A MULTI-AGENT TRAFFIC CONTROLLER WITH DISTRIBUTED FUZZY INTELLIGENCE

Ahmet SAHAN
Dokuz Eylul University
Dept. of Computer Engineering
İzmir – Turkey
Tel: +90 2323283339
asahan@aselsan.com.tr

Tatyana YAKHNO
Dokuz Eylul University
Dept. of Computer Engineering
İzmir – Turkey
Tel: +90 2324127401
yakhno@cs.deu.edu.tr

ABSTRACT
Instantaneous vehicle overloads on urban roads are the most critical problem of the traffic management domain. Time-scheduled traffic control systems can not handle these non-uniform congestion cases. Therefore, adaptive traffic control systems get more attraction. In this paper, a hierarchical multi-role agent traffic controller with distributed fuzzy characteristics is proposed to synchronize any number of traffic signals in a target region based on the vehicle volume of the road lanes. With the proposed controller, vehicle throughput is increased significantly, specifically compared to the pre-timed fixed traffic control systems

Keywords: Multi Agents, Fuzzy Control, Adaptive Traffic Control

1. INTRODUCTION
As the number of vehicles continues to grow, new demands are expected from current traffic systems. Roads and highways are unlikely to expand much due to cost and land supply, so intelligent systems such as advanced traffic light controls get specifically more attention to operate existing roadway systems at optimum performance. Moreover, poorly timed traffic signals can waste time, fuel and money and cause more pollution.

Many traffic light controller systems are still of the pre-timed type with fixed splits that operate different timing schedules based on time of day or a derivative of pre-timed type that is changing fixed signal time plans according to the traffic volume patterns. These types of signalization systems are generally effective when it operates with steady and predictable flow of traffic on an arterial street. However, those systems cannot respond to the dynamically changing traffic flow. It runs the same cycle length and light phases regardless of instantaneous traffic flow. This type of operation often leads to the congestion if unusual traffic patterns occur or if there are major fluctuations in traffic. Furthermore, the timing schedules in use may become obsolete unless they are checked regularly. Retiming also requires more staffing and experience that many administrations don't have.

As the computer technology has improved, fully automated models have been started to replace with pre-timed control and optimization systems. These models use historical data and some intelligence techniques to estimate optimal traffic light periods by minimizing total delay or maximizing total throughput in an intersection.

Inspired from this point, this paper introduces a new model to adaptive traffic control. The proposed model mainly focuses on a district intersection specifically and then all tightly coupled neighbor intersections adjacent to this intersection, thus providing regional and successive traffic flow management. The primary goal of the new model is to provide local optimal traffic flow through the intersection by continuously optimizing the signalization periods and then monitoring the neighbors to reach global optimality. Our solution has similarities with other agent-based, i.e. see (Lin, *et al.,* 2003), and fuzzy based solutions. However, the hierarchical agent architecture that is completely designed according to the nature of the problem, independency from intersection topologies and the fuzzy controller approach that is distributed over agents differentiate our model than others.

The paper is organized as follows. In Section 2, the preliminary research results about the proposed traffic control systems are described. The newly proposed traffic controller system model and its internals are described in Section 3 and its subsections. The implementation details, simulation and the analysis of the results of the proposed controller are given in Section 4. The conclusions are outlined in Section-5. Finally, the references are listed in the last section.

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3903
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3903

2. RELATED WORK

Since the beginning of eighties, many adaptive traffic light controller systems have already been proposed. Those systems can be classified into two distinct groups according to their approach to the problem (Van Katwijk, *et al.,* 2005) and the same classification is expressed in terms of the microscopic and macroscopic models (Wiering *et al.,* 2005). The groups are as follows:

- Vehicle-Oriented or Microscopic (Car-based) models focus on the control of behavior of individual vehicles.
- Measure-Oriented or Macroscopic (Traffic Light-based) models focus on the control of density of traffic.

Car-based models estimate waiting times for each car and after the evaluation of these waiting times, try to minimize by adjusting light periods. Applicability of these systems is very low because each car should be taken under consideration separately and some destination information should be kept to estimate waiting times of vehicles dynamically. On the other side, the models defined in the second group estimate densities of the instantaneous traffic flow and don't concentrate on waiting times of vehicles. With this approach, it tries to eliminate local congestion by predicting new light periods.

