Applied Formal Methods
Final Project Report: Modelling and Verification of Traffic Signal System

Soumyabrata Talukder

Date: 11th Dec'17

1. Introduction

Today in many of the dense populated cities, traffic jams have become a perpetual issue. With increase in the number of on-road vehicles day by day, the improvement of road area is disproportional because of the obvious fact that road widening is not always feasible due to limitation of free land space and involvement of huge capital investment. To harness increasing traffic jam, hence, we are left out with the only option of efficient flow-management strategy of the on-road vehicles, a significant mean to realize which, indeed, is to have an improved intersection signal strategy. However, despite technological advancement in every dimension, most of the traffic light control systems are still using predetermined time setting, as of today.

Many works have been done in the past to overcome the problem caused by the usage of traffic signals with predetermined timer. Teo and Chin [1] has extended the concept of Genetic Algorithm for the optimization of the Traffic Flow Longer green time will pass through more vehicles, but it will increase the cycle time at the same time which causes more vehicles to accumulate at the intersection during the waiting time. Using Genetic Algorithm, the optimal value of the green time of the signal is determined. They have predicted the duration of current green time based on its wait time that is duration for which the other lanes were green. Since this approach is based on prediction and not on any current real-time data, the whole system cannot be considered as reliable as far as average waiting time for each vehicle is concerned. Zhou et al [2] has proposed an algorithm which takes care of almost all the considerable parameters such as traffic volume, waiting time, vehicle density etc., to determine green light sequence and the optimal green light duration. One drawback in this approach is that they have considered 12 possible configurations of green light, which is indeed not required and is a system overhead. Cheng [3] has also proposed a similar approach by applying an improved adaptive PI algorithm to calculate the weight of each state, which is the key to determine the next state of a traffic light. If the current state is identical to the expected one, green light period is extended for the current state, otherwise the expected state is switched on, and its green light period is launched. However as in the previous paper [2] they have also considered 12 possible configurations of green light and suffers from similar drawback.

Consider the traffic scenario shown in the Figure 1, where the North-South oriented (NS) road is assumed to be an important road connecting a residential and a commercial part of a city. This road is also assumed to be a part of a state high-way and hence remains busy throughout the day. Along this road there are a series of signaled intersections (with predefined timer) which connects this main road to different residential and/or commercial sub-segments of the city via East-West oriented (EW) lanes. In morning, traffic flow in EW lanes and the flow from south to north remain very high since people drive down to their office from their home. Similarly, in evening, there exists traffic congestion in the EW lanes and from north to south due to the home-returning employees with vehicle. We also assume that traffic flow in EW lanes is insignificantly low during the rest of the day, though the through traffic in NS road still needs to unnecessarily wait at the intersections since the signal time is predefined in the intersections. The objective of this project is to explore ways to improve the above-cited traffic scenario. Different traffic signal models are coded for this purpose and the improved traffic efficiency in each case is established formally through model checking. The model is validated by verifying various safety and liveness properties of traffic signal. For model checking, NuXmv, a symbolic model checking tool has been used. One of the key reason of selecting symbolic model checker, indeed, is that it allows symbolic modelling and thus makes model abstraction easier, while it also enables the user to get rid of defining an explicit automaton of the system state space, which is cumbersome in most of the practical systems. Another reason of selecting NuXmv is to leverage its strength in performing bounded and IC3 based model checking. A subtle motivation of this project is to explore various methods of model checking through a practical case study and draw a comparative conclusion.

