Adaptive Traffic Control System based on Bayesian Probability Interpretation

Mohamed A. Khamis*, Walid Gomaa*,†, Ahmed El-Mahdy*,†, and Amin Shoukry*,†
*Department of Computer Science and Engineering
Egypt-Japan University of Science and Technology (E-JUST), Alexandria, Egypt
{mohamed.khamis, walid.gomaa, ahmed.elmahdy, amin.shoukry}@ejust.edu.eg

***Abstract*—Traffic control (TC) is a challenging problem in today's modern society. This is due to several factors including the huge number of vehicles, the high dynamics of the system, and the nonlinear behavior exhibited by the different components of the system. Poor traffic management inflicts considerable cost due to the high rate of accidents, time losses, and negative impact on the economy as well as the environment. In this paper, we develop a traffic control system based on the Bayesian interpretation of probability that is adaptive to the high dynamics and non-stationarity of the road network. In order to simulate the traffic non-stationarity, we extend the Green Light District (GLD) vehicle traffic simulator. The change in road conditions is modeled by varying vehicle spawning probability distributions. We also implement the acceleration and lane changing models in GLD based on the Intelligent Driver Model (IDM).**

***Index Terms*—reinforcement learning, traffic control, traffic simulation, multi agent systems, driver behavior**

I. Introduction

In Egypt, traffic problems are responsible for more than 25000 accidents in 2010 [1] with more than 6000 deaths per year. Moreover, deaths per million vehicle kilometers in Egypt is about 34 times higher than in the developed countries. This value is about three times higher than countries in the Middle East region [2].

Recently, some computer science tools and technologies are used to address traffic control complexities. Among these is the multi-agents system framework whose characteristics are similar in nature to the traffic problem [3] [4]. Such characteristics include distributiveness, autonomy, intelligibility, on-line learnability, and scalability.

In this paper, we develop a traffic control system using Bayesian interpretation of probability that is adaptive to the high dynamics and non-stationarity of the road network. In order to simulate the traffic non-stationarity, we extend the GLD vehicle traffic simulator. The change in road conditions is modeled by varying vehicle spawning probability distributions. We also implement the acceleration and lane changing models in GLD based on IDM.

The remaining of this paper is organized as follows, the related work is discussed in section II. A background on the GLD traffic simulator and control as well as a background on the IDM driver model is presented in section III. Our adaptive traffic control system based on Bayesian probability interpretation is described in section IV. System performance evaluation is presented in section V. Finally, section VI concludes the paper.

II. Related Work

There exist several approaches to implement an adaptive traffic control. These approaches are based on Dynamic Programming, Reinforcement Learning (RL), Fuzzy Logic, Evolutionary Algorithms, and Artificial Neural Networks.

In [5], Sen and Head present a controlled optimization of traffic phases for traffic control using a dynamic programming algorithm. The proposed algorithm optimizes traffic control on only one intersection.

RL for traffic light control was first studied in [6]. The approach is based on a State-Action Reward-State Action (SARSA) RL algorithm. A discrete state-action space is used to represent the states and actions. A state is described by the queue length, the vehicles positions in the queue, and the elapsed time of the current traffic light. A single controller is trained on a single intersection. After training is completed, the state of this single controller is replicated to all intersections of the traffic network. This system outperformed both fixed and rule-based controllers in some realistic simulation with changing speed.

In [7], Ma et al. applied Q-learning to dynamically control traffic signals at an isolated intersection. The traffic state includes the green phase index, the green light lasting time, traffic volume during green phase, queuing vehicles mean number during red phase, and the traffic flow trend prediction. The learner action is changing the green phase to the red phase, or extending the green phase until the next decision point. The reward is the traffic volume during the green phase divided by the waiting time increase during the red phase.

In [8], Abdulhai et al. applied Q-learning to a traffic signal control system. For an isolated intersection, a state includes the queue lengths on the four main roads and the elapsed phase time. The actions include extending the current phase or leaving to the next one. The reward (a penalty in this case) is the total delay happened between successive decision points by vehicles in the queues of the four main roads. The delay is a power function of the queue length. The reward is the weighted summation of the rewards of all isolated intersections. The reward of the main road is weighted more heavily.

The methods described above suffer from the growth in the number of states when scaling to larger networks even when

†Currently on-leave from the Faculty of Engineering, Alexandria University.
This work is partially funded by IBM PhD Fellowship and Faculty Award.

