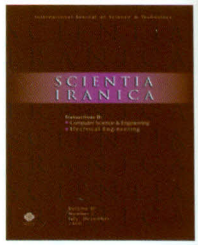
Scientia Iranica D (2011) 18 (3), 554–565

Sharif University of Technology

Scientia Iranica

Transactions D: Computer Science & Engineering and Electrical Engineering

www.sciencedirect.com



Formalized learning automata with adaptive fuzzy coloured Petri net; an application specific to managing traffic signals

S. Barzegar [a,*], M. Davoudpour [b], M.R. Meybodi [c], A. Sadeghian [b], M. Tirandazian [b]

[a] *Department of Electronic and Computer Engineering, Qazvin Islamic Azad University, Qazvin, P.O. Box 34197-1416, Iran*

[b] *Department of Computer Science, Ryerson University, Toronto, ON M5B 2K3, Canada*

[c] *Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, P.O. Box 15875-4413, Iran*

Received 5 July 2010; revised 26 October 2010; accepted 14 February 2011

This work is dedicated to Professor Lotfi Zadeh on the occasion of his 90th Birthday.

KEYWORDS

Adaptive coloured Petri nets;
Fuzzy logic;
Learning automata;
Traffic signal control.

Abstract Investigation of the chaotic behavior of traffic streams at urban intersections due to signals has involved researchers in endeavoring to predict a smooth traffic flow model for stabilizing traffic congestion and avoid unnecessary delays. In this paper, we study a hybrid adaptive model, based on a combination of coloured Petri nets, fuzzy logic and learning automata, to efficiently control traffic signals. We show that in comparison with results found in the literature, vehicle delay time is significantly reduced using the proposed method.

1. Introduction

Coloured Petri Nets (CPN) are a tool by which the validation of discrete-event systems is studied and modeled. CPNs are used to analyze and obtain significant and useful information from the structure and dynamic performance of the modeled system. Coloured Petri nets mainly focus on synchronization, concurrency and asynchronous events [1]. The graphic features of CPNs specify the applicability and visualization of the modeled system. Furthermore, synchronous and asynchronous events present their prioritized relations and structural adaptive effects. The main difference between CPNs and Petri nets (PN) is that in CPNs, the elements are separable, but in PNs, they are not. 'Coloured' indicates the elements specific feature. The relation between CPNs and ordinary PNs is analogous, in high level programming language, to an assembly code (low level programming language). Theoretically, CPNs have precise computational power, but practically, since high level programming languages have better structural specifications, they have greater modelling power.

The drawback of CPN is their non-adaptivity [2] and therefore it is not possible to access previous information available in CPNs. If there is more than one transition activated, then each transition can be considered as the next shot. This 'Coloured' Petri net characteristic indicates that since several events occur concurrently and event incidences are not similar, when events do occur, they do not change by time, and this phenomenon is in contrast to the real, dynamic world. Simulation would be similar to execution of the main program. Our purpose is to use the simulated model for analyzing the performance of the systems, and as a result, system problems and weak points would be identified. However, classic CPN tools can do nothing to improve and solve the problems, and also it is not possible to predict the next optimized situation.

In this paper, we present an Adaptive Fuzzy Coloured Petri net based on learning automata. Using the information from previous states of the system and reactions of the dynamic environment, an Adaptive Fuzzy Coloured Petri net will predict the next optimized situation. This will update the current state of the system and will change and activate the probability of occurrence in time. The performance of the reaction on systems in the dynamic environment will significantly help the Fuzzy Coloured Petri nets to learn and get trained. In this paper, we have used CPN tools for Fuzzy CPN simulation.

* Corresponding author.
E-mail address: Barzegar@qiau.ac.ir (S. Barzegar).

1026-3098 © 2011 Sharif University of Technology. Production and hosting by Elsevier B.V. All rights reserved. Peer review under responsibility of Sharif University of Technology.

doi:10.1016/j.scient.2011.04.007

Production and hosting by Elsevier

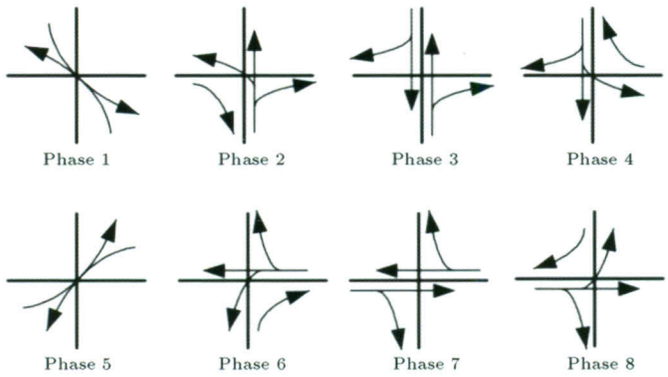


Figure 1: Eight-phase transition model.

Furthermore, we have studied application of the adaptive model mentioned in the previous paragraph to control traffic signals. The adaptive model is represented to optimize the scheduling of traffic signals across intersections. Optimizing the methods used to control the traffic signals is significant, since it reduces air pollution, fuel consumption and improves time efficiency. The proposed algorithm uses the learning automata to adjust fuzzy functions defined in the input parameters of the problem.

Fuzzy logic was first used in traffic control systems by Pappis and Mamdani [3]. They simulated an isolated intersection composed of two one-way streets with no turns. Later, Niittymaki and Pursula [4] also simulated an isolated intersection where the proposed fuzzy logic controller led to a shorter vehicle delay and fewer stops. Niittymaki and Kikuchi [5] developed a fuzzy logic algorithm to control signals for pedestrians. Through simulation, it was shown that their algorithm performed better than the conventional one. Chiu [6] used fuzzy reasoning to control multiple intersections with no turns. Fuzzy rules were used to adjust cycle time, phase split and offset parameters. Later, Niittymaki [7] introduced a simple, two-phase, fuzzy signal controller. It was shown that the fuzzy logic controller performed better than the vehicle-actuated controller.

In our test model, we study an intersection with two main directions, namely North–South and East–West [8], with a traffic signal having 8 transitional phases, as shown in Figure 1.

Controlling traffic signals could be considered event discrete systems, which represent the level of synchronization and concurrency. Since the main advantage of PNs is observing synchronization and concurrency, existing PNs are the best choice for simulation, analysis and evaluation of an urban traffic network. Therefore, we have used CPNs as a tool [1,9].

The rest of the paper is structured as follows. In Section 2, we present learning automata as the basic learning strategy used in our proposed method. In Sections 3 and 4, Fuzzy CPNs and the Adaptive Coloured Petri net are briefly introduced. The proposed algorithm is introduced in Section 5. Section 6 explains the analysis of the proposed algorithm, and its comparison with the results of fuzzy algorithms used for scheduling traffic signals. Finally, we have the concluding remarks.

2. Learning automata

Learning automata is an abstract model that randomly selects one action out of its finite set of actions and evaluates it on a random environment, then again evaluates the same action

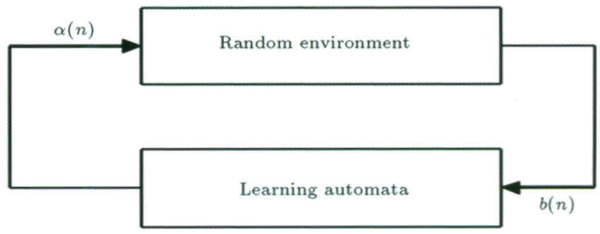


Figure 2: Relationship between learning automata and its environment.

and responds to the automata with a reinforcement signal. Based on this action and the received signal, the automaton updates its internal state and selects its next action. Figure 2 illustrates the relationship between an automaton and its environment.

The environment can be defined by $E = \{a, b, c\}$, where $a = \{a_1, a_2, \ldots, a_r\}$ represents a finite input set, $b = \{b_1, b_2, \ldots, b_r\}$ represents the output set, and $c = \{c_1, c_2, \ldots, c_r\}$ is a set of penalty probabilities; each element, c_i of c, corresponds to one input of action a_i. An environment, in which b can take only binary values, 0 or 1, is called the P-model environment. Also, by further generalization of the environment, it is possible to have finite output sets with more than two elements that take values in the interval [0, 1]. Such an environment is called the Q-model environment. Finally, when the output of the environment has continuous random variables, and assumes values in the interval [0, 1], this environment is then known as an S-model environment. Learning automata are classified into a stochastic fixed-structure, and a stochastic variable-structure. In the following, we only consider variable-structure automata.