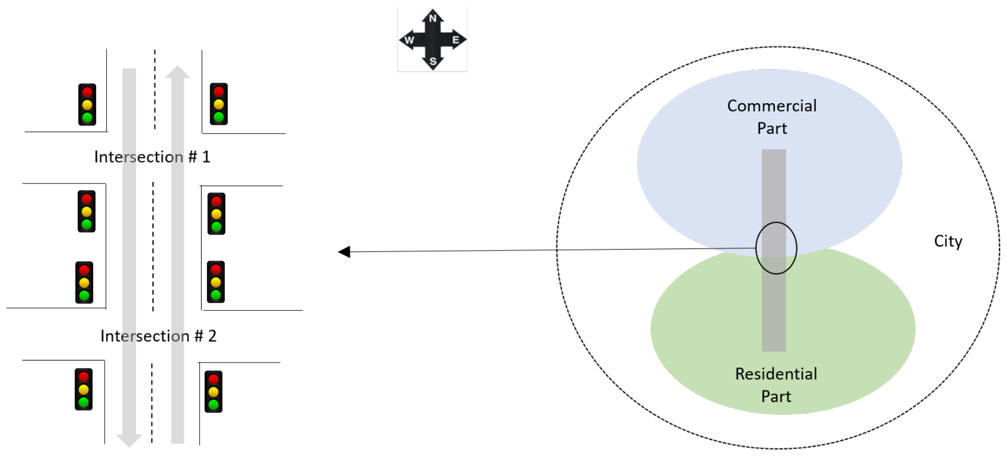


Figure 1: A Typical Traffic Scenario in City

2. Modelling

2.1. A Basic Single Intersection Model

For modelling of the traffic signal system, the first step is to create a model of a single intersection traffic signal. Three types of signal e.g. green, yellow and red, are considered for both NS and EW lane. Hence six boolean variables (viz.NS_green, NS_yellow, NS_red, EW_green, EW_yellow, EW_red) are considered for denoting the said signals. It is assumed that the green signal duration and yellow signal duration are different (which is common in practice) and duration of red signal is the sum of duration of green and yellow signal. Therefore, two integer variables (viz. Timer_G and Timer_Y) are considered here to realize the timers for both green and yellow signals. For starting of timer and detecting that timer countdown ends, two more boolean variables are considered for each of the said timers (viz.Traff_CNTDWN_Start_G, Traff_CNTDWN_Start_Y, Traff_CNTDWN_End_G, Traff_CNTDWN_End_Y).

Figure 2 represents a control flow chart of a single intersection traffic signal system and Figure 3. shows a typical configuration of such an intersection. At the initial state, the red signal in NS road and the yellow signal in EW lane are active. Since yellow signal is active, it enables the yellow timer to start countdown. Once the timer ends its countdown, the signal is changed to green and red in the NS and EW road respectively.

This enables the green timer to start countdown, followed by the change of signals to yellow and red for the NS and EW road respectively, once the green timers countdown ends. This state again starts the yellow timer countdown which ends with change of the signals to red and green for NS and EW roads respectively. Subsequently the green timer starts countdown and eventually when countdown ends, changes the signals to red and yellow in the NS and EW roads respectively, which was the initial state.

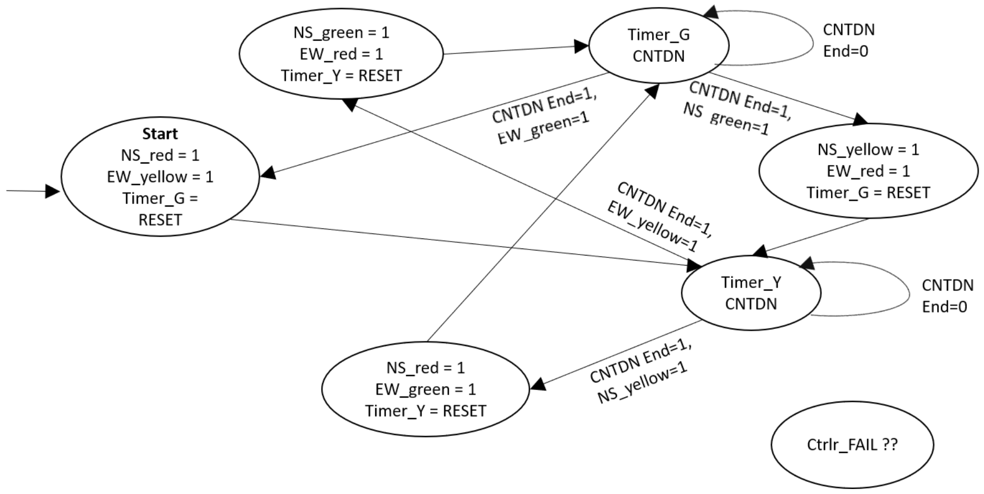


Figure 2: Control Flow Diagram of a Single Intersection Traffic Signal System

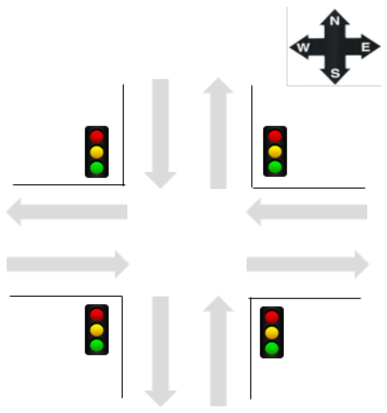


Figure 3: A Typical Single Intersection

While this cycle continues recursively, the controller at the backend keeps on verifying its self-healthiness at every state and changes the operation to a failsafe mode if any abnormality is detected. In failsafe mode, the red signal blinks continuously in both NS and EW lanes and disable all other operations of the traffic signal system. The purpose of the failsafe mode is to alert the traffic and make them yield their right of way safely instead of depending on the faulty traffic controller. A boolean variable Ctrlr_FAIL is used for realizing the healthiness status of the controller in this model. The NuXmv executable script of the model of single intersection traffic signal system is included in the project binder folder as the file