978-1-4673-0483-2

some state information is excluded (e.g. the controller uses only the queue length). Thus, those methods were only applied to relatively small traffic networks (e.g. a single intersection) or training a single intersection and using the same policy for a number of intersections.

We adopt a different approach that is a vehicle-based controller [9]. In this approach, a predictor for each vehicle is used to estimate the waiting time of the vehicle alone when the light is green or red. All vehicle predictions are then combined to make the decision of the traffic light. In the next section, a background on the adopted traffic simulation and control as well as the IDM driver model is presented.

III. Background

A. Traffic Simulation Model

In order to test our traffic control algorithm there must be some experimentation platform, that is, a traffic simulator. There are several research traffic simulators in the literature [10] [11]. However, they are still primitive and preliminary in their capabilities with a lot of oversimplifications. One of the objectives of the work presented in this paper is to extend and significantly enhance one of these simulators, namely MoreVTS [11] that is based on the GLD traffic simulator [10]. Extensions include modeling the load generation by varying vehicle spawning probability distributions, acceleration and lane changing that is used also to model the driver behavior.

GLD is a microscopic traffic simulation model such that it models the behavior of the individual vehicles. The simulation in GLD is based on cellular automata where discrete partially connected cells occupy various states [12]. A road cell can either be occupied by a vehicle or be empty. The infrastructure mainly consists of roads and nodes. Any road connects two nodes and can have several lanes in each direction. The length of any road is expressed in cells. A node can be either a junction where traffic lights are acting or an edge node.

Two types of agents occupy this infrastructure: vehicles and traffic lights controllers. Agents act autonomously and are updated every time step. Vehicles enter the traffic network at edge nodes. Each edge node has a probability of spawning a vehicle at every time step (1 means spawning a vehicle at every time step). A destination from the other edge nodes is assigned to each spawned vehicle. The GLD user can adjust the distribution of destinations for each edge node. Every time step, there exist three possibilities; each vehicle can wait at a specific traffic light, can drive to the next position in the same road-lane, or can cross a junction and go to another road-lane towards its destination.

Each traffic light controller can take some decisions representing the possible traffic light configurations that do not lead to any possible accidents between vehicles at the controlled junction. Consider a 4 roads (each of 4 lanes) junction, the ingoing lanes in each road are one lane for going left and one lane for going straight/right. There exist 8 possible decisions (4 decisions: in each road allow both traffic lights to be green for left and straight/right and 4 decisions: in each opposing roads allow both traffic lights to be green for left or straight/right).

The main objective in GLD is to determine for each traffic light either to be red or green in order to minimize the cumulative waiting time of all vehicles for all traffic lights met before exiting the city [9]. This decision depends on a cumulative vote, where each vehicle votes by its estimated advantage (or gain) of setting the traffic light for which it is currently standing to green. The gain value is the difference between the total time the vehicle expects to wait during the rest of its trip if its traffic light is red, and if it is green.

B. Reinforcement Learning for Urban Traffic Control

In this subsection, we discuss Wiering RL approach [9] for traffic control. RL is used to make the traffic light agent learns control by letting the vehicle agents interact with its environment and learn from the obtained feedback (reward). Using a trial-and-error process, the agent will learn a policy that optimizes the cumulative reward it gains over time.

The action, decision or light L means the traffic light is set to green or red. The state means each vehicle is at a traffic node $node$, a direction dir at that node, a position pos in the queue, and has a destination des address. We write $[node, dir, pos, destination]$ to denote the state, that is shortly denoted by $[n, d, p, des]$. The probabilistic state transition function is represented by a lookup table consisting of the probabilities $P([n, d, p, des], L, [n', d', p'])$ where L denotes whether the light at $[n, d]$ is red or green.

The probability that the light is red or green is given by $P(L|[n, d, p, des])$ for a vehicle waiting at $[n, d, p]$ with a specific destination des. The reward function is given by $R([n, d, p], L, [n', d', p']) = 1$ if a vehicle stays at the same position, otherwise $R = 0$ (vehicle moves to the next position).