A variable-structure automaton is defined by the quadruple $E = \{a, b, p, T\}$ in which $a = \{a_1, a_2, \ldots, a_r\}$ is a set of actions (or outputs of the automaton). The output or action of an automaton is an instant of n denoted by $a(n)$, which is an element of the finite set, $a = \{a_1, a_2, \ldots, a_r\}$. $b = \{b_1, b_2, \ldots, b_r\}$ represents the input set or response set, $p = \{p_1, p_2, \ldots, p_r\}$ represents the action probability set and finally $p(n+1) = T(a(n), b(n), p(n))$ represents the learning algorithm. The following shows the action of the automaton based on the action probability set, p. The automaton randomly selects an action, a_i, and performs it on the environment. After receiving the environment's reinforcement signal, the automaton updates its action probability set based on (i) for favorable responses, and (ii) for unfavorable ones.

$$\begin{aligned} p_i(n+1) &= p_i(n) + a.(1 - p_i(n)), \\ p_j(n+1) &= p_j(n) - a.p_j(n), \quad \forall j,\ j \neq i, \\ p_i(n+1) &= (1-b).p_i(n), \end{aligned} \tag{1}$$

$$p_j(n+1) = \frac{b}{r-1} + (1-b)p_j(n), \quad \forall j,\ j \neq i, \tag{2}$$

where a and b are reward and penalty parameters, respectively. If $a = b$, the automaton is called L_{RP}. If $b = 0$, the automaton is called L_{RI} and, if $0 < b < a < 1$, the automaton is called $L_{R\varepsilon P}$.

For the $S - L_{RP}$ model learning automata, a linear algorithm is given below:

$$\begin{aligned} p_i(n+1) &= p_i(n) + a(1-\beta_i(n))(1-p_i(n)), \\ p_j(n+1) &= p_j(n) - a(1-\beta_i(n))p_j(n), \quad \forall j,\ j \neq i. \end{aligned} \tag{3}$$

For the $S - L_{RI}$ model learning automata, a linear algorithm is given below:

$$\begin{aligned} p_i(n+1) &= p_i(n) + a.(1-\beta_i(n)).(1-p_i(n)) \\ &\quad - a.\beta_i(n).p_i(n), \\ p_j(n+1) &= p_j(n) - a.(1-\beta_i(n)).p_j(n) \\ &\quad + a.\beta_i(n).\left[\frac{1}{r-1} - p_j(n)\right] \\ &\quad - a.(1-\beta_i(n)).p_j(n), \quad \forall j,\ j \neq i. \end{aligned} \tag{4}$$

For the $S - L_{R\varepsilon P}$ model learning automata, a linear algorithm is given below:

$$\begin{aligned} p_i(n+1) &= p_i(n) + a.(1-\beta_i(n)).(1-p_i(n)) \\ &\quad - a.\beta_i(n).p_i(n), \\ p_j(n+1) &= p_j(n) - a.(1-\beta_i(n)).p_j(n) \\ &\quad + a.\beta_i(n).\left[\frac{1}{r-1} - p_j(n)\right] \\ &\quad - a.(1-\beta_i(n)).p_j(n), \quad \forall j,\ j \neq i. \end{aligned} \tag{5}$$

More information about learning automata can be found in [10].

3. Fuzzy coloured Petri nets

Coloured Petri nets were introduced by Kurt Jensen in 1987, as a developed model of Petri nets. Coloured Petri nets are appropriate tools for mathematical and graphical modeling. Coloured Petri nets have numerous applications, and much research has taken place, with respect to modeling, describing and analyzing systems, which have synchronized, asynchronized, distributed, parallel, non-deterministic or random natures. In fact, Petri nets are models that could represent the performance and state of the system at the same time. There has been enormous research done in the following areas,

(i) Controlling and learning systems using coloured Petri nets,
(ii) Optimizing Petri net structures using genetic programming,
(iii) Learning and reasoning the ambiguous problems using fuzzy coloured Petri nets.

However, there is no record of adapting coloured Petri nets and using learning automata in Petri nets.

A formal definition of CPN is as follows [1]:

Definition 1. A Coloured PN (CPN) is a 6-tuple CPN $= (P, T, C, I^-, I^+, M_0)$ where:

1. $P = \{p_1, p_2, \ldots, p_n\}$ denotes a finite and non-empty set of places,
2. $T = \{t_1, t_2, \ldots, t_m\}$ denotes a finite and non-empty set of transitions, $P \cap T = \emptyset$,
3. C is a colour function that assigns a finite and non-empty set of colors to each place and a finite and non-empty set of modes to each transition
4. I^- and I^+ denote the backward and forward incidence functions defined by $P \times T$, such that:

   $I^-(p,t), I^+(p,t) \in [C(t) \to C(p)_{MS}], \quad \forall (p,t) \in P \times T.$
5. M_0 denotes a function defined on P, describing the initial marking, such that $M_0(p) \in C(p)_{MS}$.

A formal definition of a FCPN is as follows [11]:

Definition 2. A generalized non-hierarchical Fuzzy Coloured Petri net is defined as 12-tuple FCPN $= (\Sigma, P, T, D, A, N, C, G, E, \beta, f, I)$ where:

1. $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_l\}$ denotes a finite set of non-empty types, called colour sets where $l \geq 0$.
2. $P = \{P_C, P_F\}$ denotes a finite set of places;
   - $P_C = \{pc_1, pc_2, \ldots, pc_m\}$ denotes a finite set of places that model the dynamic control behaviour of a system, and is called control places where $m \geq 0$;
   - $P_F = \{pf_1, pf_2, \ldots, pf_n\}$ denotes a finite set of places that model the fuzzy production rules, and is called fuzzy places where $n \geq 0$, and $P_C \cap P_F = \emptyset$.
3. $T = \{T_C, T_F\}$ denotes a finite set of transitions;
   - $T_C = \{tc_1, tc_2, \ldots, tc_i\}$ denotes a finite set of transitions that are connected to and from control places, and is called control transition where $i \geq 0$;
   - $T_F = \{tf_1, tf_2, \ldots, tf_j\}$ denotes a finite set of transitions that are connected to or from fuzzy places, and is called fuzzy transition where $j \geq 0$, and $TC \cap TF = \emptyset$.
4. $D = \{d_1, d_2, \ldots, d_h\}$ denotes a finite set of propositions, $|PF| = |D|$.
5. $A = \{a_1, a_2, \ldots, a_k\}$ denotes a finite set of arcs, $k \geq 0$, and $P \cap T = P \cap A = T \cap A = \emptyset$.
6. $N: A \to P \times T \cup T \times P$ denotes a node function, and it maps each arc to a pair, where the first element is the source node and the second element is the destination node; the two nodes have to be of different kinds;
   - In: an input function that maps each node, x, to the set of nodes that are connected by an input arc$(x) \to x$;
   - Out: an output function that maps each node, x, to the set of its nodes that are connected to x by output arc$(x) \to x$.
7. $C: (P \cup T) \to \Sigma_{ss}$ is a colour function, which maps each place and transition to a super-set of colour sets.
8. $G: T \to$ expression which denotes a guard function:

   $\forall t \in T : [\text{Type}(G(t)) = \text{Boolean}$
   $\wedge\, \text{Type}(\text{Var}(G(t))) \subseteq \Sigma],$

   where Type(Vars) denotes the set of types, $\{\text{Type}(v) | v \in \text{Vars}\}$. Vars denotes the set of variables, and Var$(G(t))$ denotes the set of variables used in $G(t)$.
9. $E: A \to$ expression which denotes an arc expression function,

   $\forall a \in A : [\text{Type}(E(a)) = C(p(a))\text{MS}$
   $\wedge \text{Type}(\text{Var}(E(a))) \subseteq \Sigma],$

   where $p(a)$ is a place in $N(a)$, and MS stands for multi-set.
10. $\beta: PF \to D$ denotes a bijective mapping from fuzzy places to a proposition.
11. $f: T \to [0, 1]$ denotes an association function, which assigns a certainty value to each colour used in each fuzzy transition.
12. I: denotes an initialization of double (δ, α),
    - $\delta: P \to$ expression which denotes an initialization function:
      $\forall p \in P : [\text{Type}(\delta(p)) = C(p)\text{MS}].$

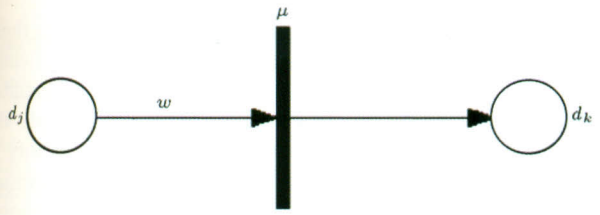


Figure 3: The FCPN denotation of fuzzy coloured rule of Type 1.