"Single_intersection.smv". To validate this basic model, several obvious safety and liveness properties of the system, represented in Linear Temporal Logic(LTL), are verified. The validation properties, alongwith their english interpretations, are as enumerated below:

•$G(Ctrlr\_FAIL \Rightarrow$ X(G ¬(NS_yellow))) : If controller fails, yellow signal will never be issued in the lanes of NS road

•$G(Ctrlr\_FAIL \Rightarrow$ X(G ¬(EW_green))) : If controller fails, green signal will never be issued in the lanes of EW road

•$G(Ctrlr\_FAIL \Rightarrow$ X(G ¬(EW_yellow))) : If controller fails, yellow signal will never be issued in the lanes of EW road

•$G(Ctrlr\_FAIL \Rightarrow$ X (G ((NS_red ⇒ X ¬NS_red) ∧ (¬NS_red ⇒ X NS_red)))) : If controller fails, red signal blinks in the lanes of NS road continuously

•$G(Ctrlr\_FAIL \Rightarrow$ X (G ((EW_red ⇒ X ¬EW_red) ∧ (¬EW_red ⇒ X EW_red)))) : If controller fails, red signal blinks in the lanes of EW road continuously

•$G(Ctrlr\_FAIL \Rightarrow$ G (Ctrlr_FAIL)) : If controller fails, it remains in failure condition till expert's intervention and an appropriate troubleshooting

•G¬(NS_green ∧ EW_green) : Green signal for both NS and EW road lanes can never be glowing simultaneously

•G¬(NS_green ∧ EW_yellow) : Green signal for both NS road lanes and yellow signal for EW road lanes can never glow simultaneously

•G¬(NS_yellow ∧ EW_green) : Yellow signal for both NS road lanes and green signal for NS road lanes can never glow simultaneously

•G¬(NS_yellow ∧ EW_yellow) : Yellow signal for both NS and EW road lanes can never be glowing simultaneously

•G(¬Ctrlr_FAIL ⇒ G (NS_red U EW_red)) : If controller is healthy, red signal in NS road lanes can never be released until red signal in EW road lanes is turned on

•G(¬Ctrlr_FAIL ⇒ G (EW_red U NS_red)) : If controller is healthy, red signal in EW road lanes can never be released until red signal in NS road lanes is turned on

•G(¬Ctrlr_FAIL) ⇒ (G ¬(((NS_green ∧ X (¬NS_green)) ∧ X (NS_red)))) : If controller is healthy, green signal can never change directly to red signal in NS road lanes

•G(¬Ctrlr_FAIL) ⇒ (G ¬(((EW_green ∧ X (¬EW_green)) ∧ X (EW_red)))) : If controller is healthy, green signal can never change directly to red signal in EW road lanes

•G(¬Ctrlr_FAIL) ⇒ (G ¬(((NS_yellow ∧ X (¬NS_yellow)) ∧ X (NS_green)))) : If controller is healthy, yellow signal can never change directly to green signal in NS road lanes

•G(¬Ctrlr_FAIL) ⇒ (G ¬(((EW_yellow ∧ X (¬EW_yellow)) ∧ X (EW_green)))) : If controller is healthy, yellow signal can never change directly to green signal in EW road lanes

•G(¬Ctrlr_FAIL) ⇒ (G ¬(((NS_red ∧ X (¬NS_red)) ∧ X (NS_yellow)))) : If controller is healthy, red signal can never change directly to yellow signal in NS road lanes

•G(¬Ctrlr_FAIL) ⇒ (G ¬(((EW_red ∧ X (¬EW_red)) ∧ X (EW_yellow)))) : If controller is healthy, red signal can never change directly to yellow signal in EW road lanes

•G(¬Ctrlr_FAIL) ⇒ (G (F EW_green) ⇒ F NS_green) : If controller is healthy, green signal infinitely many times in EW road lanes always implies green signal will eventually be issued to NS road too

•G(¬Ctrlr_FAIL) ⇒ (G (F NS_green) ⇒ F EW_green) : If controller is healthy, green signal infinitely many times in NS road lanes always implies green signal will eventually be issued to EW road too

•((G(¬Ctrlr_FAIL)) ⇒ G (F NS_green)) : If controller is healthy green signal will be issued to NS road lanes infinitely many times

•$((G(\neg \text{Ctrlr\_FAIL})) \Rightarrow \text{G (F NS\_yellow)})$: If controller is healthy yellow signal will be issued to NS road lanes infinitely many times

•$((G(\neg \text{Ctrlr\_FAIL})) \Rightarrow \text{G (F NS\_red)})$: If controller is healthy red signal will be issued to NS road lanes infinitely many times

•$((G(\neg \text{Ctrlr\_FAIL})) \Rightarrow \text{G (F EW\_green)})$: If controller is healthy green signal will be issued to EW road lanes infinitely many times