For computing $P(L|[n, d, p, des])$, some counters are updated after each time step. $C([n, d, p, des], L)$ counts the number of times the light is red or green when a vehicle is standing at $[n, d, p]$ with a specific destination des. $C([n, d, p, des])$ counts the number of vehicles standing at $[n, d, p]$ with a specific destination des. This probability is computed as following:

$$\Pr(L|[n, d, p, des]) = \frac{C([n, d, p, des], L)}{C([n, d, p, des])}$$

The state transition probability is computed as following:

$$\Pr([n, d, p, des], L, [n', d', p']) = \frac{C([n, d, p, des], L, [n', d', p'])}{C([n, d, p, des], L)}$$

The total estimated waiting time at all traffic lights for a vehicle at $[node, dir, pos]$ until it reaches its destination des given the traffic light action is given by $Q([node, dir, pos, destination], action)$. This value is shortly denoted by $Q([n, d, p, des], L)$. $V([n, d, p, des])$ denotes the average waiting time for a vehicle at $[n, d, p]$ until it reaches its destination des without knowing the traffic light decision.

The Q-function for the original method TC-1 [9] is computed as following:

$$Q([n, d, p, des], L) = \sum_{(n', d', p')} \Pr([n, d, p, des], L, [n', d', p']) \\ (R([n, d, p], L, [n', d', p']) \\ + \gamma V([n', d', p', des])),$$

Where γ is the future discount factor ($0 < \gamma < 1$) to ensure that Q-values are bounded. The V-function is computed as following:

$$V([n,d,p,des]) = \sum_{L} \Pr(L|[n,d,p,des])Q([n,d,p,des],L)$$

The traffic light controller sums up the individual advantages of all vehicles standing at the traffic lights i set to green by the traffic node n decision (all other traffic lights are red). If the vehicle is neither standing on the first position nor waiting in the queue (i.e. the vehicle is still driving until it stands in the queue $(n,d,p,des) \notin Q_i$), the vehicle is not allowed to vote for the total waiting time. The gain is computed as following:

$$A_n^{opt} = arg\max_{A_n} \sum_{i \in A_n} \sum_{(n,d,p,des) \in Q_i} Q([n,d,p,des],red) - Q([n,d,p,des],green)$$

C. Acceleration and Lane Changing Models

In Wiering simulation [9] [10], once a lane is selected, the vehicle cannot switch to a different lane.

We implement the acceleration and lane changing models in GLD based on the IDM driver model [13] that is used to simulate the longitudinal dynamics, i.e. accelerations and braking decelerations of the drivers [14].

In [15], Helbing cited IDM by: *"It is easy to calibrate, robust, accident-free, numerically efficient and yields realistic acceleration and braking behavior. Therefore, it reproduces the empirically observed phenomena."*

The acceleration dv/dt of a vehicle depends on its velocity v, the distance s to the front vehicle, and the velocity difference Δv (positive when approaching the front vehicle),

$$\frac{\mathrm{d}v}{\mathrm{d}t} = a\Big[1 - \Big(\frac{v}{v_0}\Big)^{\delta} - \Big(\frac{s^*}{s}\Big)^2\Big],$$

Where

$$s^* = s_0 + min\Big[0, \Big(vT + \frac{v\Delta v}{2\sqrt{ab}}\Big)\Big]$$

The acceleration consists of two terms: (1) the desired acceleration $a[1-(\frac{v}{v_0})^{\delta}]$ on a free road, (2) braking decelerations due to the front vehicle.

The parameters of the IDM driver model are [14]: (1) desired velocity on a free road, v_0, (2) desired safety time headway, T, (3) acceleration in usual traffic, a, (4) braking deceleration in usual traffic, b, (5) minimum bumper-to-bumper distance, s_0, (6) acceleration exponent, δ.

Lane changes depend on two criteria [16]: (1) A safety criterion is satisfied if the IDM braking deceleration of the back vehicle on the target lane B' after a possible change $acc' = acc'_{IDM}$ does not exceed a certain threshold b_{save}: $acc'(B') > -b_{save}$. (2) An incentive criterion is evaluated by weighting *my own advantage* on the target lane measured by the increased acceleration or decreased braking deceleration $acc'(M') - acc(M)$ against the *disadvantage of other drivers* $[acc(B) + acc(B')] - [acc'(B) + acc'(B')]$ weighted with a *politeness factor p* whose values are typically less than 1: $acc'(M') - acc(M) > p[acc(B) + acc(B') - acc'(B) - acc'(B')] + a_{thr}$

An additional lane-changing threshold a_{thr} has been added to the balance of the above equation.

A study made in the United States by [17] mentions that careless lane-changing is the 10^{th} position out of 25 driver's behavior as the cause of accidents happened in highway.

