this article, the behavior of event-based and rule-based systems is modeled using Hierarchical Fuzzy Petri nets (HFPN). In such systems, a large number of rules with fuzzy variables can lead to increase complexity in deducing behavior. So far, several FPN methods have been presented for these systems. In this paper, we present an HFPN leading to a reduction in the number of arcs, places and transitions as well as a reduction in the ML language code on Petri-net arcs and the increase of the code constructiveness. Finally, we applied our method for modeling and reasoning a secure water treatment system against burst pipe attack.

Keywords: State Transition Table, Fuzzy Petri Net, Fuzzy Inference, Critical Systems, Knowledge-Based Systems

1. Introduction

Usually, the behavior of non-deterministic (fuzzy) systems [1] is described by fuzzy rules where challenges exist in incorporating uncertainties, particularly environmental uncertainties, into their models. Based on FPNs, various studies have addressed cases in which rules are expressed using high-level variables [2]. Their application in the reasoning process has resulted in more realistic outcomes [3]. However, these methods become very complicated and unimplementable for systems with a large number of rules and variables. In our previous work [4], we presented a model for reasoning and visualizing a large number of rules in a hierarchical manner, as well as managing numerous features using colored tokens. However, it does not incorporate the considerations discussed in previous studies regarding rules. In our other work [5], inspired by [6], we described the rule based system using a State Transition Table (STT) and map it to FPN.

Our current paper aims to address the shortcomings of the methods previously presented. We describe the behavior of the system using STT whose each row corresponds to a rule where each rule consists of several conditions and a conclusion; then, STT is

*Corresponding author. Email: babamir@kashanu.ac.ir
†Email(s): first-author@email.com, babamir@kashanu.ac.ir

mapped FPN leading to creating a model that serves as the *knowledge base* for the system. However, when a system contains too many rules and many fuzzy variables, the system's behavior becomes complex and difficult to deduce. Using the hierarchical structure, HFCPN (Hierarchical fuzzy colored Petri nets) can mange such complexity. Moreover, it can decrease the number of transitions and places when there exist the same conditions in multiple rules; this reduces the complexity. To tackle these issues, inspired by our previous method [5], we present a new HFCPN model where the reasoning process is incorporated into the model. The technique we used for reducing transitions and places in our HFPN is enriching and deepening the conditions on the HFPN transitions using the *ML language functions*. While our previous model uses n conditions on n Petri-net transitions for n fuzzy values of a variable, in the current model we used a *case command*, instead. This led to two advantages: (1) reducing the time order of checking conditions on the transitions, (2) constructive making decisions on the arcs using the *case command*, and (3) making more understandability of the Petri-nets using the aggregation of *if commands* to a *case command*. The *the case command* uses *jump table* where the case checking has the execution constant order (O(c)) for checking n conditions. We could not use the *case command* in our previous method because of scattering conditions on arcs. The detail of using ML programming with Colored Petri nets (CPN) was explained in [7, 8].

We applied our proposed method to monitor the attacks on a *Secure Water Treatment System* where the system behavior is modeled and inference is done to detect attacks [9].

2. Proposed model

Our proposed model can solve problems where the behavior of the system is described in STT whose typical row was shown in Table 1. Each row is mapped to a rule and as the table shows, each row indicates the change of n variable value that leads to making a transition from a *normal state* to an *abnormal one*. A change in a variable value is considered as *event* and no change in a variable value is done as a *condition*. Based on the mathematical method used in the reasoning process and the opinion of domain experts, various parameters of the rule can be created. A rule contains parts: (1) premise and (2) conclusion where the premise part contains events and conditions with probabilities. Moreover, (1) the threshold values for the conditions, which expressed in terms of propositions, and the conclusion, and(2) the *certainty factor* for the rule are considered.

TABLE 1. The STT structure

Previous State	variable 1	...	variable n	Current State
normal	condition or event of variable 1 or -	...	condition or event of variable n or -	one of m abnormal states

Given a system has n variables (features) and $m + 1$ states consisting an *unknown* state and m *abnormal* ones, Table 1 consisting of a number of rows is created. When the value of a variable remains unchanged during the transition from the previous state to the current one, it is considered as a *condition*; otherwise, it is done as an *event*, which is shown in "@old value_new value" where the old and new values indicate the variable value in the previous and current states, respectively. Notation "-" for a variable value indicates "don't care" (no event and no condition). Finally, after the reasoning process, the degree of truth of abnormal states is determined. After mapping the problem specification to

STT, for each row, a rule is extracted where conditions and events constitute the premise part of the rule and the abnormal state is considered as the conclusion part.