•$((G(\neg \text{Ctrlr\_FAIL})) \Rightarrow \text{G (F EW\_yellow)})$: If controller is healthy yellow signal will be issued to EW road lanes infinitely many times

•$((G(\neg \text{Ctrlr\_FAIL})) \Rightarrow \text{G (F EW\_red)})$: If controller is healthy red signal will be issued to EW road lanes infinitely many times

By ensuring that the model exhibits the desired result in all of the abovementioned verifications, we can conclude that the modelling of the system is alright.The NuXmv script of sub-model of a prototype timer, which is implemented in the single intersection traffic signal system, is also included in the project binder folder as file timer_test.smv, for reference. This timer model can be validated looking at the counterexample generated by verification of satisfiability of the LTL property G $\neg$(Timer = 2).

2.2. A Fixed Coordinated Timer(FCT) based Four-intersection Model

Next, using this basic model, a four-intersection model is developed, where the signal timers of different intersections are predetermined but they are not arbitrary in time-zone. A diagram of the model is depicted in Figure 4. This model is developed as an improvised traffic signal model, for emulating and verifying the morning time traffic scenario in the NS road depicted in Figure 1 (the reason in explained in subsequent sections). Hence for measuring traffic flow efficiency we will consider traffic flow only from south to north in this case. The timers of different intersections are all coordinated to each other with some predefined time delay.

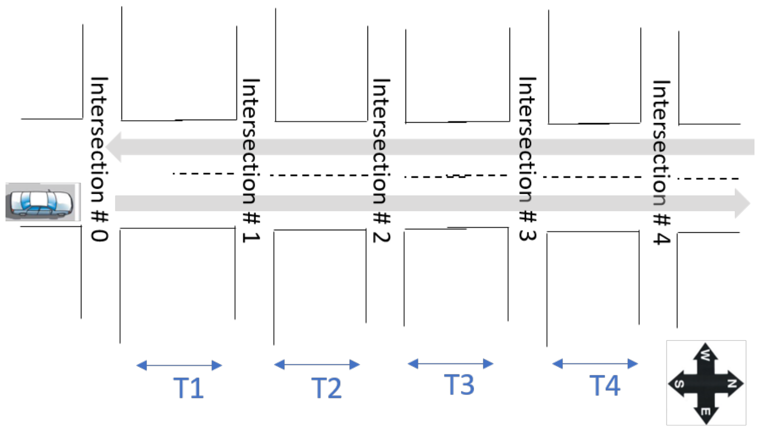


Figure 4: A Four Intersection Traffic Signal Configuration

Such a time dependency between traffic signals of a pair of intersections is pictorially represented in Figure 5. This time delay is a function of the length of the immediate upstream road segment and the maximum permitted vehicle speed in that segment with a designed average speed (as a fraction of the maximum speed). A vehicle module is added into the model where a vehicle starts at a given point in the upstream of

Intersection-1 and runs at a given average speed along the road segment till it meets an intersection and waits for the signal to become green to cross the intersection and proceed so on, until it crosses Intersection-4. The vehicle is enabled to start from its origin when a random boolean variable Vehicle_enable is set. In NuXmv a variable can be easily made random, if not initialized and no transition relation is defined. To count the vehicle transit time in a road segment, a timer is realized by an integer variable Vehicle_transit_time which is initialized to a given value when the vehicle enters a road segment. This value is derived from the given length of that road segment and the average speed of the vehicle (as a fraction of the maximum permitted speed in this model). When the timer ends countdown, it implies that the vehicle has crossed the road segment and arrived at the next intersection.

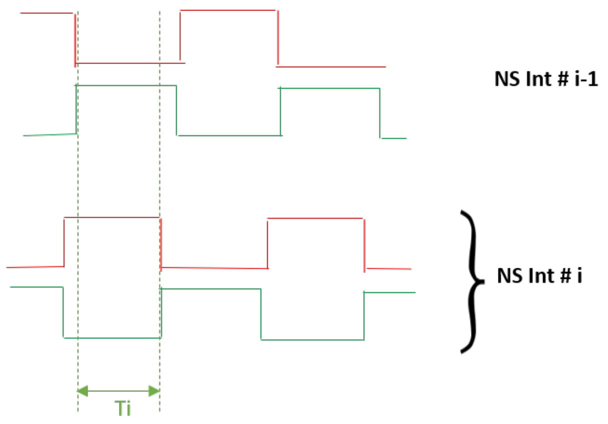


Figure 5: Time Dependency of Signals of a Pair of Intersections (i corresponds to the index of an intersection, i ϵ (1,2,3,4)

An integer variable Vehicle_wait is considered in which the total time the vehicle spends at intersections, waiting for green signal, is counted. Another integer variable Vehicle_crossed_signals is considered which is initialized to zero and increases by one, every time the vehicle crosses an intersection. A flow diagram of the vehicle module, in line with the said modus operandi is presented in Figure 6.