SCATS (Sydney Coordinated Traffic Adaptive System) and SCOOT (Split Cycle and Offset Optimization Technique) are currently the most popular adaptive traffic control applications (Athmaraman, *et al.,* 2005). SCATS is a dynamic control system with a decentralized architecture (Pearson, 2001). It updates intersection cycle length using the detectors. The basic traffic data used is the "degree of saturation", defined as the ratio of the effectively used green time to the total available green time. Basic limitation of this approach is that no major changes are permitted. Therefore, SCATS can not make taking the advantage of major shifts in traffic patterns.

SCOOT is a system that minimizes the sum of the average queues in an area (Pearson, 2001). At fixed intervals, it modifies the splits, offsets and cycle times of each signal. SCOOT is slightly more sophisticated design than SCATS. However, it has still the same limitation with SCATS.

In another study (Findler, *et al.,* 1992), a network of roads connected by traffic light-based expert systems is described. The expert systems can communicate to allow for synchronization. Performance on the network depends on the rules that are used. For each traffic light controller, the set of rules can be optimized by analyzing how often each rule fires, and the success it has. The system could even learn new rules. The author concludes that the proposed system could improve performance. The main problem with this solution is that too many computations exist and they had to make some simplifying assumptions to avoid them.

A similar work to our approach is given by Roozemond, see (Roozemond, *et al.,* 1995). It defines intelligent agent architecture for the traffic light control. Intelligent agents try to perform their own tasks, and try to achieve local optimality. One or more coordinator agents can communicate with the group of these agents for global performance. All agents act upon their BDI (Belief, Desire, and Intention) properties. Although the model has a well-defined architecture, no practical results are presented.

Another similarity exists in the study of Lee, see (Lee, *et al.,* 2000). Their work describes the use of fuzzy logic in controlling multiple junctions. Controllers at target point collect information at previous and next junctions, thus to provide green wave functionalities. The model outperforms a fixed light controller, and is best at both light and heavy traffic. The controller could easily handle changes in traffic flow. However it is strictly dependent on intersection topologies and requires very specific parameter modifications.

3. TRAFFIC CONTROL

In order to optimize traffic flow through an intersection or in a group of tightly coupled intersections, a combination of multi-agent paradigm and fuzzy logic controller is applied to the problem domain. Our traffic controller is a type of measure-oriented, in other words, it focuses on traffic light objects and local vehicle densities.

The general properties and assumptions are given in the following list:

- The system tries to optimize vehicle densities on target road lanes, incoming to the intersection. Optimization is applied by decreasing or increasing of red period of the traffic lights and the cycle time is assumed unchangeable. However, because of the pre-empted switches between red and green light periods, cycle-time may sometimes increase or decrease slightly
- There is no yellow light period defined in the system. It is assumed as a portion of green light. In addition to this assumption, there is no uncontrolled turn movement evaluation in this intersection model. However, if it is controlled by a dedicated traffic light, it is taken into consideration.
- Upper and lower limits for red and green light periods are pre-defined. They are initially assumed %10 of total cycle time. However, it can be changed to any rate upon request.
- There is no direct synchronization between neighbor intersections. However, on behalf of communication between intersection agents, it is assumed that there is a cascaded integrity and it may sometimes provide a kind of green wave opportunity for the vehicles, especially when one intersection has much lower density than the other one.

3.1. Multi-Agent Hierarchy

An Agent is special software that has its own life-cycle called as autonomy. It is capable of making independent decisions and taking actions to satisfy internal goals based upon its perceived environment.

Our system is composed of several agent roles and they communicate each other in the form of message communications or changes in their shared environment. There are four types of agents defined: Road Agent, Light Agent, Intersection Agent and Area Agent. The general system view is given in Figure 1.

The objective of constructing a hierarchy is to decrease the complexity of control. Each upper agent controls its lower level agents by means of peer to peer communication methods. The agent hierarchy of an intersection is defined based on its topology.