We use a function called *res function* for reasoning in which the extracted rules are exploited. In addition to the rules, this function incorporates a mathematical algorithm in the model. In a transition from the previous state to the current one, the proposed model considers the change/no change of values of the system variables. Then, it calls the res function to perform the reasoning process according to the reasoning method.

STT is mapped to the proposed model is performed as follows:

Variable j (column j) is mapped to:

- A high-level transition
- (degree of truth, value), which is part of the n-colored token and indicates the value and degree of truth of variable j.
- The previous value of variable j is considered as $Lvarj$,
- The current value of variable j is considered as $Cvarj$,
- The check result for variable j (condition or event) is considered as $varj$
- The truth degrees of conditional propositions are considered as $pj1$ and $pj2$. If $varj$ is an event, both variables will contain a value, but if is a condition, $pj1$ will contain a value and $pj2$ will be null.
- Two low-level transitions are included to evaluate the condition or event.
- each abnormal state is mapped to a place in HFPN at level zero.

Figures 1, 2, and 3 depict a general model of a system with n variables, representing levels zero, 1, and n of the model, respectively. Each colored token represents n values and truth degrees for n variables at each moment where are enclosed in quotation marks ("") and in brackets ([ ]), respectively. By firing a transition, the colored tokens from the places "last state" and "current state" are combined and are put in the place "ready to compare", which initiates checking the status of the variables. In the proposed model, to determine the change/no change of each variable value, we consider one level (except the zero level) for the variable. Therefore, for n variables, the proposed model will have $n+1$ levels.

The process of checking the status of variables is similar at all levels. We explain level 1 (Fig. 2), for example. The colored token in place "Ready to compare" in Fig. 1 evaluates the status of variables and enters level 1. Each of the expression variables on the input arc in level 1 contains the value and the truth degree, which are referred to by #1 and #2, respectively. Two variables, $Lvar1$ and $Cvar1$ represent the values of the first variable in the last and current states, respectively. Expressions $\#1Lvar1$ and $\#1Cvar1$ are used in the conditions for low-level transitions related to the first variable for enabling and firing the corresponding transition. Having fired the transition, there are two situations in which a distinct colored token enters the next level for continuing the process:

- If the variable value dose not change in the transition from the previous state to current state (i.e., it is considered as a condition), expression $\#1Lvar1$, $\#2Lvar1$,[] replaces expression ($Lvar1$,$Cvar1$) as a conclusion and it is transferred to the next level. The null expression "[]" remains null because the value of the variable has not changed.
- If the variable value changes (i.e. it is considered as an event), expression "@"^#1Lvar1^ "_"^"value",$\#2Lvar1$,$\#2Cvar1$ as the conclusion replaces expression ($Lvar1$,$Cvar1$) and it is transferred to the next level.