Since the vehicle is made to start from its origin at a random instant, the variable Vehicle_wait stores the maximum possible wait time of the vehicle, at the end of the state space execution. During validation of this model in a method similar to what was done in the previous model, it was observed that the counter example trails are too long and seemingly convoluted as the state space consists of 56 variables, making debugging very difficult and intractable. Therefore, a strong requirement of developing an abstraction of the basic model, for the purpose of replicating for signals at the four intersections, is realized. Eventually, it is noticed that the basic model can be alternatively realized just by three variables e.g. NS_green, Timer_G and Ctrlr_FAIL, without loss of generality. In this alternative model, NS_green is defined as a state combining both green and yellow signal and the red signal is defined by complement of NS_green. Now that the yellow signal no longer exists in the model, NS_green variable also provides complete information about EW road signal, since the signals of EW and NS road essentially complements each other. Moreover, elimination of yellow signal eliminates the requirement of Timer_Y too. It is realized that the failsafe property is not relevant for the given project objective and hence the related variable can also be dropped.

The model is redesigned with only two variables per intersection and the variables that are required for the vehicle module. An additional random integer variable is added (viz. temp_G) here which abstracts the green signal duration (also referred as signal duration hereafter) with a predefined upper limit. By this variable, it is ensured that Vehicle_wait provides the maximum possible cumulative waiting time on execution of the state space, with any possible non-negative signal duration within a given bound.

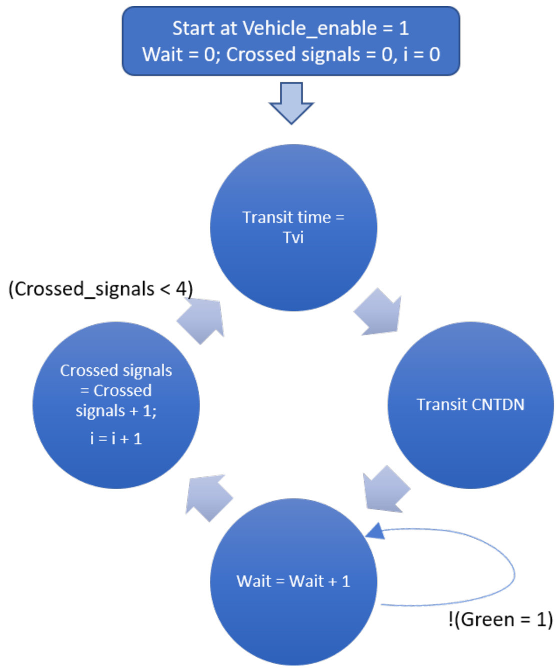

Start at Vehicle_enable = 1
Wait = 0; Crossed signals = 0, i = 0
Transit time = Tvi
Transit CNTDN
Wait = Wait + 1
!(Green = 1)
Crossed signals = Crossed signals + 1;
i = i + 1
(Crossed_signals < 4)

Figure 6: Control Flow Diagram of the Vehicle Module (i corresponds to the index of an intersection, i ϵ (1,2,3,4)

One integer variable is used for each intersection (viz. Start_delay_timer_1, Start_delay_timer_2, Start_delay_timer_3 and Start_delay_timer_4) to realize its time coordination wrt Intersection-1. Its represent the time by which the event of NS_green_i (i corresponds to an intersection) turning red to green, is delayed from the similar event at Intersection-1. The NuXmv script of this four-intersection coordinated timer model is included in the project binder folder as "Fixed_coordinated_timer.smv". Validation of this redesigned model is much simpler since the number of variables are reduced from 56 to 17 now. The model is validated by verifying the following safety and liveness properties, defined in LTL, in the mode:

•$Vehicle\_enable \Rightarrow$ (G (X (Vehicle_wait) $\geq$ Vehicle_wait)) : If vehicle starts from its origin, the cumulative waiting time at the intersections never decreases with time

•$Vehicle\_enable \Rightarrow$ (F (G (Vehicle_crossed_signals=4))) : If vehicle starts from its origin, eventually the vehicle crosses four intersections in totality

•G(F NS_green_i$*$) : Infinitely many times, green signal of NS road lanes of $Intersection - i*$ is turned on

•G(F $\neg$(NS_green_i$*$)) : Infinitely many times, green signal of NS road lanes of $Intersection - i*$ is turned off

•F(G (Start_timer_i$* = 0$)) : Eventually the delay timer of $Intersection - i*$ counts down to zero and never changes after that

•G(F (Timer_G_i$* = 0$)) : The green timer of $Intersection - i*$ counts down to zero infinitely many times

•$G(F\ \neg(\text{Timer\_G\_}i* = 0))$: The green timer of $Intersection-i*$ is re-initialzed to a preset signal duration infinitely many times
* i ϵ (1,2,3,4)

Though the model is designed to verify traffic flow from south to north (i.e. the morning time traffic scenario in the NS road in Figure 1, it doesnt loss generality and can still be applied for the north to south case (i.e. the evening time traffic scenario in the NS road in Figure 1 by making the vehicle module to run in the reverse direction i.e. from north to south and coordinating the signal timers at different intersections wrt Intersection-4, in lieu of Intersection-1 as it is done in this model.