Different human driving behaviors can be modeled by varying the politeness factor p [16]: (1) $p > 1$: a very humane behavior. (2) $p \in]0, 0.5]$: a realistic behavior such that advantages of other drivers have a lower priority but are not neglected. (3) $p = 0$: a purely selfish behavior. (4) $p < 0$: an aggressive personality who discomforts other drivers even at the cost of own disadvantages.

IV. Proposed Traffic Control System

In this section, the details of our adaptive traffic control system based on Bayesian probability interpretation and simulating the non-stationarity in road conditions are presented.

A. Accidents and Congestion Models

In order to simulate the traffic non-stationarity, we extend the GLD vehicle traffic simulator. The change in road conditions is modeled by varying the vehicle spawning probability distributions.

We categorize the vehicle spawning distributions as following: (1) fixed (no inter-arrival probability distribution), or probabilistic (not fixed; one of our contributions), (2) static (the same vehicle spawning distribution for all time intervals), or dynamic (not static; one of our contributions in the probabilistic case).

We check if the generated random number after the last arrival equals one time step or if the time step of the last arrival plus the random number generated after the last arrival equals the current time step, we do the following: (1) spawn new vehicle, (2) set the time step of the last arrival to the current time step, and (3) generate a new random number following the specified inter-arrival distribution.

For modeling the vehicle inter-arrival probability distribution, we use the following continuous distributions: Uniform, Triangular, Exponential, Erlang, Weibull, and Gaussian.

Examples on the dynamic spawning probability distributions:

First Example: Initially (1) the current time step is zero, (2) the time step of the last arrival is the current time step minus one i.e. equals -1, (3) the last generated random number is one that means that the inter-arrival time between the two vehicles is one time step. Thus, the time step of the last arrival plus the last generated random number is zero (that is the current time step). Then, a vehicle is spawned at time step zero and the time step of the last arrival is the current time step.

Second Example: (1) the current time step is 130, (2) the time step of the last arrival is 120, and the spawning frequencies are defined as following: (i) from time step 120 to time step 129, the spawning frequency is fixed to 0.4, (ii) from time step 130 to time step 140, the inter-arrival distribution is $\mathcal{U}(a = 0, b = 1)$, and the generated random number is one. Then, a vehicle is spawned at time step 130.

Third Example: (1) the current time step equals 120, (2) the time step of the last vehicle is 100, and the spawning frequencies are defined as following: (i) from time step 100 to time step 115, the inter-arrival distribution is $\mathcal{N}(\mu = 20, \sigma = 0.5)$, and the generated random number is 20, (ii) from time step 115 to time step 120, the inter-arrival distribution is $Weibull(k = 1, \lambda = 1)$, and the generated random number is 3. Then, two vehicles are spawned at time steps 120 and 123.

Following the Poisson process, the inter-arrival time can not be equal zero, i.e. no two vehicles can be spawned in the same time step on the same lane.

B. Adaptive Traffic Control based on Bayesian Probability Interpretation

In [18], Lior discusses the stationary versus the non-stationary environments. In a stationary environment, as an assumption the dynamics is always fixed. This means that the transition probability $\Pr(s'|s,a)$ is fixed such that the next state s' is either unique for a given action a and state s or there is some fixed probability distribution over the possible next states which is fixed over time.

This assumption is violated in real-world systems where $\Pr(s'|s,a)$ for a given action a and state s changes over time due to the change in the environment dynamics. In the non-stationary environment, agents will need to learn the whole history even for environment dynamics which have been previously experienced since the policy that was computed is no longer valid when the dynamics change.

In [9][10], Wiering et al. learn the transition probability functions $\Pr(s'|s,a)$ and $\Pr(a|s)$ by counting the frequency of observed experiences as mentioned in the previous section. We handle the current weak adaptation that depends only on counting the number of vehicles facing the same situation.

Each next state s' has a bias probability $\Pr(s'|s,a)$ for a given action a and state s where the summation of those probabilities equals one. We implement the case where $\Pr(s'|s,a)$ changes over time due to the change in the dynamics of the environment. To make the transition probability non-stationary (adaptive), we estimate the weights of the next states/actions using the Bayesian learning depending on the current road conditions.

The possible next states s' bias probabilities can be simulated by changing the corresponding spawning frequency e.g. decreasing the traffic in some situation (one possible next state) while increasing the traffic in other situation (other possible next state). Our derivation to the posterior transition probability as a function of the simulation time step t is listed below.