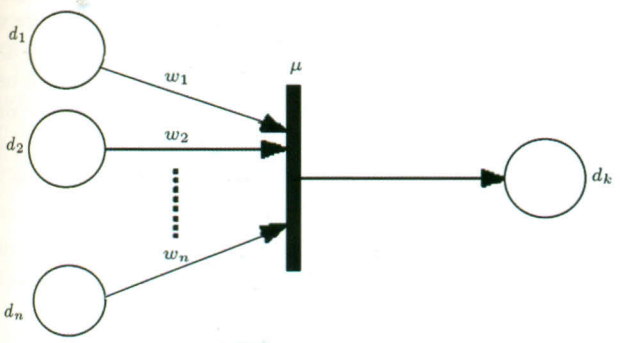


Figure 4: The FCPN denotation of fuzzy coloured rule of Type 2.

- α: denotes an association function, which assigns a certainty value in the range [0, 1] to each token in the fuzzy places.

The structure of Fuzzy Coloured Petri nets depends on fuzzy production rules. The composite fuzzy production rules could be distinguished by following three rule-types, respectively [12,13]:

Type 1: Simple fuzzy production rule (Figure 3):
IF d_j THEN d_k $(CF = u)$.
Type 2: Compound joined fuzzy production rule (Figure 4):
If d_1 AND d_2 AND $\cdots$, AND d_n THEN $d_k(CF = u)$.
Type 3: Compound disjoined fuzzy production rule (Figure 5):
If d_1 OR d_2 OR $\cdots$, OR d_n THEN $d_k(CF = u)$.

4. Adaptive coloured Petri net

By looking at the Petri net structure, we realize that the Petri net is considered a limited presentation tool with low extension for solving real world problems which are considered non-linear and dynamic. This tool is not able to self-educate in order to adapt to environmental changes. On the other hand, with a more precise look at intelligent techniques for analyzing, tools such as DAG (Directed acyclic graph) and Petri nets are needed, which have the capability to adapt education by intelligent techniques, such as neural networks and fuzzy logic. All these intelligent techniques assess one or more viewpoints. As an example of these viewpoints, we can mention human knowledge or data processing, knowledge presentation and learning. Therefore, it is necessary to design a model, by which we can express intelligent techniques and Petri nets as a single hybrid model and their combination. In recent years, emerging solutions use a combination of intelligent and other techniques, which include another dimension of real time systems [14–16]. Different methods of combined methods have been designed and implemented for solving various problems. Values, such as membership values, weights, certainty factors and learning rules, with emphasis on intelligent techniques, are

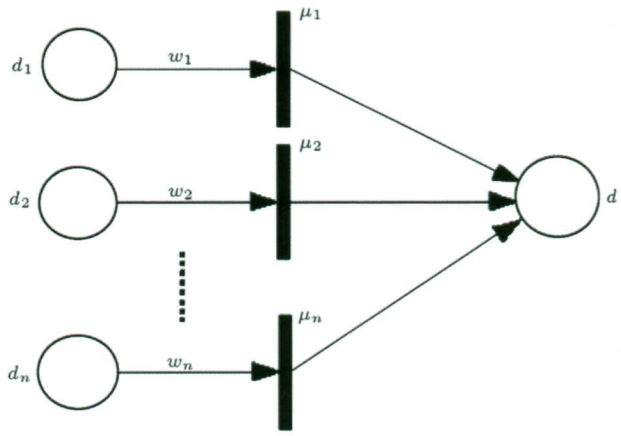


Figure 5: The FCPN denotation of fuzzy coloured rule of Type 3.

in the Adaptive Petri net design, and have been investigated for different problems. In the continuation, we assess two groups of articles that focus on these trends. Asar et al. [2] analyzed various systems that are modeled with Petri nets as adaptive features. These systems are separated into two groups:

1. Fusion hybrid,
2. Combination hybrid.

Fusion hybrid. A large group of searches are categorized into this group. Generally, papers and research which are placed in this group have used intelligent technology like fuzzy logic with the help of Petri nets in real world problems. Examples of these articles are as follows.

Tsuji et al. [17] proposed an extended PN model that merges the attributes of fuzzy inference and neural networks. In this model, each token is given a positive real number value (not more than 1). This value is a threshold of firing for a transition. Here, the boundedness, aliveness and reachability features of a model have been analyzed.

Hadjinicolaou [18] proposed a neural Petri net model that utilizes the Petri net for analyzing the behavior of a neural network. The neural Petri net model presented in this paper uses the certainty factor (that is, a real number) of weight per unit time, which changes the number of tokens over some unit of time.

Venkatesh et al. [19] presented a basic and fundamental approach to model neuron behavior through a High Level Petri Net (HLPN). In this paper, a new element was added to HPN to implement all neural network features in the model. This element has been applied to places, transitions, arcs and weights.

Ahson [20] proposed a Fuzzy Neural Petri net model that incorporates neural network features with fuzzy inference together. In this model, the Neural Petri Net (NPN) utilizes places involving three subsets including input, hidden and output. The transitions are of two types including ordinary and thresholding. Weights of Transitions are ordinary and thresholding. In input-place, transition weights are fixed to 1, while the weights on the hidden and output layers are trainable and can only take values between 0, 1 and -1. Therefore, The NPN model is modified using fuzzy inference features.

Hirasawa et al. [21,22] examine Petri net potential from the perspective of the function distribution that exists in the human brain in the biological neural network, and proposes a revised model of the Petri net, having the ability to learn as a Neural network. In this paper, a model based on the Petri net, named

Learning Petri Network (LPN), proposed that the learning ability of LPN is gained through the setting and weight adjustment.

Zha et al. [23] presented a Neural Fuzzy Expert Petri Net Model (NFEPN) with similar structure as [8], and demonstrated its unified approach to two applications, one involving the evaluation problem of non-rational design and its redesign, and the other on the intelligent robotic assembly control process.

Tsang et al. [24] studied the learning capability in fuzzy Petri nets by modeling fuzzy production rules and incorporating a neural network learning algorithm. The learning algorithm applied in this proposed model involves initialization, presenting training examples, feed forward computation and the backward arc adjustment method.

Li et al. [25,26] proposed an Adaptive Fuzzy Petri Net (AFPN) model that has not only the descriptive advantages of fuzzy Petri nets, but also a learning ability like the neural network. The work emphasizes the design of a dynamic knowledge inference framework, which is adjustable according to knowledge variation as human cognition and thinking. It introduces the weighted fuzzy production rule concept, which gives a relative degree of importance to each proposition in the antecedent, contributing to the consequence of a rule. In this paper, separate learning algorithms have been developed, called fuzzy reasoning and back propagation, respectively, to ensure convergence of adjustable weights and knowledge learning under generalized conditions.

Chen et al. [27] proposed the model of a fuzzy Petri net and an algorithm to generate a classifier network. In this proposed model, fuzzy rules are applied to the domain knowledge in order to help the local weight and certainty factor to get adjusted by online learning. Here, each input place is considered a feature of the model and, at each transition, is considered fuzzy production rules. Here, a token in any place is a membership function.

Jones and Tracy [28] attempted to model a railroad laboratory test bed based on a real time, shared resource, multiprocessing design, incorporating the attributes of neural network, timing, and colored tokens in a Petri net. Gao et al. [29] focused on a Fuzzy Reasoning Petri Net (FRPN) to represent a Fuzzy Production Rule system.

The first group emphasizes the fact that we can model a real executed model by the help of Petri nets and can change the simulated model. Here, many changes in the original structure of the Petri net and the intelligent algorithm are undertaken in order to get settled next to each other for simulating real system behaviors. For further details of this group, the reader should refer to [19]. In this paper, with the help of Petri nets, we can control brain cell activity. This idea is a comparison between a Petri net and a neural network. Here, places can include both the body of brain cells and the synaptic clefts (cell cleft node). Each place has various functions. The core cleft acts like internal and external points, while the cell body collects signal points and possesses a threshold level in order to fire. Arcs are representatives of dendrites (the long branch of a neural cell for connecting to another cell) in neural cells, and axons (small branches of a neural cell) are functions or relational arcs. As mentioned in the definition of process in this paper, some change has occurred in the original structure of the Petri net. Here, each place or arc is representative of some kind of task and behavior in the real world. Therefore, this model is a biological sample of a human mind. In this model, any biological operations done in the brain (such as an eye closing order) are shown in the Petri model as probable formulas and probability distribution function changes that have been defined for each part of the brain. So in the group of Fusion Hybrid Systems, there is a really implemented system that we are going to artificially model. We can easily analyze and assess this artificial model, which is a sample of the real world, and we can apply the results to the real world. The reason for the adaptability of this group is related to basic changes in Petri net configuration and its similarity to the real system framework. Some changes in the basic structure of the Petri net in this group of papers have occurred, and the outcome is fuzzy logic, which can give the ability of learning and teaching to the Petri net by use of intelligent algorithms, such as neural networks. This group has also some disadvantages. Most often, learning in the Petri net of this group causes an increase in operation complexity and calculation overhead, because it takes a long time to model all characteristics of a real system along with using an intelligent algorithm in the Petri net.