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3903
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3903

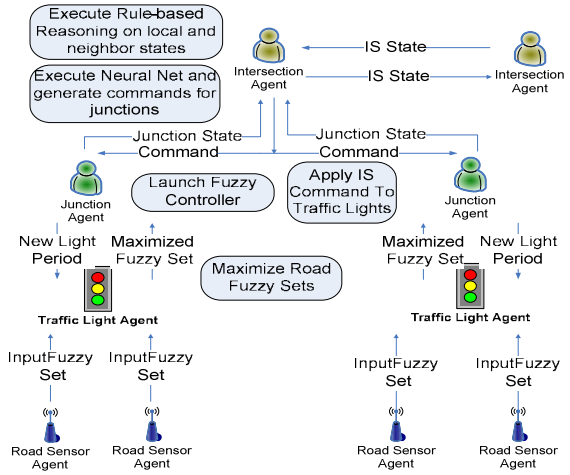

Figure 1. The general system view

Starting from the bottom layer, Road Agent represents one-way lane traffic density sensor. Light Agent represents the manager of coupled road lanes on correlated directions. A Light Agent may have more than one Road Agent. A junction is a road cross between two directions in an intersection. Junction Agent represents the first transition controller on traffic lights and new light periods are basically decided by this agent. Each Junction Agent has at least 2 Light Agents. However, if light controlled turn movements are taken consideration, it can be more. Intersection Agent represents a bridge between local and global states of the traffic network. An intersection may have more than one junction if many tightly coupled junctions exist in an intersection. All Junction Agents in an intersection are controlled by the same Intersection Agent.

3.2. Distributed Fuzzy Controller

Road Agents, located at the bottom of the hierarchy, sense the target road lane volume. They can be thought as road volume detectors in hardware level. However, in this model, the hardware side is completely transparent: it can be Radar, Camera or any other device providing road traffic density. Light Agents periodically send a request to its Road Agents and they generate instant vehicle density in percentage unit and then this data is fuzzified according to the Figure 2.

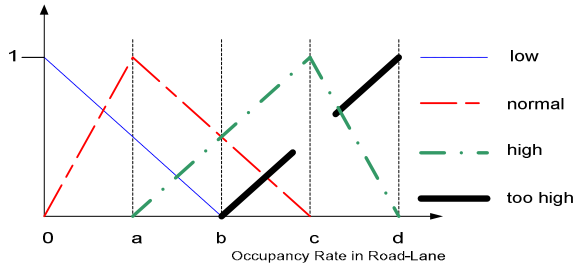

Figure 2. The input fuzzy set (d>b>c>a in [1,100])

As it is seen from Figure 2, the fuzzy input set consists of Low, Normal, High and Too High values. The a, b, c, and d values represent the fuzzy set transition points of the traffic volumes. Although they are pre-defined values (a: %25, b: %50, c: %75, d: %100), they are up to change and note that these values are applied at system level.

After Light Agent collects all fuzzy data sets of dependent Road Agents, it forwards them into a maximization function where the highest fuzzy values of members are selected for Light Agent's corresponding final member set values.

A Junction Agent collects the fuzzy data sets of its slave Light Agents periodically and launches the fuzzy controller on received data sets. Inputs of fuzzy controller consist of Reference and Opponent Light Agent group fuzzy states. The classification of Light Agents as Reference or Opponent is done by a pre-configured table and this property is just a matter of selection during the initial configuration of the system.

Input values are then processed by the fuzzy controller. There are sixteen rules applied on input sets by the controller. Some are given below:

1) if Reference is Low and Opponent is Normal then Do Nothing
2) if Reference is Too High and Opponent is Too High then Do Nothing
3) if Reference is Low and Opponent is High then change Reference Red period in Positively Medium
4) if Reference is High and Opponent is Too High then change Reference Red period in Positively Small
5) if Reference is Too High and Opponent is High then change Reference Red period in Negatively Small

The output of the each fuzzy rule becomes the fuzzy change type of red light period for Reference group. The output fuzzy set is given in Figure 3. The values: x, y and z represent the highest points of output fuzzy sets correspondingly. Although they are pre-defined values (x: %25, y: %50, z: %75), they are up to change and note that these values are applied at system level.