2.3. Adaptive Timer based Four-Intersection Model

In this step, a four-intersection traffic signal model is developed in which the signal times of the different intersections are independent but adaptive. A pictorial representation of the model is given in Figure 7 where the locations of the sensors are depicted by the red dots. Since At every intersection, two sensors are placed along each of the lanes of EW road, one near the intersection and the other one is at a definite distance from the intersection in the upstream side.

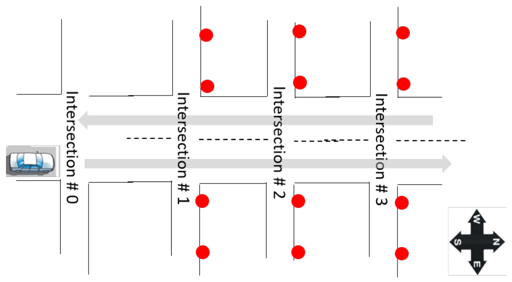


Figure 7: A Four Intersection Traffic Signal Configuration with Sensors

These sensors count the nos. of vehicles crossing the sensor location. The key principle of operation is that if there is no queue in both the lanes of the EW road, the state of NS signal and EW signal remains green and red respectively. While the EW road signal is red in an intersection and there is a traffic queue accumulated in any of the EW lanes of the intersection, the controller shall ensure that the vehicles queued up will be eventually given a green signal to cross the intersection, which takes place as soon as the Timer_G of the intersection counts down to zero. Before the EW road signal turns green, it updates the Timer_R based on the maximum queue length over the lanes of EW road. This queue length in a lane is determined by the controller from the difference of the data transmitted from the sensors in that lane. The signal time can be predicted from the road width and the maximum queue length. For this model, we have considered a random integer variable for each intersection (viz. EW_queue_1, EW_queue_2, EW_queue_3, EW_queue_4) that represents the maximum queue length over the lanes of EW road of that intersection. In this model, the Timer_R value is adapted/reinitialized by a constant scaled value of the current maximum queue length before the signal of EW road lanes turn green from red. A flow diagram for the recursive process of updating red timer is depicted in Fig.8. The Start_delay_timer_i and temp_G variables introduced in the previous model are dropped here since the operation of the signals at different intersections is independent now and the signal duration is no longer fixed. The vehicle module (to verify the maximum cumulative waiting time) and the basic structure of an intersection remain unaltered in this model from what was implemented in the previous one. The NuXmv script of this four-intersection coordinated timer model is included in the project binder folder as file "Adaptive_timer.smv". The validation properties of this model is similar to those of the previous model, excluding the property related to Start_timer_delay_i. In addition the following property is included in the list of validation properties of adaptive timer model.

•$G(\neg(\text{EW\_queue\_i}* = 0)) \Rightarrow$ G (F $\neg$(Timer_R_i$*$ = 0)) : $If queue length in ith intersection is always non-zero, red signal timer of that intersection will be reset to a non-zero value infinitely many times$

3. Model Verification

In the Fixed Coordinated Timer Model, the LTL property ($\neg$(Vehicle_wait ¿ Max_wait)) has been verified with an arbitrary guess of constant Max_wait. Max_wait is a defined constant in this model. The other model parameters defined for the purpose of model verification are as follows:

Duration_green = 20 (The signal duration, considered to be same for all intersections for the sake of simplicity)
Max_wait = 51 (Max wait time for verification, varied multiple times in each iteration)
Distance_0_1 = 1000 (Distance between intersection 0 and 1)
Distance_1_2 = 550 (Distance between intersection 1 and 2)
Distance_2_3 = 750 (Distance between intersection 2 and 3)
Distance_3_4 = 1500 (Distance between intersection 3 and 4)
Max_Speed_Limit_0_1 = 40 (Max permissible speed in road segment 1)
Max_Speed_Limit_1_2 = 30 (Max permissible speed in road segment 2)
Max_Speed_Limit_2_3 = 40 (Max permissible speed in road segment 3)
Max_Speed_Limit_3_4 = 40 (Max permissible speed in road segment 4)
Assumed_Avg_to_Max = 90 (Assumed average speed for design as percentage of max permissible speed)
Vehicle_Actual_to_Max = 71 (Actual vehicle average speed as percentage of max permissible speed)