Combination hybrid. In this group, a relatively small group of researchers are active in applying adaptive intelligent techniques, in conjunction with Petri net methodology, to real world problems. Here, design complexity is less than the first method. Models of this group are simpler than models of the group of Fusion Hybrid, accordingly. Some papers are:

Hanna et al. [30] demonstrated the approach of integrating the fuzzy Petri net and ANN to do two separate jobs to maintain the product quality characteristics of a CNC milling machine center. Here, it utilizes a fuzzy Petri net with ANN for the modeling and control of surface roughness in the milling machine process. This paper is an example of work in terms of a hybrid approach, using a Petri net and ANN. In this hybrid model, besides other transitions related to the milling machine process, one transition, when fired, triggers the ANN to activate, and gets the ANN based output. If the ANN output is the same as the PN, then another transition fires to repeat the cycle, otherwise the error recovery transition fires to activate various parts of the machine for readjustment.

Song et al. [31] proposed an optimization technique for a Flexible Manufacturing System (FMS). With the help of this technique, the scheduling problem of resource allocation is based on the colored Petri net and Hopfield neural network model. Here, timing delay confederates with places and time propagation with tokens. Also time delay is associated with places, and time propagation with tokens. This model involves mapping the constraint relations between tasks, machines and the objective function, onto an energy function. Here, real world problems are modeled by using intelligent techniques along with Petri net methodology. In this method, different technologies are placed next to each other and, by relative changes in each technique, the presented model, for execution in the environment, will be implemented with more effectiveness and precision, in comparison with the normal mode. This group uses the initial characteristics of the Petri net, along with intelligent techniques, which are applied to the internal works and requests of each part of the real system. Also a real and clear combination of intelligent characteristics with current conditions of the system has been also suggested in order to improve the current system position. Completed jobs of this group are less than in the group of Fusion Hybrid Systems.

As an example of this group, we study the idea of Fukuda et al. [32], regarding optimized systems for controlling the

Table 1: Features and their linguistic descriptions.

Variables	Linguistic descriptions			
Q_C, Q_N	Few	Moderate	Many	Too many
AR_C, ER_C, AR_N		Short	Medium	High
SNe_C, Sne_N		Worse	No change	Better
ES_C, ES_N		Worse	No change	Better

accuracy of electromyogram sets, using the intelligence of neural networks and the task model by Petri nets. This idea has been assessed in the Robot Laboratory of Hiroshima University in Japan, on an artificial hand. The goal of this method is to create an optimized method for reducing extra movement in an artificial hand. This is an example of integrating two different technologies as a single system. In this idea, myoelectric control has been organized for improving the accuracy of electrocardiograph sets for making different statistic plans, by using a neural network, and has been suggested as an event driven task model. Each model and pattern of the electrocardiograph is as an input of the neural network in which the teaching process wants to be executed. Sensors of electrocardiograph signals are placed on the hand and joints, and send a model of hand movements and the environment status to the model of the neural network, periodically. The neural network makes a task model of jobs and tasks of a model, with regard to the primary information of a model, and gives it to the Petri net. This net compares it with regard to optimized and ideal conditions. This trend is applied until creation of an optimized model, and finally it is implemented in the environment. Each request sent to the Petri net is considered a task. So the applicable Petri net in this project is responsible for analyzing the given request to the artificial hand as a task model. The task model of the Petri net is for describing independent information that is used for describing the output of the neural system. The task model approximates the task agent by use of the records gained from neural network results. The neural network is connected to electrocardiograph models, and the model enters the neural network as an input. The output of the neural network is a task model, which has been analyzed and designed by the Petri net. As seen in the example, here, the Petri net will consider better conditions for the environment, with regard to the input and output of the neural network, in an adaptable mode.

5. Proposed algorithms

In this section, we propose a combinative algorithm to control traffic signals across intersections. In this algorithm, fuzzy logic and learning automata are used for intelligent control of traffic signals. In the proposed algorithm, learning automata are used to adjust membership functions of input and output parameters.

Before a detailed description of the algorithm and the process of adjusting fuzzy membership functions used in learning automata, we first explain the following topics: fuzzy logic, features for intelligent control of traffic signals, primitive selection of fuzzy membership functions and fuzzy rules, details of the proposed algorithm and modelling the proposed algorithm.

5.1. Fuzzy logic system

Rule base fuzzy logic systems contain four components: rules, fuzzifier, inference engine and output processor (defuzzifier). In the rule base, there are collections of 'If-Then' statements. After the information is gathered, the inputs should get fuzzified to become usable. Defuzzification is used for concurrency in the fuzzy system. Moreover, in the rule base, the output variable of the 'Then' part in each 'If-Then' statement is a fuzzy amount, and the output of different rules would not necessarily be the same. Thus to drive to a result, a fuzzy inference engine is designed and, with defuzzification, the crisp value of a deciding variable will be calculated.

5.2. Specifications to intelligently control traffic signals

The issue of controlling traffic signals could be divided into two parts:

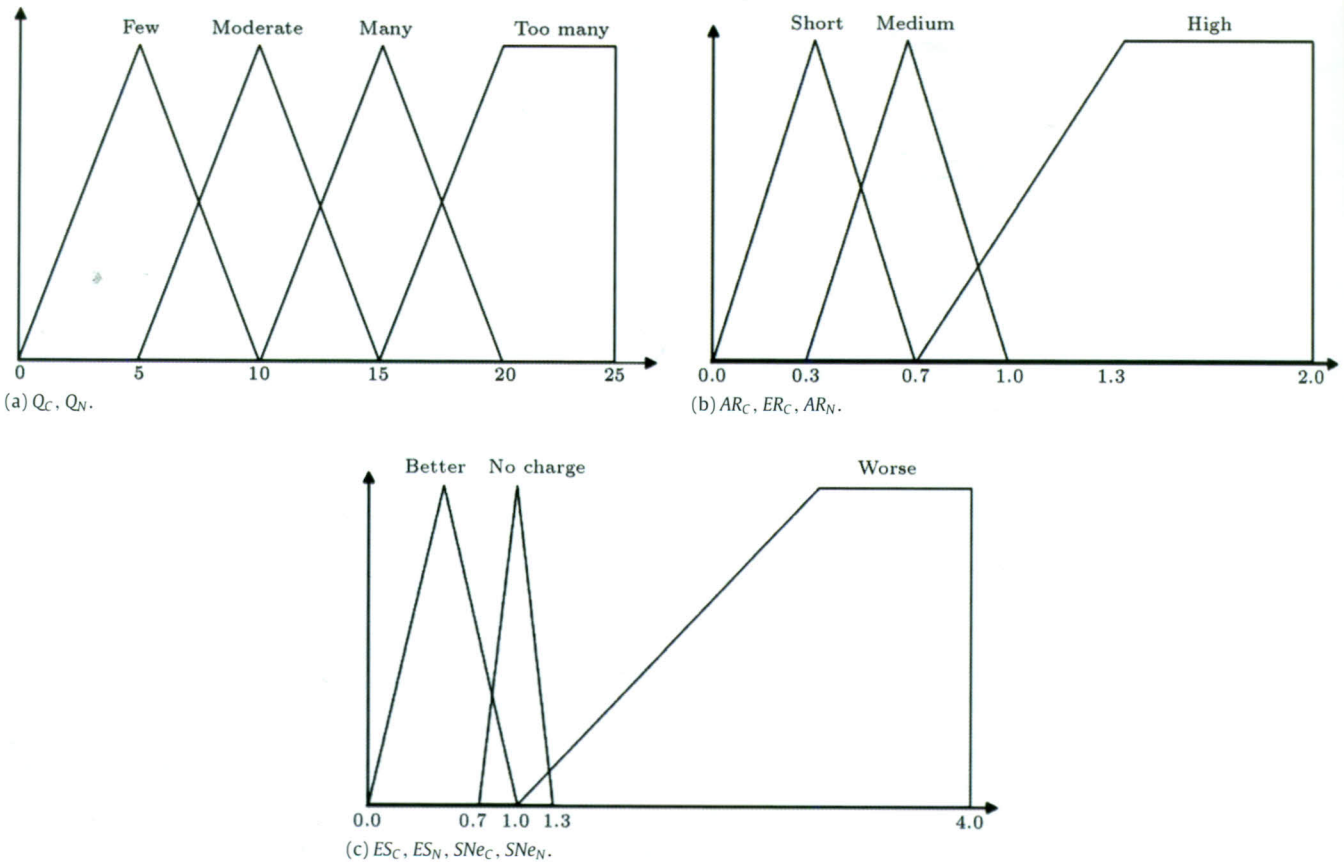
1. To determine the priority of the phases,
2. To schedule the traffic signals. In this paper, we concentrate on the second part [33–35].