The primitive Bay's rule where A represents the posterior probability and B is the set of observed experiences $x_i's$ is given by:

$$\Pr(A|B) = \Pr(B|A)\Pr(A)/\Pr(B)$$
$$Posterior(t+1) = \frac{Likelihood(t+1)Prior(t+1)}{Normalizingfactor(t+1)};$$
$$Prior(t+1) = Posterior(t)$$

P_t could be (1) the posterior probability when fixing $s = (n, d, p, des)$ with different a or s', or (2) the posterior probability when fixing (s,a) with different s'. In case (1) the observed experience x_i equals 1 when the same situation (s,a,s') occurs and equals 0 when the same s with different situation a or s' occurs. t equals the number of times the same situation s occurs. In case (2) x_i equals 1 when the same (s,a,s') situation occurs and equals 0 when the same (s,a) with different s' situation occurs. t equals the number of times the same situation (s,a) occurs:

$$\Pr(P_t|(x_1,x_2,\ldots,x_{t+1})) = \frac{\Pr((x_1,x_2,\ldots,x_{t+1}|P_t)\Pr(P_t)}{\Pr(x_1,x_2,\ldots,x_{t+1})}$$

The joint probability equals $\prod_{i=1}^{t+1} P_t^{x_i}(1-P_t)^{(1-x_i)}$, then the $Posterior(t+1)$ is given by:

$$\begin{aligned} f(t+1,P_t) &= \frac{L(t+1,P_t)f(t,P_t)}{\prod_{i=1}^{t+1} P_t^{x_i}(1-P_t)^{(1-x_i)}} \\ &= \frac{\prod_{i=1}^{t+1} L(i,P_t)f(0,P_t)}{\prod_{i=1}^{t+1} P_t^{x_i}(1-P_t)^{(1-x_i)}}; f(0,P_t) = 1 \\ &= \frac{\prod_{i=1}^{t+1}\prod_{j=1}^{i} P_t^{x_j}(1-P_t)^{(1-x_j)}}{\prod_{i=1}^{t+1} P_t^{x_i}(1-P_t)^{(1-x_i)}} \end{aligned}$$

For an easier differentiation, we find $\ln f(t+1,P_t)$:

$$\begin{aligned} \ln f(t+1,P_t) &= \sum_{i=1}^{t+1} \ln(\prod_{j=1}^{i} P_t^{x_j}(1-P_t)^{(1-x_j)}) \\ &\quad - \sum_{i=1}^{t+1} \ln(P_t^{x_i}(1-P_t)^{(1-x_i)}) \\ &= \sum_{i=1}^{t+1}\sum_{j=1}^{i}\Big[x_j \ln P_t + (1-x_j)\ln(1-P_t)\Big] \\ &\quad - \sum_{i=1}^{t+1}\Big[x_i \ln P_t + (1-x_i)\ln(1-P_t)\Big] \end{aligned}$$

Differentiating with respect to P_t and equating to 0, where $\ln f(t+1,P_t)$ and consequently $f(t+1,P_t)$ are maximum:

$$\begin{aligned} \frac{\partial \ln f(t+1,P_t)}{\partial P_t} &= \sum_{i=1}^{t+1}\sum_{j=1}^{i}\Big[\frac{x_j}{P_t} - \frac{(1-x_j)}{(1-P_t)}\Big] \\ &\quad - \sum_{i=1}^{t+1}\Big[\frac{x_i}{P_t} - \frac{(1-x_i)}{(1-P_t)}\Big] \\ 0 &= \sum_{i=1}^{t+1}\sum_{j=1}^{i} x_j - \sum_{i=1}^{t+1}\sum_{j=1}^{i} P_t - \sum_{i=1}^{t+1} x_i + \sum_{i=1}^{t+1} P_t \\ 0 &= \sum_{i=1}^{t}\sum_{j=1}^{i} x_j - \frac{P_t(t+1)(t+2)}{2} + P_t(t+1) \end{aligned}$$