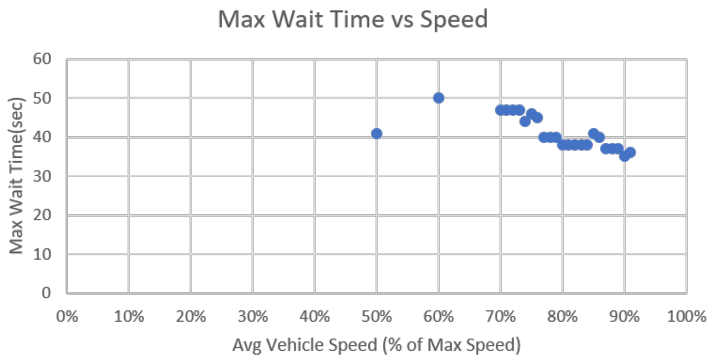


Figure 8: Scatter diagram of Maximum Cumulative Waiting Time wrt Actual Average Vehicle Speed

As stated earlier this property verification implies verification of maximum cumulative waiting time of the vehicle over all the intersections. This verification is being performed once for all possible starting time of the vehicle and all possible non-negative signal duration within a given upper bound. After verification if it turns out that the property doesnt satisfy, the value of Max_wait is increased in steps till the property is satisfied by the model. If the property is found satisfied during verification, the value of Max_wait is reduced in steps and the property till the property is found unsatisfied. If the satisfiability status doesnt change in a successive iteration, the step size is increased and otherwise decreased. The algorithm of finding out the maximum waiting time by varying Max_wait is explained through a flow chart in Fig.10. Finally, we reach a value of Max_wait = k (k can be any integer) for which the said property is satisfied, while the property is not satisfied for Max_wait = k-1. From this, we conclude that the max cumulative waiting time of the vehicle is k. While the signal duration is kept constant, the vehicle speed is now varied from 70% to 90% in steps of 1% and the same verification in the iterative way explained above, is performed. The maximum cumulative waiting time of the vehicle in each step is noted. It is evident that the maximum cumulative waiting time without any time coordination among the signals of the intersections is 80 sec. (i.e. 4 times the signal duration). A scatter diagram of the noted result wrt average speed of vehicle is depicted in Figure 8.

In the next verification, the average speed of the vehicle is kept constant at 71% of the maximum permitted vehicle speed in the respective road segments and the same property, as mentioned for the previous verification, is used to verify the model in an iterative way, as explained in Fig.10., while the signal duration is varied from 10 sec. to 60 sec. in steps of 5 secs. The maximum cumulative waiting time obtained at each step is noted and a plot of the noted data is provided in Figure 9. If the signals of the different intersections are not coordinated, the maximum cumulative time is 4 times the signal duration and hence linearly increases with signal duration. This linearly varying maximum cumulative waiting time without time coordination is combined in the same plot in Figure 9. for the sake of comparison.

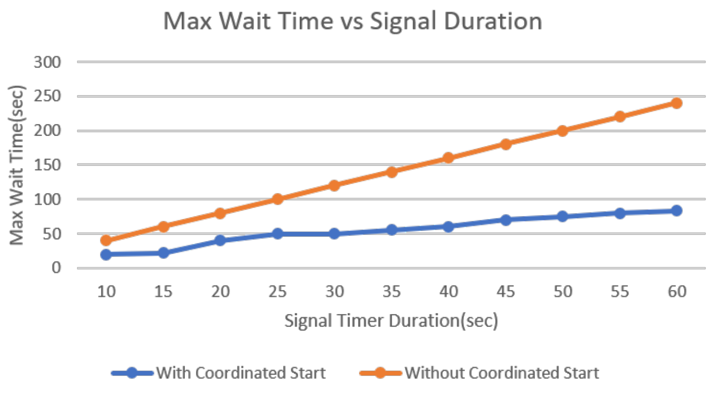


Figure 9: Scatter diagram of Maximum Cumulative Waiting Time wrt Signal Duration

In the Adaptive Timer Model, the verification is done using the similar property, as it was done in the previous model but, in conjunction with an upper bound on the non-negative maximum vehicle queue length at every intersection. The other model parameters defined for the purpose of model verification of this model, are as follows:

Duration_green = 60 (The green/red signal duration, considered to be same for all intersections for the sake of simplicity)

Max_queue_length = 5 (Constant used to represent the upper bound of the maximum queue length over the two lanes in EW road)

Max_wait = 51 (Max wait time for verification, varied multiple times in each iteration)

Distance_0_1 = 100 (Distance between intersection 0 and 1)

Distance_1_2 = 150 (Distance between intersection 1 and 2)
Distance_2_3 = 125 (Distance between intersection 2 and 3)
Distance_3_4 = 90 (Distance between intersection 3 and 4)
Max_Speed_Limit_0_1 = 20 (Max permissible speed in road segment 1)
Max_Speed_Limit_1_2 = 30 (Max permissible speed in road segment 2)
Max_Speed_Limit_2_3 = 25 (Max permissible speed in road segment 3)
Max_Speed_Limit_3_4 = 18 (Max permissible speed in road segment 4)
Vehicle_Actual_to_Max = 71 (Actual vehicle average speed as percentage of max permissible speed)

A common bound is considered for all the intersections for analysis in this case. However, the model can also be used for verification with different maximum queue length bounds for different intersections without any alteration of script.