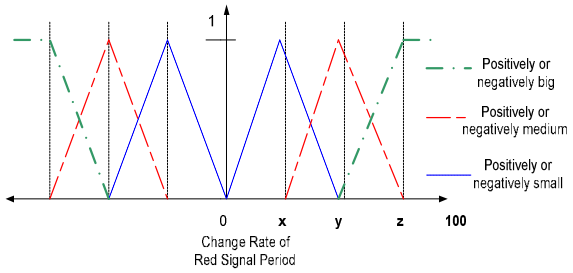

Figure 3. The output fuzzy set (z>y>x in [1,99])

Since the result of Fuzzy Controller is still in fuzzy, one more step is required to complete new light period prediction cycle. To get a crisp value, de-fuzzification is applied. It is implemented using Weighted Centroid Method. With this method, the output is obtained by combining the centroid values of output fuzzy sets that are stored in the knowledge based of the system and result of each fuzzy rule input variables. The weighted average de-fuzzification equation can be expressed as

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3903
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3903

$$x^* = \frac{\sum_{i=1}^{n} m^i w_i}{\sum_{i=1}^{n} m^i} \quad (1)$$

Where x^* is the defuzzified output, m^i is the membership value of the output of each rule (left size of the rule, it refers to minimum one of input values), and w_i is the weight associated with each rule. *Each* w_i value represents x-coordinate centroid value of output fuzzy set. This method is selected for defuzzification because of its high computational speed and accuracy rate.

After the above formula is applied to the fuzzy rules, the final output value, x^* that refers to the estimated change rate of reference red light period is obtained.

3.3. Reasoning at Intersection

Intersection Agent controls pre-defined Junction Agents in a tightly coupled road crosses. It can be seen as a Super-junction Agent. All Junction Agents send their local state information to their Intersection Agent. Moreover, Intersection Agents adjacent to the target Intersection Agent send their states periodically. Neighboring relations are based on previous-next position of the junctions and intersections. The Intersection Agent evaluates the collected local and global state information to generate the increase or decrease commands of traffic light periods that will be sent to its Junction Agents.

Intersection Agent keeps two configuration tables. First one simply refers to the list of neighbor intersections. This table is used to send local intersection state to neighbors. The other relation table, its structure given in Table 1, is about the links between junctions in an intersection and between intersections. The data in this table are used at reasoning phase to help sub-junctions or neighbors in congested situation. (The agent is in High or Too High state in fuzzy terms)

Table 1 The junction to junction and junction to neighbor intersection connection list template

Junction and Edge	Linked To	Edge Type	Position Type
Junction-X, Ref. or Opponent	Junction or Neighbor	Neighbor or Junction	Previous or Next

Each intersection agent launches an algorithm after collecting local junction and neighbor intersection states. According to the algorithm, the local state has priority over neighbors' states. In other words, if the local state is in high or too high state, Intersection agent focuses to solve local congestion and ignores the neighbors' states. The evaluation cycle of the algorithm is as follows:

```
//Domain refers to Neighbor or Local Junction
For i 1 TO N (Number of Domain)
If (Domain[i].State is High or Too High)
  //Link[] refers to Table-1.
  Link[] = GetLinks(Domain[i])
  For j 1 TO N (Number of Links)
    XState = Link[j].State; XEdge = Link[j].Edge
    XPos = Link[j].Pos; XJunc = Link[j].Name
    If (XState is not Too High)
     XRes = abs(Domain[i].State – XState)* XRate
     Cmd[ XJunc] += XRes * XVal[XPos] [XEdge]
    End if
   End For
End if
End For
```

"XState" and Domain[i].State refer to one of the fuzzy states (from Low: 0 to Too High: 3). "XEdge" refers to a light agent group (Reference or Opponent). "XPos" refers to the position of the domain according to the link object (previous or next). "XJunc" refers to a Junction that is connected to target domain type and its state is eligible to help target object in trouble. "XVal" array is a pre-constructed table based on "XEdge" and "XPos" value matrix. It can be either -1 or 1. "XRate" is a coefficient to factorize command between 0 and +/-1 and finally "XRes" refers to the degree of command.