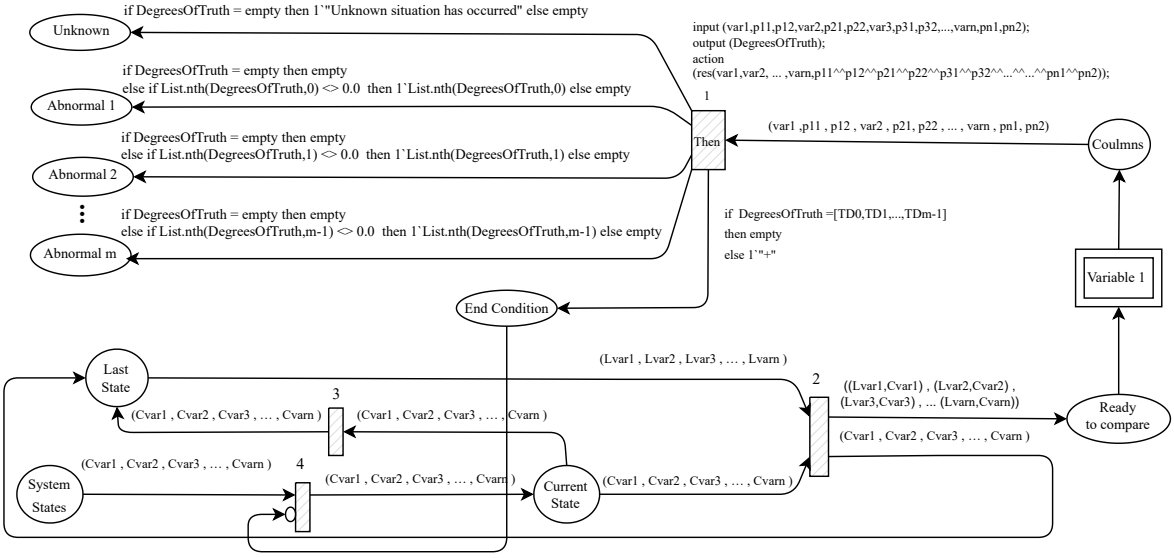

FIGURE 1. Level zero of the proposed model.

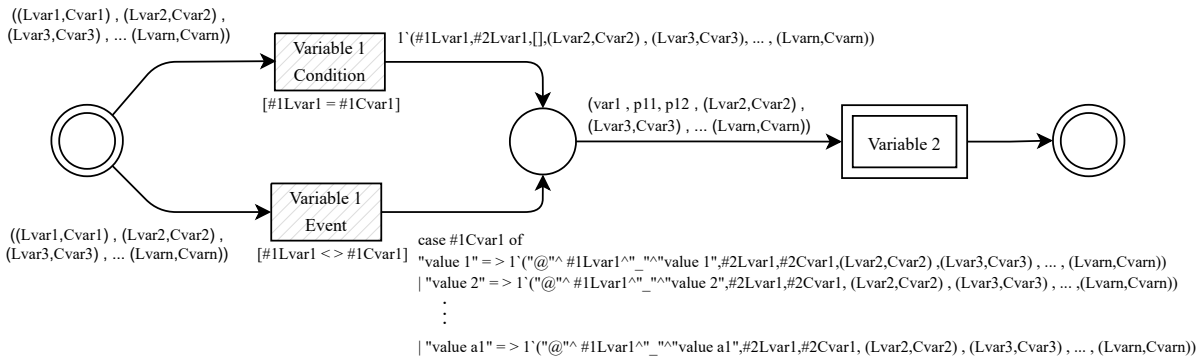

FIGURE 2. Level 1 of the proposed model.

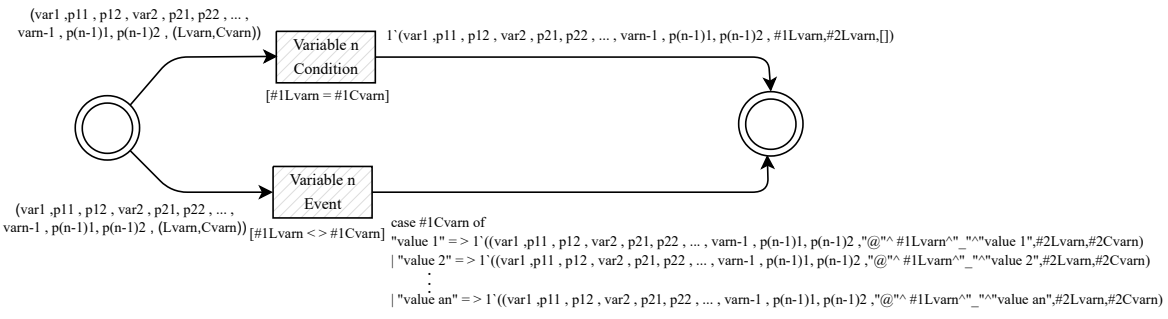

FIGURE 3. Level n of the proposed model.

By firing transition and executing the code segment of this transition, at zero level, *res function*, which is performed for the reasoning process, is called. The input parameters of this function are $var1$, $var2$,..., and $varn$, which contains the specified status for n variables, and an array, containing the truth degrees of the n variables, which is defined as operator "^^", and p11^^p12^^p21^^p22^^...^^pn1^^pn2 is added. Variable *DegreesOfTruth* contains the output of *res function* and a list, each element of which indicates the truth degree of one of the *abnormal* states. The probability of not entering abnormal states based on the type of the variable used in the reasoning process, replaces terms TD0, TD1,..., and TDm-1 at the zero level of the model.