The posterior probability P_t as a function of t is given by:

$$P_t = \frac{2}{t(t+1)}\sum_{i=1}^{t}\sum_{j=1}^{i} x_j$$

Learning the whole history even for environment dynamics

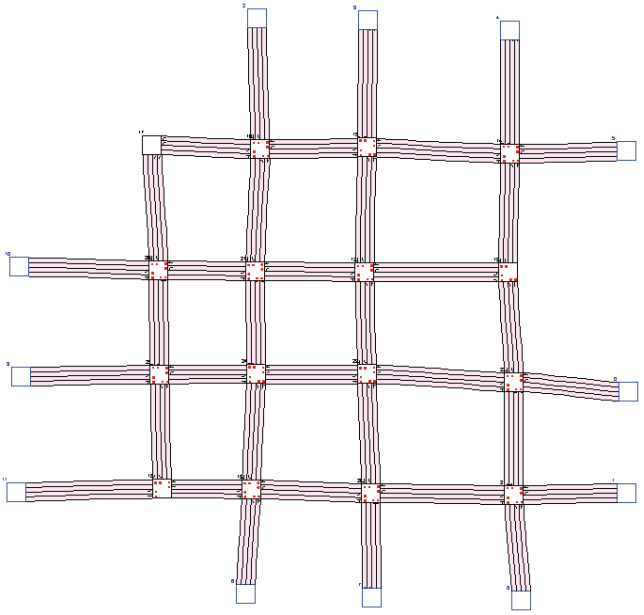

Fig. 1. Traffic network with 12 edge nodes, 15 traffic light nodes and 1 node without a traffic light

which have been previously experienced gives our controller the ability to deal with congestion situations that occur for some limited time (which is typical to rush hours and accidents).

V. Performance Evaluation

The original experiments in [9] [10] were done with fixed spawning rates. In our experiment, we use dynamic spawning probability distributions. We use the traffic network in Fig. 1 with 12 edge nodes, 15 traffic light nodes and 1 node without a traffic light and 36 roads each consists of 4 lanes. We set the γ (discount factor) to 0.9 and set a random traffic light decision chance to 0.01 for exploration purposes. All edge nodes are set to the same vehicle spawning frequencies and each edge node has an equal chance of being the destination of a new vehicle except its source node. In this experiment, the spawning probability distributions change over time simulating the realistic road situations of varying traffic amount entering and leaving a city due to rush hours and accidents. We assume that all vehicles have equal length and number of passengers. The results of the experiment is the average of 10 independent runs each has seed equals to its starting PC clock time. Every run consists of 50000 time steps.

At time step 1, the spawning distribution is set to $Weibull(k = 20, \lambda = 20)$ (i.e. at maximum a vehicle is spawned every 17 time steps and at minimum a vehicle is spawned every 21 time steps, this vehicle spawning rate can lead to a free traffic situation). At time steps 5000, 10000 and 15000 the spawning rate is fixed to 0.1, 0.2 and 0.3 respectively.

At time step 20000, the spawning distribution is set to $\mathcal{U}(a = 2, b = 4)$ (i.e. at maximum a vehicle is spawned every 2 time steps and at minimum a vehicle is spawned every 4 time steps, this vehicle spawning rate can lead to a congested traffic situation). At time step 25000, the spawning rate is fixed to 0.37 that represents the rush hour peak. The same initial behavior is repeated again; at time step 30000, the spawning distribution is set to $Weibull(k = 20, \lambda = 20)$. At time steps 35000, 40000 and 45000 the spawning rate is fixed to 0.1, 0.2 and 0.3 respectively.

We set the driver politeness factor in the IDM driver model settings to -1 (i.e. aggressive driver) that increases the probability of accidents and road non-stationarity.

Three performance measures are considered good indicators for measuring the congestion inside a traffic network [19]:

Average Trip Waiting Time: represents the amount of time spent by the vehicles waiting not driving in the whole trip.

Average Junction Waiting Time: represents the average time a vehicle has to wait at each traffic light. This measure is a good indicator of the congestion level in the traffic network.

Total Waiting Queue Length: The new spawned vehicles entering the traffic network at any edge node are put in a waiting queue if the lanes leading from this edge node is fully congested. This measure is calculated by summing the waiting queue lengths at all edge nodes. When there are unoccupied positions for the queued vehicles, i.e. congestion is being resolved, the total waiting queue length will decrease.

Figures 2, 3, 4 compare the three performance measures of our Bayesian-based controller versus TC-1 with the pre-set dynamic spawning distributions. In low traffic congestion, the two algorithms almost have the same performance. This is due to the bias probabilities to the next states of each vehicle position have equal chance. Under congested traffic periods, our controller significantly outperforms the TC-1 controller (paired t-test, $P_{chance} < 0.0001$, by conventional criteria this difference is considered to be extremely statistically significant). This is due to some next states (vehicles stay in the same positions due to congestion) have higher weights. This situation is quickly detected by our controller.