3.3. Learning Support by Neural-Net

In order to provide learning ability at local intersection level and attach other environmental parameters into the new light period decision making cycle, a neural network layer is designed for each Intersection Agent. The neural net layout is selected as back-propagation so that the supervised learning can be applied. In order to obtain accurate and reliable results, the training data must be well produced in high and effective density. Table 2 shows the input parameters of the neural network.

Table 2. The input parameters of the designed neural net

Neural Net Input Parameters
Command Rate coming from Reasoning Engine $-1 \leq fCommand < 1$
Target ID Type (Local Junc or Neighbor) Either 0 or 1
IsHoliday: 0 or 1
DayOfWeek: 001 to 111 (from Monday to Sunday)
Hour of Time: 00000 to 10111 (from 0 to 23)
Minute of Time: 000000 to 111011 (from 0 to 59)
Junction ID to give support (2^n+1 input neurons, refers to number of junctions in target intersection)
Target ID to get help (2^n+1 input neurons, refers to maximum number of junctions or neighbors for target intersection)

All these inputs attached to the input layer of designed neural network are connected to the neurons in hidden layer and the output of hidden nodes is connected to the neuron in output layer. The result of the output node represents the command change rate for the red light period of the target Junction Reference Group. The generated command rates are then sent to the related Junction Agents. Each command is directly put in charge by Junction Agent to accelerate traffic.

Neural net implementation enables the system to keep only the configuration data and the historical data is not needed. Neural net keeps the history in its internal structure.

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3903
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3903

4. IMPLEMENTATION

After completing the design cycle of the proposed model, the software development and simulation phases have been implemented. The software coding of this project has been done on Windows XP platform by using Java programming language facilities. Java is the preferred programming language because of its platform independence property overall operating systems and its object-oriented nature that is very close to the agent system paradigms. Moreover, more components and libraries are available on Java platforms.

In order to build agents, an agent programming environment called Jade was used. Jade (Java Agent Development Framework) is compliant with FIPA (Foundation for Intelligent Physical Agents) specifications and free software with GNU public license, see (Bellifemine, *et al.,* 2003). Moreover, in order to generate neural-network components, another free software library, Joone (Java Object Oriented Neural Engine) was used. See (Marrone, 2005) for more details about Joone and its architecture..

4.1. Simulation Results

In order to get some results and make some comparisons, the system was tested for a selected intersection scenario. The topology of the scenario to be simulated is illustrated in Figure 4.

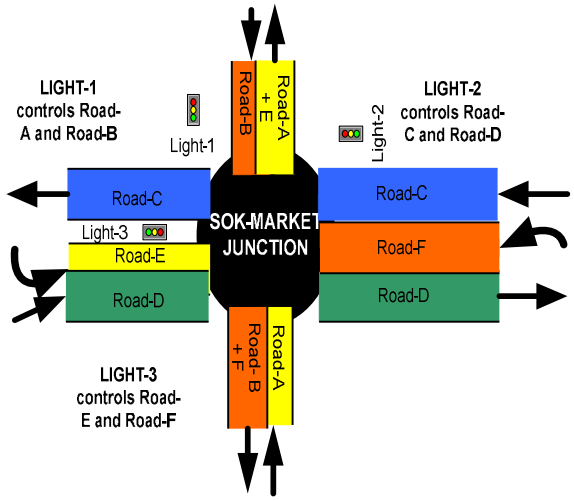

Figure 4. The sample intersection scenario

Initial test configuration is as follows: Total Cycle is set to 90 seconds. The data collection frequencies from sub agents are 30, 45, and 60 seconds corresponding to the Light, Junction and Intersection Agents. A vehicle can pass through the intersection in 1 second: in other words, the leak rate of an intersection is 1veh/sec. The input data tables show the number of vehicles driving on target road lane.