3. Case study

Consider a water treatment system with five variables: FIT101, LIT101T, MV101, P101, and P102. The system can encounter four abnormal states *pipe bursts, tank overflow, tank underflow, and pump damage* where the water flows stops.

Suppose the scenario where an attacker initiates a *pipe burst* attack by changing the pump value from 'off' to 'on.' In this scenario, the variables $MV101$, $P101$, and $P102$ have deterministic values with a probability of 1, while variable $FIT101$ and $LIT101$ exhibit fuzzy values with probabilities of 1 and 0.7, respectively. Table 2 shows the token values in the model at the previous state and the current one (attack).

TABLE 2. Tokens values for previous state and current abnormal state (as an attack) in the model.

Variables of system (FIT101,LIT101,MV101,P101,P102)	Colored token
(min,high,close,on,off) normal state(previous state)	1`("min",1.0),("high",0.7),("close",1.0),("on",1.0),("off",1.0)
(min,high,close,on,on) attack state(current state)	1`("min",1.0),("high",0.7),("close",1.0),("on",1.0),("on",1.0)

Moreover, Eq. 1 is implemented in *res function* for the reasoning process. In this equation, based on the rule corresponding to the system's status, the minimum truth degree of the variables in the premise part of the rule is considered according to [2]. This value is then multiplied by the rule certainty factor and displayed as the conclusion of reasoning in the model. The number of truth degrees varies from 1 to $2n$.

$$\alpha_{Attack} = min(\alpha_1, \alpha_2, ..., \alpha_j) * CF \qquad 1 \leq j \leq 2n \tag{1}$$

In the following rule, the first list in the conclusion part of the rule indicates the rule is related to which attack, where value 1 denotes the desired attack and value zero does another attack. The second list shows the certainty factor associated with the rule. Figure 4 shows the reasoning outcomes for this particular system status.

if fit101="min" and also lit101="high" and also mv101="close" and also p101="on" andalso p102="@off_on" then [[1,0,0,0],[0.98]]

4. Analysis of results

In this section, considering our previous model, we analyze results of the current model. Given n and m show the number of system variables and the number of abnormal states,

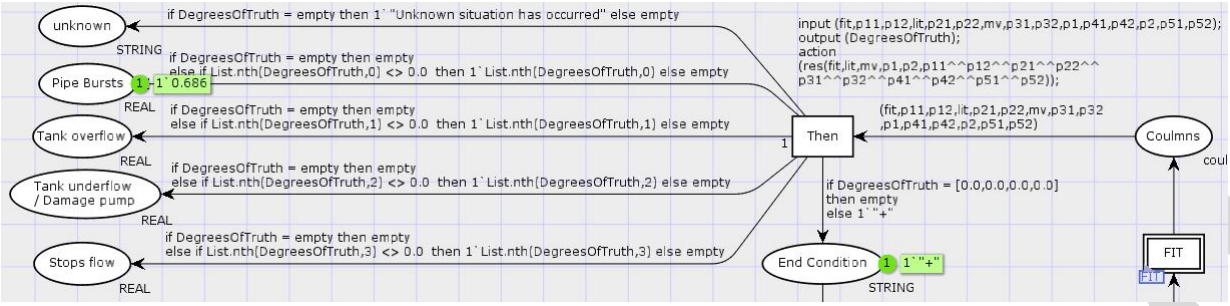

FIGURE 4. The result of the reasoning process.

respectively. Additionally, let a_1 to a_n denote the number of values considered for each variable from 1 to n, respectively. According to the case study in the previous section, $m = 4$ and $n = 5$. The number of values considered for the variables are $a_1 = 3$, $a_2 = 5$, $a_3 = 3$, $a_4 = 3$ and $a_5 = 3$, respectively. Table 3 shows the parametric formulas for calculating the number of arcs, transitions, and places for the case study. In this table, for the current model, there are seven fixed places at level zero (see Fig. 1), one of which shows the *unknown abnormal* state. Additionally, there are m places for abnormal states, totaly $m+7$. For all levels but levels zero and n (see Fig. 2), there are three places, totally $3(n-1)$. At the last level of the model (Fig. 3), there are 2 places. Therefore, the total number of places in the current model can be expressed as $3(n-1)+m+9$.

TABLE 3. Total transitions, places, and arcs in the current and the previous model for the case study.