The following are specifications considered for intelligent scheduling of traffic signals at an intersection: The average number of vehicles waiting to cross the intersection is denoted by (Q_C) (when the light is green). The average number of vehicles waiting to cross the intersection is denoted by (Q_N) (when the signal turns green in the next phase). The average rate of vehicles crossing through the green signal is denoted by (AR_C, ER_C). The average rate of vehicles crossing through the green light in the next phase is denoted by (AR_N, ER_N). The state of traffic at the intersection against the traffic in the neighbourhood cross section is denoted by (SNe_C, SNe_N). Predicting the state of traffic at a particular given time is denoted by Δt (within the next 10 min) on the basis of achieved information from the previous time (ES_C, ES_N). These features would be considered as inputs to control the traffic system. The final result expected from the overall system is to improve traffic congestion by increasing the period between signal changes. Each feature is defined as a fuzzy variable and each have a specific linguistic description. Table 1 shows the features and their corresponding linguistic descriptions. The output specifications that determine the increased duration of the green signal are value Zero, A Few, Few, Moderate, Many and Too Many.

5.3. Primitive selection of fuzzy membership functions

For each linguistic description of a fuzzy variable, a membership function is considered. All membership functions are equipped with learning automata and a variable structure, which have the responsibility of regulating the fuzzy function parameters. Fuzzy membership functions are either triangular or trapezoidal in shape, shown in Figures 6 and 7. The beginning and end parts of the membership functions are constant and pre-determined. The learning automata would regulate the center of membership functions to achieve the best timing and control of the traffic signals. In each automaton, a number of actions (denoted by m) are defined. The probability to choose each action of the learning automata at the beginning of the learning process is determined by $\frac{1}{m}$.

The ratio of traffic at the neighbourhood intersections to traffic at the current intersection (SNe_C, SNe_N), and prediction of traffic in a specific period of time, Δt (next 10 min), in the selected phase, on the basis of gathered information from previous times (ES_C, ES_N) are calculated by the following



(a) Q_C, Q_N.

(b) AR_C, ER_C, AR_N.

(c) ES_C, ES_N, SNe_C, SNe_N.

Figure 6: Membership functions of input variables. (a) Q_C, Q_N; (b) AR_C, ER_C, AR_N; and (c) ES_C, ES_N, SNe_C, SNe_N.

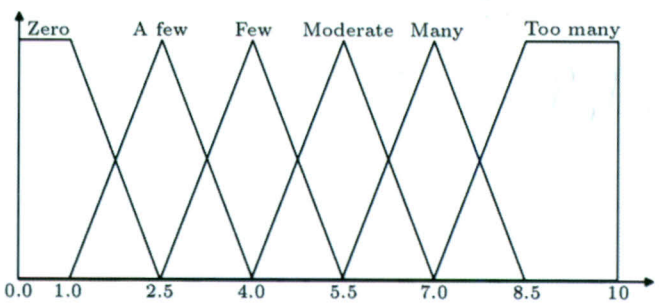


Figure 7: Membership function in output variable.

formulas:

$$ES_C, ES_N \Rightarrow \frac{\text{State of average old traffic in } t_C + \Delta t}{\text{State of average current traffic}}, \tag{6}$$

$$SNe_C, SNe_N \Rightarrow \frac{\text{State of traffic in neighbor intersections}}{\text{State of average current traffic}} + \frac{\text{Rest time of current green signal in current intersection}}{\text{Rest time of current green signal in neighbor intersection} + a} * b. \tag{7}$$

a is the required time for the vehicle to arrive from the neighbourhood intersection to the detecting intersection, and its value is determined to be 7 s. b is probability of exit of vehicles from the neighbourhood intersection to arrive at the mentioned intersection, with value 0.8.

5.4. Fuzzy rules

Some of the fuzzy production rules are as follows:

1. IF Q_C is Few AND Q_N is Too Many, THEN EX is Zero.
2. IF Q_C is Moderate AND Q_N is Few AND AR_C is Short AND AR_N is High AND ER_C is Medium, THEN EX is Zero.
3. IF Q_C is Few AND Q_N is Moderate AND AR_C is Short AND AR_N is short AND ER_C is Medium, THEN EX is A Few.
4. IF Q_C is Moderate AND Q_N is Too Many AND AR_C is Medium AND AR_N is Medium AND ER_C is High, THEN EX is Few.
5. IF Q_C is Moderate AND Q_N is Moderate, THEN EX is Moderate.
6. IF Q_C is Many AND Q_N is Many AND AR_C is Medium AND AR_N is short AND ER_C is Short AND SNe_C is Worse AND SNe_N is Worse AND ES_C is Better AND ES_N is No Change, THEN EX is Many.

Table 2: Average delay per vehicle due to uniform arrivals.

Phases	Reward and penalty values			
	$a = 0.1$ $b = 0.01$	$a = 0.1$ $b = 0.05$	$a = 0.5$ $b = 0.01$	$a = 0.5$ $b = 0.05$
1	45	46	49	50
2	47	48	52	53
3	51	51	55	55
4	48	48	51	52
5	52	51	54	56
6	50	51	54	56
7	48	49	53	53
8	34	36	39	39

7. IF Q_C is Many AND Q_N is Many AND AR_C is High AND AR_N is short AND ER_C is Medium AND SNe_C is Worse AND SNe_N is No Change AND ES_C is No Change AND ES_N is Worse, THEN EX is Too Many.
8. IF Q_C is Too Many, THEN EX is Too Many.

5.5. Details of the proposed algorithm

To control a traffic signal in an eight phase intersection, we have used an algorithm that is a combination of fuzzy logic methods and learning automata. The algorithm has two main parts:

(i) To determine the green signal priority of phases;
(ii) To increase the time of the selected phase (extender).

The minimum determined time for the green light is 10 s, that the extender of the phase can increase that amount to 5 times more, which each time would be 0 to 10 s. As a result, the maximum time of a green signal would be determined as 60 s [36]. Each time the algorithm is performed one second before, the time is increased.

The performance of the algorithm is prioritized as follows:

1. One phase out of eight phases is selected to make the traffic signal green.
2. The phase prioritizing section considers the priorities of the vehicles that have an emergency. Then, the average number of vehicles waiting to cross the intersection is calculated and accordingly the signal switches to green. For example, in any phase, if a vehicle exists having an emergency state, then the traffic signal in which the vehicle is located, turns green and the previous phase, having a green signal, turns red. Accordingly, in this way, the phase prioritizing section performs itself.
3. In a selected phase, if there is no request for any vehicle to cross the intersection, then this phase is neglected and control goes to part 2.
4. If there is a vehicle with an emergency state present, then the extender should increase the duration of the green signal to enable the vehicle to exit the phase.
5. Repeat for 10,000 times:
   I. The extender of the green signal, in each learning automaton, would select one event considering the determined probability. Therefore, as a result, for all input parameters, Q_N, Q_c, four membership functions, and for all input parameters, E_{SC}, E_{SN}, SN_{eC}, SN_{eN}, A_{RC}, E_{RC}, A_{RN}, three membership functions would be created.
   II. The membership degree for each parameter, AR_C, ER_C, AR_N, ES_C, ES_N, SNe_C, SNe_N, Q_C, Q_N, would be calculated considering the achieved information from the following:
      (i) The sensors, located at the intersections;
      (ii) The traffic information related to prior periods;
      (iii) Created membership functions in the previous part (I).
   III. Considering the amount of achieved membership for input parameters, and by activation of fuzzy rules, the output function would be determined.
   IV. Considering the new traffic conditions at the intersections, due to the increase of time in the green signal, a bonus or penalty is allotted to learning automata, accordingly. Probability vectors of the learning automata of input parameter membership functions are updated according to the following step:
      - If the Q_c Queue condition is worse than before, a penalty is assigned to the selected action; otherwise, a bonus is assigned to the selected action.
6. If no extra time was allotted to the green signal by the extender, or the extender was repeated five times, then the phase has to be changed, therefore, go to part 2, otherwise, go to part 3.

5.6. Modelling the proposed algorithm

A model is created and shown in the following figure, which illustrates Coloured Petri net tools for controlling the traffic [1,37,38].

The created model consists of three main modules:

- LA's actions selector module,
- Fuzzifier module,
- Rule and defuzzification module.

LA's actions selector module. To obtain the best result, model S in learning automata with different parameters has been tested. Each leaning automata consists of 10 actions with initial probability, 0.1. In the S model, if α_i action is selected after (n) repetitions, with the environment responding to it, then the unfavourable response would be $\beta_i(n) = 1$. If the queue gets worse, a penalty is allotted, otherwise the environment favourable response would be:

$$\beta_i(n) = \frac{1}{1 + \frac{\text{previous}}{\text{new}} \frac{Qc}{Qc}},$$

which means that if the queue gets better, a reward is given, and we could get a better response from the environment, which is shown in the simulation section. For each membership function in $S - L_{ReP}$, learning automata are used with parameters $a = 0.1$ and $b = 0.01$, which are the rate of reward and penalty, respectively.