We noticed that when an extreme traffic congestion lasts for a very long time period, TC-1 controller and our Bayesian-based controller almost have the same performance. This is due to our controller learn the whole history. We can avoid that by learning partially from the history. For now, this is rare to happen where congestion in real world lasts for some limited time that our controller perfectly deals with (which is typical to rush hours and accidents) in which the performance measures will not overshot during this congested traffic period.

VI. Conclusion And Future Work

In this paper, we present an adaptive traffic control system based on Bayesian probability interpretation. Non-stationarity in traffic road conditions are modeled by varying spawning probability distributions. We implement the acceleration and lane changing models in GLD based on the IDM driver model that is used to simulate the longitudinal dynamics and the behavior of the drivers. We compare three important performance measures: Average Trip Waiting Time, Average Junction Waiting Time, and Total Waiting Queue Length of our controller versus TC-1 with pre-set dynamic spawning distributions. Under sudden congested traffic conditions, our controller significantly outperforms the TC-1 controller.

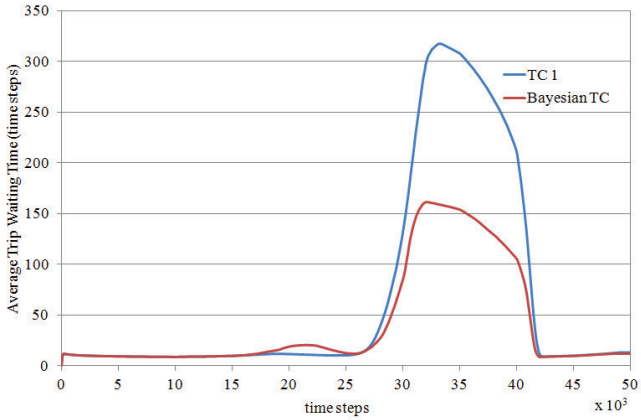

Fig. 2. Average Trip Waiting Time of our Bayesian-based controller versus TC-1 with dynamic spawning distributions

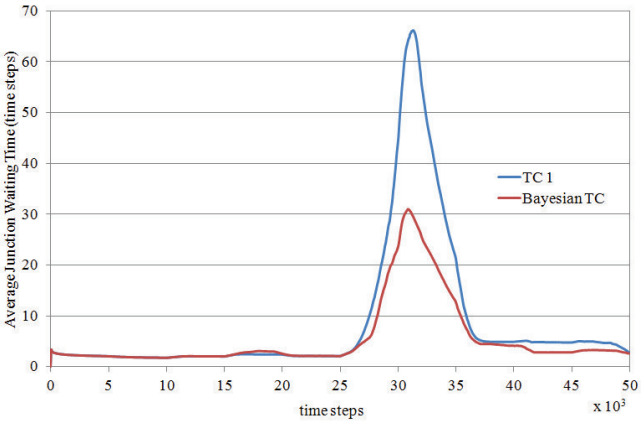

Fig. 3. Average Junction Waiting Time of our Bayesian-based controller versus TC-1 with dynamic spawning distributions

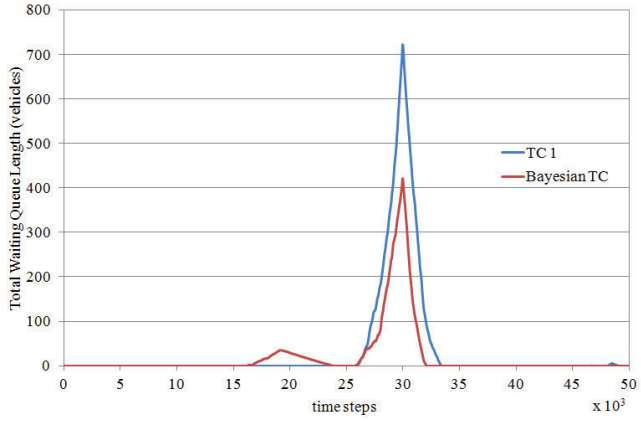

Fig. 4. Total Waiting Queue Length of our Bayesian-based controller versus TC-1 with dynamic spawning distributions

As a future work, we want to test our controller behavior when generating random perturbation that will lead to sudden braking or lane-changing manoeuvres. We want to apply our controller on many traffic patterns with varying spawning distributions and find the optimal traffic patterns in which our controller optimally behaves. Our long-term goal is to implement and test the proposed controller on real traffic systems. This can be applied by physical deployment on a real zone in Alexandria city. Before reaching that goal, more results are needed for checking the controller accuracy and making real time validation. We need to integrate the traffic control system with sensors (through loop detectors in roads, cameras, and/or communication with vehicles using GPS/Wi-Fi sensors).