	Previous Model	Current Model	Case study in:Previous model-Current model
The number of transitions	$2(a_1 + ... + a_n)$+n+5	3n+4	44-19
The number of places	4(n-1)+m+11	3(n-1)+m+9	31-25
The number of arcs	$5(a_1 + ... + a_n)$+2n+m+14	6n+m+12	113-46

In contrast, the previous model contains eight fixed places at level zero (Fig. 5), along with m places for the m abnormal states, totaling $m+8$. For n variables, all levels but levels zero and n had four places, totaly $4(n-1)$. The last level (Fig. 6) always contains three places; therefore, we have totally $4(n-1)+m+11$. For the transitions, the current model has four low-level transitions at level zero (Fig. 1). For each variable n, there are two low-level and one high-level transitions, totally $3n+4$. In contrast, the number of low-level transitions the in the previous model depends on the number of the values assigned to the variables. In the previous model, there are five low-level transitions at level zero (Fig. 5). For other levels, there are two low-level transitions for each fuzzy value considered for each variable and one high-level transition for each variable, totally $2(a_1 + ... + a_n) + n + 5$.

For the arcs, in the previous model, there exist $16 + m$ arcs at level zero (Fig. 5) where m denotes the number of input arcs to abnormal states at level zero. At each other level, there are five arcs corresponding to each fuzzy value assigned to variables, totally $5(a_1 + ... + a_n)$. In all levels, but level zero, there were two arcs for entering and exiting the high-level transition, totally $2(n-1)$ arcs. The sum of all arcs becomes $5(a_1 + ... + a_n)$ + $2n$ +m+14. In the current model, there are 14 arcs at level zero (Fig. 1) and four arcs for each variable in order to enter and exit from low-level transitions (at the level of the variable, Figs. 2 and 3); therefore, there will exist totally $4n$ arcs. For all variables but