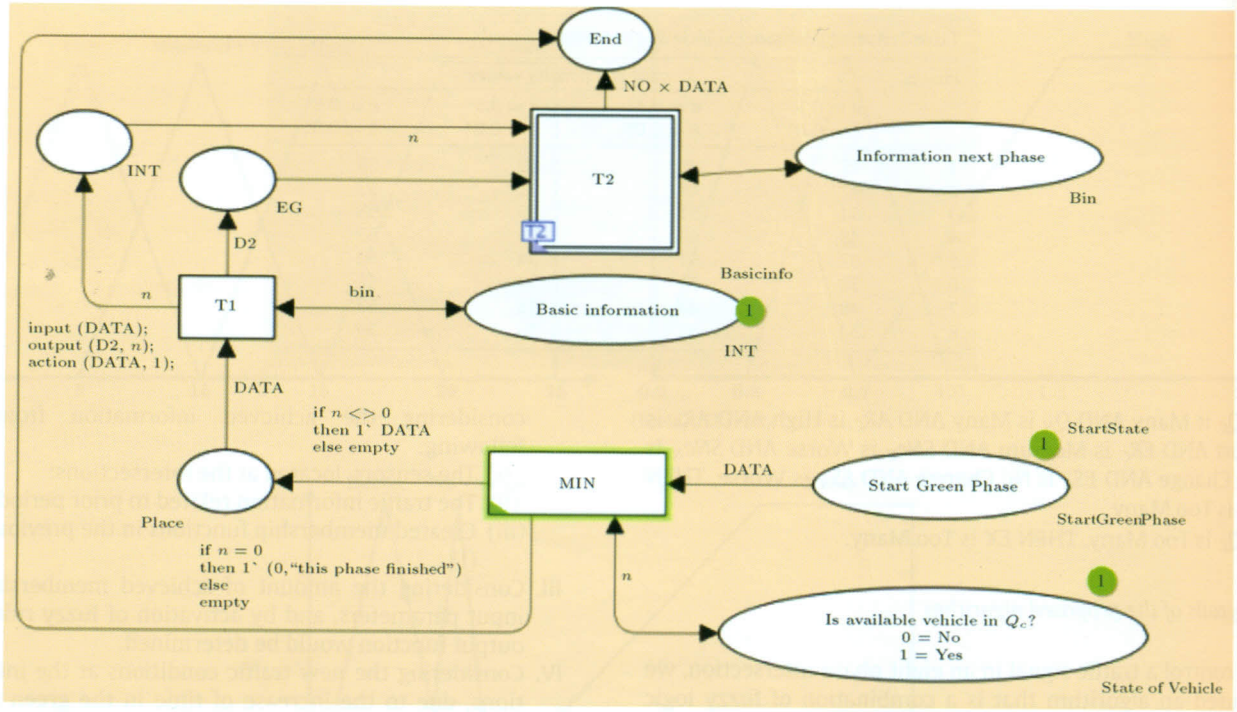


Figure 8a: Main model.

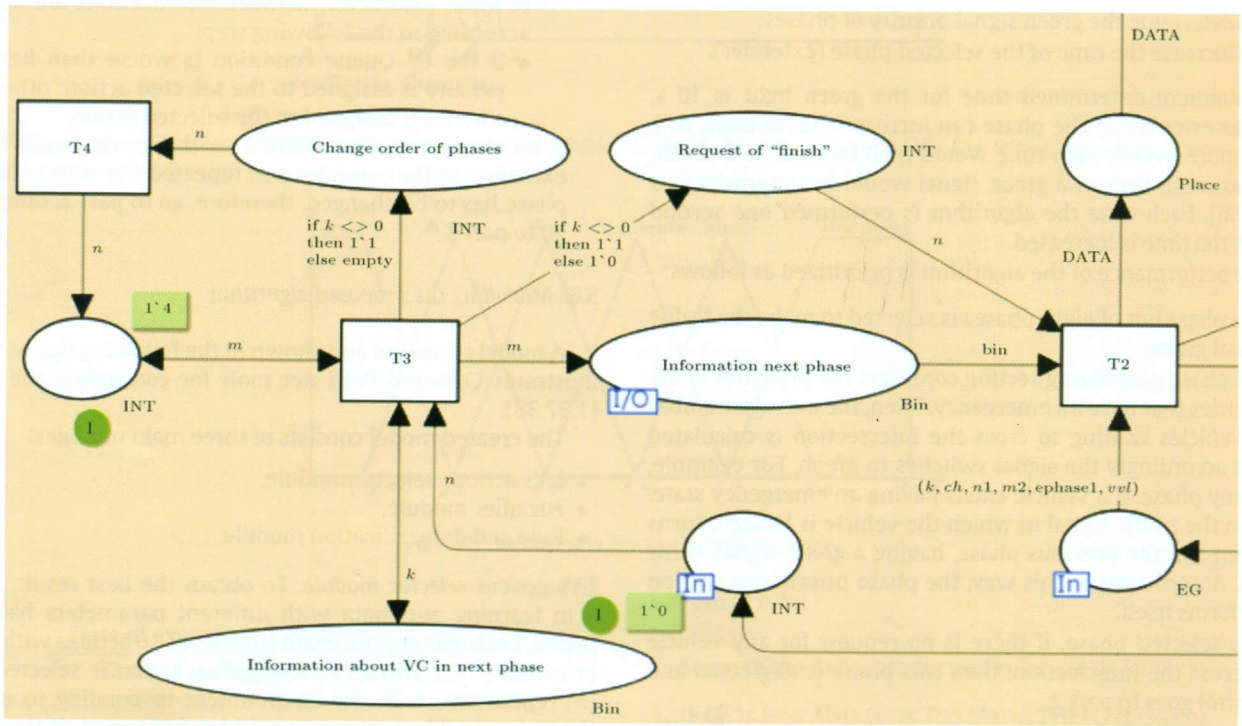


Figure 8b: Determining the phases' priorities submodel.

Fuzzifier module. Here, according to selected actions, the amounts of input parameter are converted to a fuzzy module. Note that for producing input parameters, Q_C and Q_N, Poisson distribution and, for remaining inputs, uniform distribution have been used. Figure 8 represents a sample coloured Petri net fuzzifier module of parameter Q_N.

Rules and defuzzification module. Rules are applied, and achieved results are converted to numerical values. This shows the amount of increased time applied to the green signal (Figure 8) [39–41].

6. Simulation results

A uniform delay formula is used to calculate the average delay of each vehicle [42,43]. A simulated model was performed 600 times by the colored Petri net tool to achieve the required

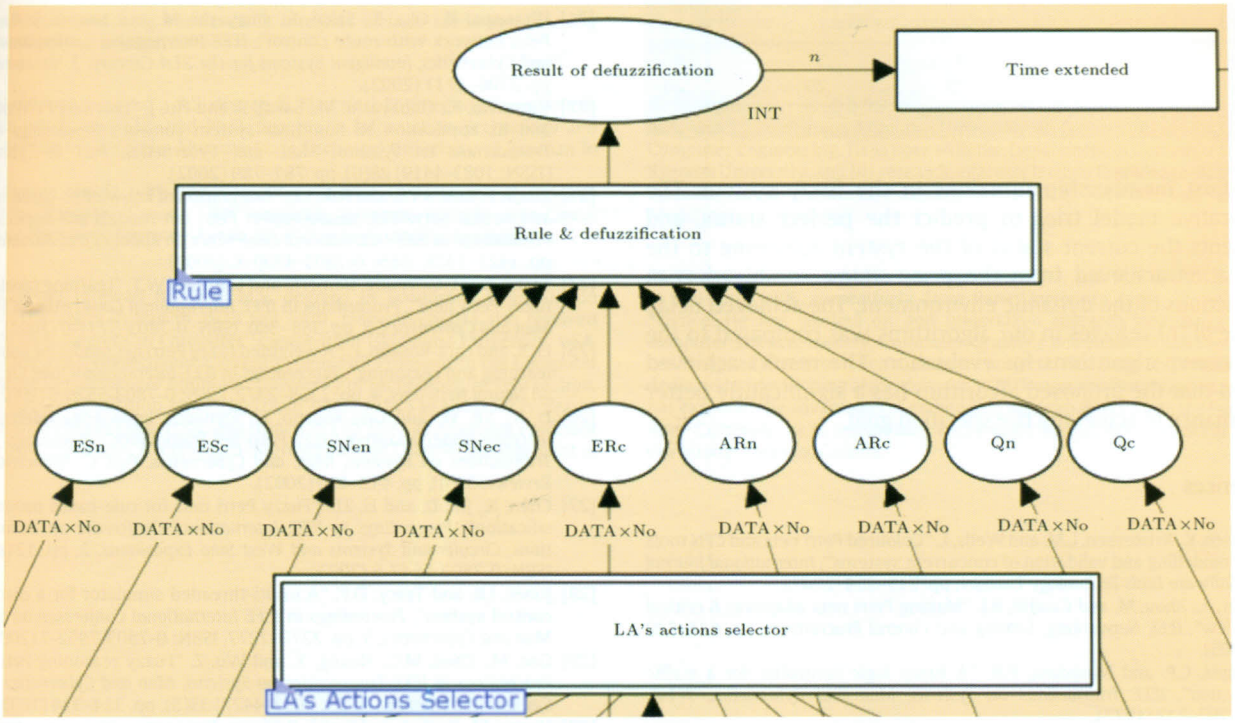


Figure 8c: Rule and defuzzification submodel.