ACKNOWLEDGMENT

Special thanks are due to Dr Hisham El-Shishiny, manager of IBM Center for Advanced Studies in Cairo, for fruitful discussions.

REFERENCES

[1] Egypt vehicles accidents statistics. Last accessed at 23th Jan 2012. [Online]. Available: http://www.capmas.gov.eg/pdf/indicators/tables/22.pdf
[2] K. Abbas, "Traffic safety assessment and development of predictive models for accidents on rural roads in egypt," *Accident Analysis & Prevention*, vol. 36, no. 2, pp. 149–163, 2004.
[3] Y. Shoham and K. Leyton-Brown, *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge Univ Pr, 2009.
[4] L. De Oliveira and E. Camponogara, "Multi-agent model predictive control of signaling split in urban traffic networks," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 1, pp. 120–139, 2010.
[5] S. Sen and K. Head, "Controlled optimization of phases at an intersection," *Transportation science*, vol. 31, no. 1, pp. 5–17, 1997.
[6] T. Thorpe and C. Anderson, "Traffic light control using sarsa with three state representations," in *IBM Corporation*. Citeseer, 1996.
[7] M. Shoufeng, L. Ying, and L. Bao, "Agent-based learning control method for urban traffic signal of single intersection," *Journal of Systems Engineering*, vol. 17, no. 6, pp. 526–530, 2002.
[8] B. Abdulhai, R. Pringle, and G. Karakoulas, "Reinforcement learning for true adaptive traffic signal control," *Journal of Transportation Engineering*, vol. 129, p. 278, 2003.
[9] M. Wiering, J. Veenen, J. Vreeken, and A. Koopman, "Intelligent traffic light control," *UU-CS*, no. 2004-029, 2004.
[10] M. Wiering, J. Vreeken, J. Van Veenen, and A. Koopman, "Simulation and optimization of traffic in a city," in *Intelligent Vehicles Symposium*. IEEE Intelligent Transportation Systems Society, 2004, pp. 453–458.
[11] S. Cools, C. Gershenson, and B. DHooghe, "Self-organizing traffic lights: A realistic simulation," *Advances in Applied Self-Organizing Systems*, pp. 41–50, 2008.
[12] L. Kuyer, S. Whiteson, B. Bakker, and N. Vlassis, "Multiagent reinforcement learning for urban traffic control using coordination graphs," *Machine Learning and Knowledge Discovery in Databases*, pp. 656–671, 2008.
[13] M. Treiber and D. Helbing, "Realistische mikrosimulation von straßenverkehr mit einem einfachen modell," in *16th Symposium Simulationstechnik ASIM*, Rostock. Djamshid Tavangarian and Rolf Grützner, 2002, pp. 514–520.
[14] Intelligent-driver model (idm). Last accessed at 23th Jan 2012. [Online]. Available: http://www.vwi.tu-dresden.de/~treiber/MicroApplet/IDM.html
[15] D. Helbing, "Traffic and related self-driven many-particle systems," *Reviews of modern physics*, vol. 73, no. 4, p. 1067, 2001.
[16] Minimizing overall braking decelerations induced by lane changes. Last accessed at 23th Jan 2012. [Online]. Available: http://www.vwi.tu-dresden.de/~treiber/MicroApplet/MOBIL.html
[17] J. Mason Jr, K. Fitzpatrick *et al.*, "Identification of inappropriate driving behaviors," *Journal of transportation engineering*, vol. 118, p. 281, 1992.
[18] L. Kuyer, B. Bakker, and S. Whiteson, "Multiagent reinforcement learning and coordination for urban traffic control using coordination graphs and max-plus," Master's thesis, Intelligent Autonomous Systems group Faculteit der Natuurwetenschappen, Wiskunde en Informatica, Universiteit van Amsterdam, 2008.
[19] J. Iša, J. Kooij, R. Koppejan, and L. Kuijer, "Reinforcement learning of traffic light controllers adapting to accidents," 2006, reinforcement Learning of Traffic Light Controllers Adapting to Accidents.