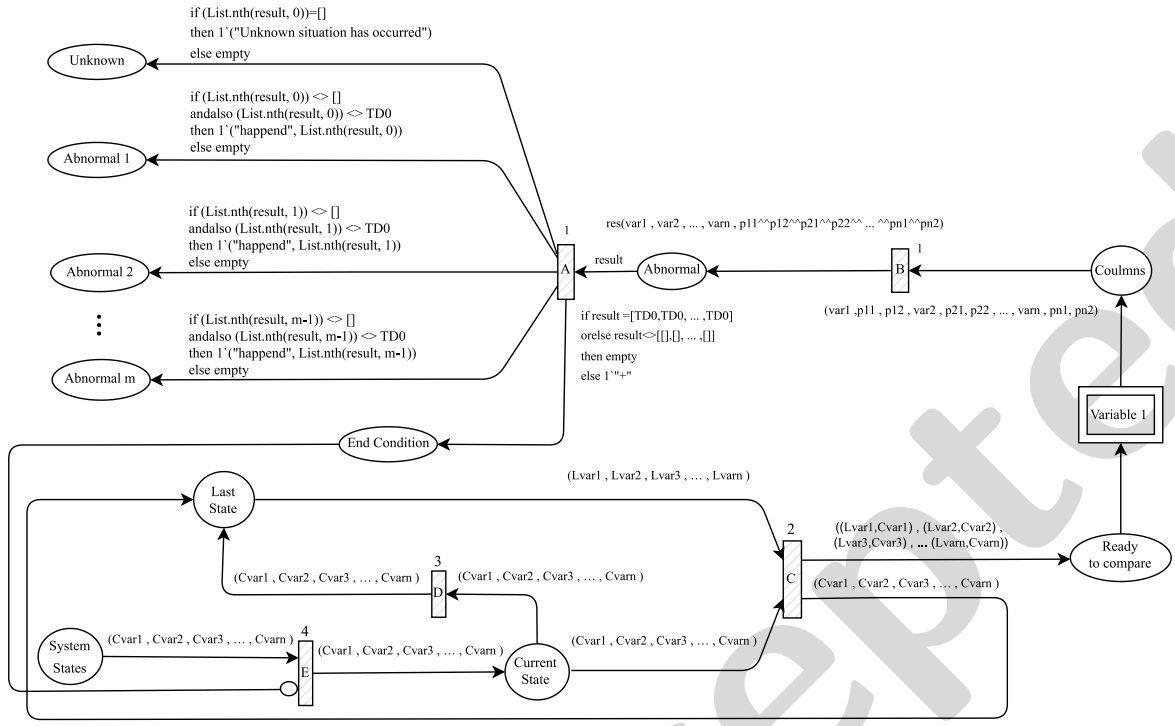

FIGURE 5. Level zero of the previous model

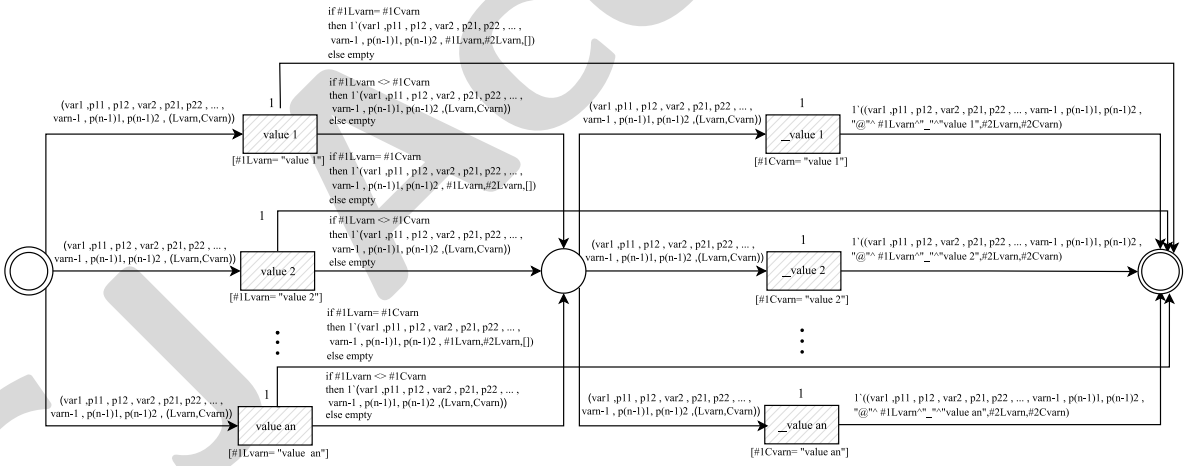

FIGURE 6. Level n of the previous model

the last one, two arcs are considered for entering and exiting from high-level transitions, totally $2n-1$ arcs (Fig. 3). Therefore, the sum of all arcs are $6n+m+12$ where m denotes the number of input arcs to abnormal states at level zero.

As Table 3 shows, the number of transitions, places, and arcs decreases, comparing to the previous model. In fact, in the current model, only two low-level transitions are considered for each variable but in the previous one, for the fuzzy values defined for each

variable, two low-level transitions were created. When a system contains a large number of fuzzy values for variables, it becomes a complex model. However, by mapping the process of checking the state change of each variable onto arcs, the model becomes less dependent on the number of the fuzzy values assigned to each variable, regardless of their diversity or multitude. This is why the current method leads to less complexity of the model.

4.1. Complexity reduction factor. The Main reason for the complexity reduction in our current HFPN is that the transitions that were considered for the *fuzzy* values in the previous HFPN, are removed from the model in the current one; instead, *case* statements were added to the arcs and the conditions for checking variables were moved from arcs to transitions. For instance, consider level 1 of the HFPN, which are shown in Fig. 7 for the previous model and in Fig. 2 for the current one. Transitions "value 1" to "value a_n" and "_value 1" to "_value a_n for "Variable 1" in Fig. 7 were removed from Fig. 2. Similarly, we have such reduction for other levels. Consider the first levels of the previous model in Fig. 7 and the current one in Fig. 2 and their last levels in Figs. 6 and 3; for each level that is added to the previous and current models for checking a variable, the current one will have a place less than the previous one. Also, level zero of the current one has a place less than the previous one. Therefore, the current model totally will have $n+1$ places less than the previous one.