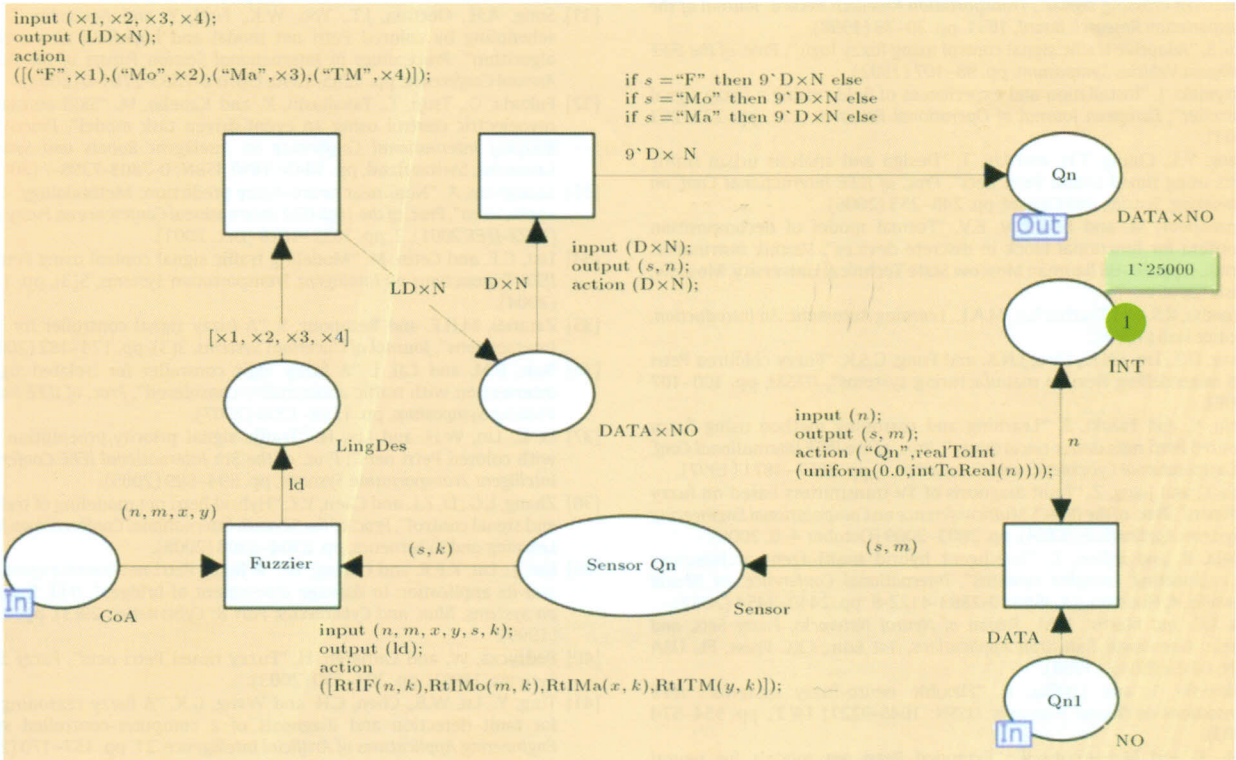


Figure 8d: Q_C fuzzifier module submodel.

results. The average delay of the 8 transitional phases is calculated and shown in Table 2.

In [4], when the traffic volume in all directions reaches 1600, the mean delay in 2 transitional phases is approximately 27 (s). The calculated mean delay of vehicles in 4 transitional phases is reported in [36] (Table 3).

7. Conclusions

In this paper, an adaptive fuzzy coloured Petri net has been presented, based on learning automata, to efficiently control traffic signals across intersections. The basis of the recommended algorithm was to combine fuzzy logic and learning automata. Learning automata were used to regulate

Table 3: Average delay per vehicle with abnormalities.

Phases	1	2	3	4
AV (s)	32	19	20	20

and adjust membership functions in the fuzzy system. The comparative model tries to predict the perfect status, and represents the current status of the system according to the information achieved from the prior states, combined with the reactions of the dynamic environment. The achieved delay average of (r) vehicles in our algorithms was compared to the other known algorithms for evaluation. The results achieved showed that the proposed algorithm has a significantly better performance in achieving the specified goal.

References

[1] Jensen, K., Kristensen, L.M. and Wells, L. "Coloured Petri nets and CPN tools for modelling and validation of concurrent systems", *International Journal of Software Tools Technology Transfer*, pp. 213–254 (2007).

[2] Asar, A., Zhou, M. and Caudill, R.J. "Making Petri nets adaptive: A critical review", *IEEE Networking, Sensing and Control Proceedings*, pp. 644–649 (2005).

[3] Pappis, C.P. and Mamdani, E.H. "A fuzzy logic controller for a traffic junction", *IEEE Transactions on Systems, Man and Cybernetics*, 7(10), pp. 707–717 (1977).

[4] Niittymaki, J. and Pursula, J. "Signal control using fuzzy logic", *Fuzzy Sets and Systems*, 116(1), pp. 11–22 (2000).

[5] Niittymaki, J. and Kikuchi, S. "Application of fuzzy logic to the control of a pedestrian crossing signal", *Transportation Research Record: Journal of the Transportation Research Board*, 1651, pp. 30–38 (1998).

[6] Chiu, S. "Adaptive traffic signal control using fuzzy logic", *Proc. of the IEEE Intelligent Vehicles Symposium*, pp. 98–107 (1992).

[7] Niittymaki, J. "Installation and experiences of field testing a fuzzy signal controller", *European Journal of Operational Research*, 131, pp. 273–281 (2001).

[8] Huang, Y.S., Chung, T.H. and Lin, T. "Design and analysis urban traffic lights using timed colour Petri nets", *Proc. of IEEE International Conf. on Networking, Sensing and Control*, pp. 248–253 (2006).

[9] Davoudpour, M. and Rudakov, E.V. "Formal model of decomposition algorithm for functional block in discrete devices", *Vestnik Instruments Journal*, 1, pp. 90–98 Bauman Moscow State Technical University, Moscow, Russia (2006).

[10] Narendra, K.S. and Thathachar, M.A.L., *Learning Automata: An Introduction*, Prentice Hall (1989).

[11] Yeung, D.S., Liu, J.N.K., Shiu, J.N.K. and Fung, G.S.K. "Fuzzy coloured Petri nets in modelling flexible manufacturing systems", *ITESM*, pp. 100–107 (1996).

[12] Ouchi, Y. and Tazaki, E. "Learning and reasoning method using fuzzy coloured Petri nets under uncertainty", *Proc. of the IEEE International Conf. on Computational Cybernetics and Simulation*, 4, pp. 3867–3871 (1997).

[13] Zhou, C. and Jiang, Z. "Fault diagnosis of TV transmitters based on fuzzy Petri nets", *Proc. of the IMACS Multiconference on Computational Engineering in Systems Applications (CESA)*, pp. 2003–2009 (October 4–6, 2006).

[14] Khosla, R. and Dillon, T. "Intelligent hybrid multi-agent architecture for engineering complex systems", *International Conference on Neural Networks*, 4, Houston, TX, ISBN: 0-7803-4122-8, pp. 2449–2454 (2002).

[15] Jain, L.C. and Martin, N.M., *Fusion of Neural Networks, Fuzzy Sets, and Genetic Algorithms: Industrial Applications*, 1st Edn., CRC Press, FL, USA ISBN: 0849398045, (1998).

[16] Rutkowski, L. and Cpalka, K. "Flexible neuro-fuzzy systems", *IEEE Transactions on Neural Networks*, (ISSN: 1045-9227) 14(3), pp. 554–574 (2003).

[17] Tsuji, K. and Matsumoto, T. "Extended Petri net models for neural networks and fuzzy inference engines — their net structural properties", *IEEE International Symposium on Circuits and Systems*, New Orleans, LA, 4, pp. 2670–2673 (2002).