As it is seen from the figure, there are three lights available here. The classical junction definition, given in this proposed model controls two traffic lights. However, here 3 traffic lights are available: 2 for directional control and 1 for turn movement. The controller resolves it by applying fuzzy logic to 3 double combinations of these 3 traffic lights and applying average function to get final traffic light periods. Based on these settings and according to the given data sample set in Table 3, the Figure 5 graph is produced.

Table 3. The sample data input set-1

Time (min)	Road-A	Road-B	Road-C	Road-D	Road-E	Road-F
1	20	40	20	60	40	10
2	20	40	20	55	40	15
3	20	40	20	50	40	20
4	20	40	20	45	40	25
5	20	40	20	40	40	30
6	20	40	20	35	40	35
7	20	40	20	30	40	45
8	20	40	20	25	40	50
9	20	40	20	15	40	55
10	20	40	20	10	40	60

In this table, Road-A, Road-B have steady traffic volume, and the others have dynamically changing traffic volume. The result graph (Figure 5) shows that our adaptive light controller outperforms the fixed pre-timed light controller. Moreover, it is successful to distribute queued vehicle count on road lanes uniformly.

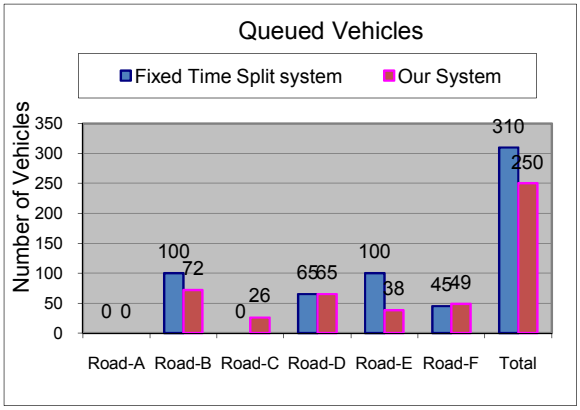

Figure 5. The result graph of the data set-1

Another data set and the output graph for the selected traffic network are illustrated in Table 4 and Figure 6 correspondingly.

Table 4. The sample data input set-2

Time (min)	Road-A	Road-B	Road-C	Road-D	Road-E	Road-F
1	40	50	10	60	10	10
2	40	50	15	55	15	15
3	40	50	20	50	25	20
4	40	50	25	45	30	25
5	40	50	30	40	35	30
6	40	50	35	35	40	35
7	40	50	40	30	45	45
8	40	50	45	25	50	50
9	40	50	55	15	55	55
10	40	50	60	10	60	60

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3903
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3903

As it is seen from the graph in Figure 6, our system outperforms fixed-time split system once again.

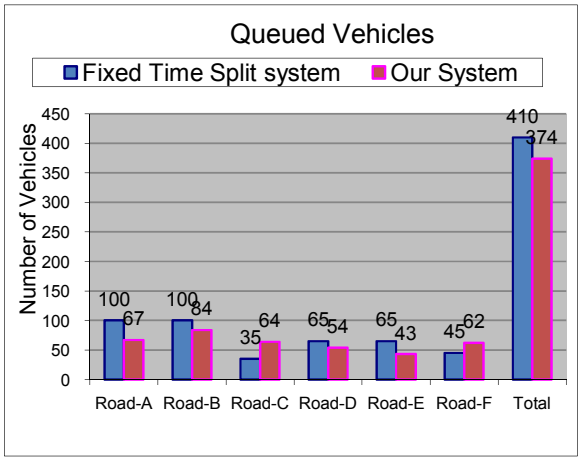

Figure 6. The result graph of the data set-2

According to the obtained results in simulation tests, it is seen that our adaptive traffic controller has high efficiency, specifically when one road direction has low level steady vehicle volume and other road has high level steady vehicle volume. Moreover, if the one road direction has high level steady vehicle volume and the other road direction has rising vehicle volume curve, the controller boosts the performance.

If both road directions have parallel characteristics in rising or falling vehicle volume level, then the controller seems a pre-timed controller. On the other side, If the one road direction has low level steady vehicle volume or rising vehicle volume curve and the other road direction has falling vehicle volume curve, it performs a bit worse than pre-timed controller.