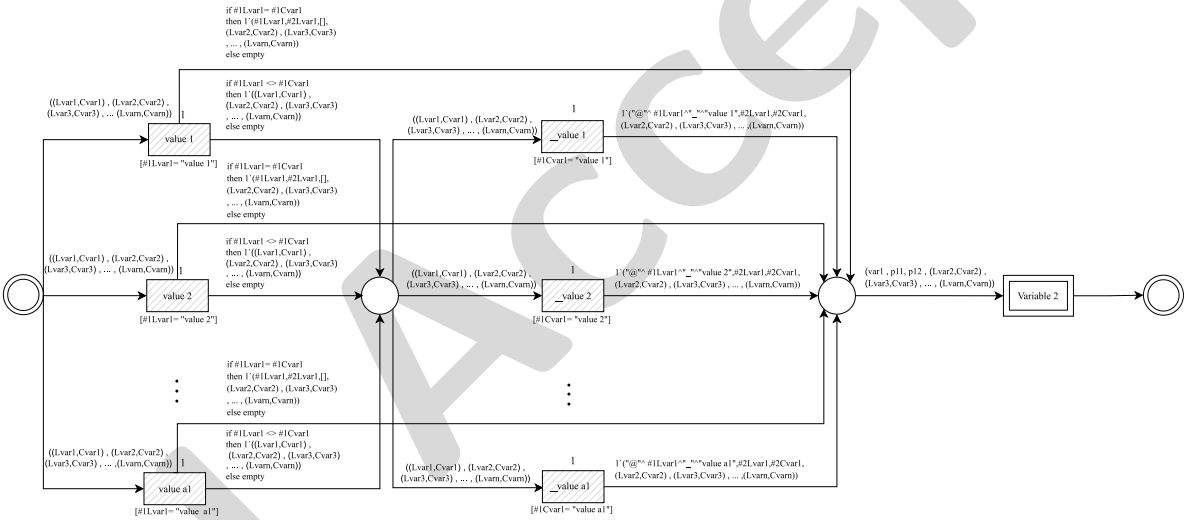

FIGURE 7. Level 1 of the previous model

Time complexity. The time complexity for our current and previous methods are considered as follows. Consider Table 3; among $3n+4$ transitions, $2n+4$ transitions fire in the current model but in the previous model, among $2(a_1 +...+ a_n)$+n+5 transitions, at most $3n+5$ transitions fire, which is about n transitions more than the current model.

5. Conclusion

In this article, we introduced a model based on a hierarchical fuzzy color Petri net for modeling systems, which is less complicated than the previous one. The proposed model is highly flexible in incorporating various parameters for the rules and expressing them

using different high-level variables and mathematical reasoning methods based on algebra. We used a *case command* instead of several *if commands* leading to using a constructive command. Moreover, The number of transitions was reduced leading to reducing the time order of checking conditions on the transitions, constructive making decisions on the arcs, and making more understandability of the Petri-nets using the aggregation of conditions on variable to a *case command*.

References

1. Varshney AK, Torra V. Literature review of the recent trends and applications in various fuzzy rule-based systems. International Journal of Fuzzy Systems. 2023 Sep;25(6):2163-86.
2. Shi H, Liu HC. Dynamic Adaptive Fuzzy Petri Nets for Knowledge Representation and Acquisition. InFuzzy Petri Nets for Knowledge Representation, Acquisition and Reasoning 2023 Sep 19 (pp. 63-83). Singapore: Springer Nature Singapore
3. Chen S.M., Fuzzy backward reasoning using fuzzy Petri nets. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). vol. 30, no. 6, pp. 846856 (2000), https://doi.org/10.1109/3477.891146
4. Majma N., Babamir S.M., Model-based monitoring and adaptation of pacemaker behavior using hierarchical fuzzy colored petri-nets. IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 50, no. 9, pp. 33443357 (2018), https://doi.org/10.1109/TSMC.2018.2861718
5. Chavoshi M., Babamir S.M., Extracting Rules from Specifications and Their Modeling using Colored Fuzzy Petri-Nets. Soft Computing Journal, vol. 11, no. 1, pp. 2247 (2022), https://doi.org/10.22052/scj.2022.246562.1078.
6. Heitmeyer C. Software cost reduction. Encyclopedia of software engineering. 2002 Jan 15;2:1374-80.
7. Jensen K, Kristensen LM, Jensen K, Kristensen LM. CPN ML Programming. Coloured Petri Nets: Modelling and Validation of Concurrent Systems. 2009:43-77.
8. Jensen K, Kristensen LM. Colored Petri nets: a graphical language for formal modeling and validation of concurrent systems. Communications of the ACM. 2015 May 21;58(6):61-70.
9. Goh J, Adepu S, Junejo KN, Mathur A. A dataset to support research in the design of secure water treatment systems. InCritical Information Infrastructures Security: 11th International Conference, CRITIS 2016, Paris, France, October 1012, 2016, Revised Selected Papers 11 2017 (pp. 88-99). Springer International Publishing.