[18] Hadjinicolaou, M.G., Abdelrazik, M.B.E. and Musgrave, G. "Structured analysis for neural networks using Petri nets", *Proceedings by the 33rd IEEE Midwest Symposium on Circuit and Systems*, Calgary, Canada, 2, pp. 770–773, ISBN: 0-7803-0081-5, (2002).

[19] Venkatesh, K., Masory, O. and Pandya, A. "A high level Petri net model of olfactory bulb", *Proceedings of the IEEE International Conference on Neural Networks*, 2, Sanfrancisco, CA, pp. 766–771 (2002).

[20] Ahson, S.I. "Petri net models of fuzzy neural networks", *Proceedings in IEEE Transactions on Systems, Man and Cybernetics*, 35(6), pp. 926–932 (2002).

[21] Hirasawa, K., Oka, S., Sakai, S., Obayashi, M. and Murata, J. "Learning Petri network with route control", *IEEE International Conference on Man and Cybernetics, Intelligent Systems for the 21st Century*, 3, Vancouver, BC, pp. 2706–2711 (2002).

[22] Hirasawa, K., Ohbayashi, M., Sakai, S. and Hu, J. "Learning Petri network and its application to nonlinear system control", *Proceedings in IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, (ISSN: 1083-4419) 28(6), pp. 781–789 (2002).

[23] Zha, X.F., Lim, S.Y.E. and Fok, S.C. "Integration of knowledge-based systems and neural networks: neuro-expert Petri net models and applications", *Proceedings in IEEE International Conference on Robotics and Automation*, 2, pp. 1423–1428, ISBN: 0-7803-4300-X, (2002).

[24] Tsang, Eric C.C., Yeung, Daniel S. and Lee, John W.T. "Learning capability in fuzzy Petri nets", *Proceedings in IEEE International Conference on Systems, Man and Cybernetics*, 3, pp. 355–360, ISBN: 0-7803-5731-0 (2002).

[25] Li, X. and Lara-Rosano, F. "A weighted fuzzy Petri net model for knowledge learning and reasoning", *Proceedings in IEEE International Joint Conference on Neural Networks*, 4, pp. 2368–2372, ISBN: 0-7803-5529-6 (2002).

[26] Li, X., Yu, W. and Lara-Rosano, F. "Dynamic knowledge inference and learning under adaptive fuzzy Petri net framework", *Proceedings in IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 30(4), pp. 442–450 (2002).

[27] Chen, X., Jin, D. and Li, Zh. "Fuzzy Petri nets for rule-based pattern classification", *Proceedings in IEEE International Conference on Communications, Circuits and Systems and West Sino Expositions*, 2, pp. 1218–1222, ISBN: 0-7803-7547-5 (2003).

[28] Jones, I.R. and Tracy, D.P. "A multi-threaded simulator for a distributed control system", *Proceedings in IEEE International Conference on Systems, Man and Cybernetics*, 3, pp. 2272–2277, ISBN: 0-7803-7952-7 (2003).

[29] Gao, M., Zhou, M.C., Huang, X. and Wu, Z. "Fuzzy reasoning Petri nets", *Proceedings in IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, (ISSN: 1083-4427) 33(3), pp. 314–324 (2003).

[30] Hanna, M., Buck, A. and Smith, R. "Fuzzy Petri nets with neural networks to model products quality from a CNC-milling machining centre", *Proceedings in IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, (ISSN: 1083-4427) 26(5), pp. 638–645 (2002).

[31] Song, A.H., Ootsuki, J.T., Yoo, W.K., Fujii, Y. and Aekigchu, T. "FMS's scheduling by colored Petri net model and hopefield neural network algorithm", *Proceedings in International Session Papers of the 34th SICE Annual Conference*, pp. 1285-1290, ISBN: 0-7803-2781-0 (2002).

[32] Fukuda, O., Tsuji, T., Takahashi, K. and Kaneko, M. "Skill assistance for myoelectric control using an event-driven task model", *Proceedings in IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2, Lausanne, Switzerland, pp. 1445–1450, ISBN: 0-7803-7398-7 (2002).

[33] Sadeghian, A. "Nonlinear neuro-fuzzy prediction: Methodology, design & application", *Proc. of the 10th IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'2001)*, 2, pp. 1022–1026 (Dec. 2001).

[34] List, G.F. and Cetin, M. "Modeling traffic signal control using Petri nets", *IEEE Transactions on Intelligent Transportation Systems*, 5(3), pp. 177–187 (2004).

[35] Zarandi, M.H.F. and Rezapour, S. "A fuzzy signal controller for isolated intersections", *Journal of Uncertain Systems*, 3(3), pp. 174–182 (2009).

[36] Nair, B.M. and Cai, J. "A fuzzy logic controller for isolated signalized intersection with traffic abnormality considered", *Proc. of IEEE Intelligent Vehicles Symposium*, pp. 1229–1233 (2007).

[37] Li, L., Lin, W.H. and Liu, H. "Traffic signal priority/preemption control with colored Petri nets", *Proc. of the 8th International IEEE Conference on Intelligent Transportation Systems*, pp. 694–699 (2005).

[38] Zhang, L.G., Li, Z.L. and Chen, Y.Z. "Hybrid Petri net modeling of traffic flow and signal control", *Proc. of the Seventh International Conference on Machine Learning and Cybernetics*, pp. 2304–2308 (2008).

[39] Lee, J., Liu, K.F.R. and Chiang, W. "A fuzzy Petri net-based expert system and its application to damage assessment of bridges", *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 29(3), pp. 238–247 (1999).

[40] Pedrycza, W. and Camargo, H. "Fuzzy timed Petri nets", *Fuzzy Sets and Systems*, 140(2), pp. 301–330 (2003).

[41] Ting, Y., Lu, W.B., Chen, C.H. and Wang, G.K. "A fuzzy reasoning design for fault detection and diagnosis of a computer-controlled system", *Engineering Applications of Artificial Intelligence*, 21, pp. 157–170 (2007).

[42] Barzegar, S., Davoudpour, M., Meybodi, M.R., Sadeghian, A. and Tirandazian, M. "Traffic signal control with adaptive fuzzy coloured Petri net based on learning automata", North American Fuzzy Information Processing Society (NAFIPS) (2010).

[43] Fambro, D.B. and Rouphail, N.M. "Generalized delay model for signalized intersections and arterial streets", *Transportation Research Record*, pp. 112–121 (1997).

S. Barzegar was born in Mashhad, Iran, in March 1984. He received his B.S. degree from Iran University of Science and Technology in September 2006, and his M.S. degree from Qazvin Islamic Azad University in August 2010, both

in Computer Engineering. His research interests include Modeling, Learning Automata, and Fuzzy Logic.

M. Davoudpour received a B.S. degree in Electrical Engineering from Tehran Azad University, and M.S. and Ph.D. degrees in Information Technology and Computer Science from the Moscow State Technical University of Bauman in 2006.

Currently, she is with the Computer Science Department at Ryerson University, Toronto, Canada. Her research interests include Non-Linear Modeling, Petri Nets, Learning Automata, Fuzzy Logic and its Applications.

M.R. Meybodi received B.S. and M.S. degrees in Economics from Shahid Beheshti University in Iran in 1973 and 1977, respectively. He also received M.S. and Ph.D. degrees in Computer Science from Oklahoma University, USA, in 1980 and 1983, respectively. Currently, he is Full Professor in the Computer Engineering Department of Amirkabir University of Technology, in Tehran, Iran. Prior to his current position, he worked from 1983 to 1985 as Assistant Professor at Western Michigan University, and from 1985 to 1991 as Associate Professor at Ohio University, USA. His research interests include Channel Management in Cellular Networks, Learning Systems, Parallel Algorithms, Soft Computing and Software Development.

A. Sadeghian received a B.S. degree from Tehran Polytechnic University, and M.S. and Ph.D. degrees from the University of Toronto, all in Electrical and Computer Engineering. He is now with the Department of Computer Science at Ryerson University, and his research interests include Knowledge-Based Expert Systems, Neural Networks, Adaptive Neuro-Fuzzy Networks and Non-Linear Modeling.

M. Tirandazian received his B.S. degree in 1992, and M.S. and Ph.D. degrees in 1994 and 2006, respectively, from Pune University, India, all in Computer Science. He has worked on and undertaken research into many areas of computer science, namely, Computer Graphics and Artificial Intelligence, and has designed algorithms for numerous projects. His main Ph.D. research was dedicated to developing algorithms for VLSI Design and Optimization, where he has explored the advantages of embedding Boolean algebra into the Polynomial Ring. Currently, he is researching Fuzzy Logic and Its Applications at Ryerson University, Toronto, Canada.