If the traffic pattern changes occur sharply (once to up, then immediately to down) and happen in a very short time that is less than the sensor data collection and other evaluation periods, the new controller cannot handle the traffic and may show poor performance.

As a result, it is seen that this controller could be very productive for intersections which have characteristics like that if one direction has low or rising volume and the other one has high volume in local junction decision level. In addition to this property, it is seen that when the controller reaches the limits of traffic light periods, it stays at that level until the other road direction gets congested.

5. CONCLUSION

With all this completed study, it was shown that the proposed adaptive traffic controller solved the congestion problems efficiently at micro control (basic junction) level. It gives the best performance when one direction has low straight vehicle volume level and the other one has high steady or continuously changing vehicle volume curve, either in increasing or decreasing mode.

The proposed solution is quite different from many other adaptive models in terms of the independent nature of the intersection topologies, neuro-fuzzy approach embedded into multi-role agents and global state evaluations that are taken into the decision mechanisms of the local level agents.

As a future study, some enhancements could be applied to the solution model. Neural Network internals are well defined and integrated to the Intersection Agent. However, it may still require refinement and more training data to give accurate and reliable results. Another enhancement is possible through the comparisons of different and complex scenario implementations with the other adaptive systems. In order to succeed it, the popular simulation tools such as TSIS (Traffic Software Integrated System) simulation program, see (Owen, *et al.,* 2000), that contains CORSIM traffic simulator tool can be employed. Thus, the strength of the proposed solution might be better appreciated.

6. REFERENCES

[1] Athmaraman, N., Soundararajan, S., (2005). Adaptive Predictive Traffic Timer Control Algorithm. *Proceedings of the 2005 Mid-Continent Transportation Research Symposium,* **ITS Section.**

[2] Bellifemine, F., Caire, G., Trucco, T., Rimassa, G. (2003), *Jade Programmers Guide v3.0b1*, Retrieved January 15, 2005, from http://jade.cselt.it, TILab S.p.A and University of Parma, Italy.

[3] Findler, N., Stapp, J., (1992). A distributed approach to optimized control of street traffic signals. *Journal of Transportation Engineering, 118*, **No. 1,** 99110.

[4] Lee, J., Lee, K., Seong, K., Kim, C., Lee-Kwang, H. (1995). Traffic Control of Intersection Group Based On Fuzzy Logic, *Int. Conf. 6th IFSA '95, Sao Paulo, Brazil, P. 465 – 468.*

[5] Lin, S., Chen, D., Chen, R. (2003). Apply Adaptive and Cooperative multi-agent system to urban traffic signal control, *11. IEEE Mediterranean Conference on Control and Automation,* **T4-010**.

[6] Marrone, P. (2005), *The Joone Complete Guide*, Joone's Team, Retrieved November 15, 2005, from http://www.joone.org

[7] Owen, L., Zhang, Y., Rao, L. and McHale, G. (2000). Traffic Flow Simulation Using Corsim. *Proceedings of the 2000 Winter Simulation Conference,* **vol.2,** pages 1143-1147.

[8] Pearson, R. (2001). *ITS- Traffic Signal Control.* California Center for Innovative Transportation. Retrieved May 15, 2005, from *http://www.calccit.org/itsdecision/serv_and_tech/Traffic_signal_control/traffsigrep_print.htm*

[9] Roozemond, D.A., Rogier, J.L.H. (2000). Agent controlled traffic lights, *In European Symposium on Intelligent Techniques*, **AD-01,** p 77-82.

[10] Van Katwijk, R.T., P. Van Koningsbruggen, B. De Schutter, and J. Hellendoorn, (2005). Test bed for multiagent control systems in road traffic management, *Transportation Research Record: Journal of the Transportation Research Board*, **no. 1910,** pp. 108–115.

[11] Wiering, M, van Veenen J., Vreeken J., and Koopman A. (2004), *Intelligent Traffic Light Control,* Technical Report UU-CS-2004-029, Utrecht University, Holland.

Digital Object Identifier: 10.4108/ICST.MOBIQUITOUS2008.3903
http://dx.doi.org/10.4108/ICST.MOBIQUITOUS2008.3903