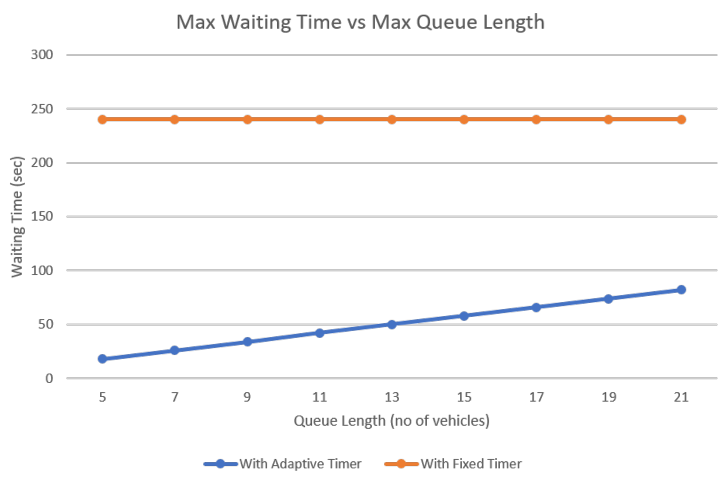


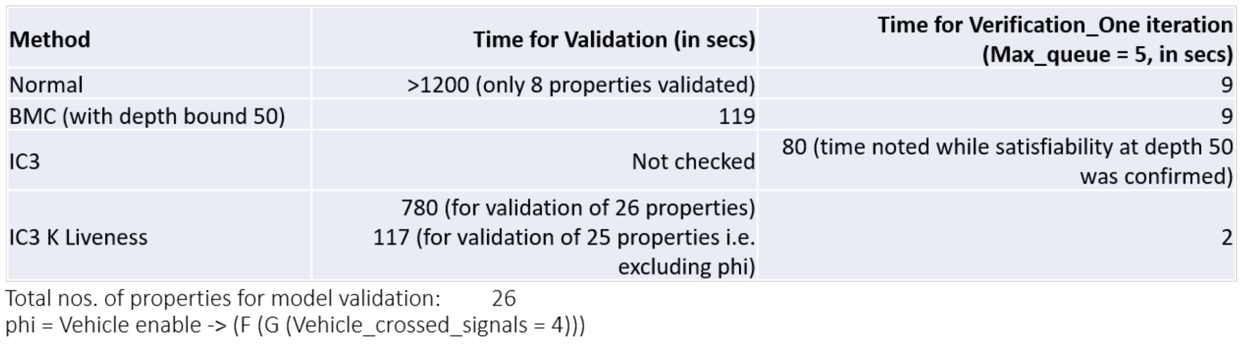
Figure 10: Scatter diagram of Maximum Cumulative Waiting Time wrt Signal Duration

Keeping the other model parameters constant, the upper bound on maximum queue length is increased from 5 to 21 in steps of 2 and the maximum cumulative waiting time is obtained following the algorithm mentioned in Fig.11. The maximum cumulative waiting time for each step is noted and a plot of the maximum cumulative waiting time wrt maximum vehicle queue length is provided in Figure 10. If the signal doesnt have a sensor based adaptive timer system (i.e. a conventional predefined timer type), the maximum cumulative waiting time shall be 4 times the signal duration and will not change with the vehicle queue length.

4. Analysis of Results and Performance

From the plot in Figure 8, it is understood that if the vehicle maintains an average speed between 70% and 90% of the maximum permitted speed of the respective road segments, the maximum cumulative waiting time at intersections can be minimized, by judiciously coordinating the operation of the traffic signal of multiple intersections, to almost half of the maximum cumulative waiting time in case of a conventional fixed timer traffic signal system. If the average vehicle speed is below 70% or more than 90% of the maximum permitted speed, the maximum cumulative waiting time in a coordinated timer traffic signal system will increase in a non-monotonic way with an upper bound of 4 times the signal duration, since the average speed of vehicle, in this case, largely deviates from the designed average speed (i.e. 80% of maximum permitted speed) based on which the signal timers of the intersections are coordinated. The plot in Figure 9 depicts that if the average vehicle speed is maintained in close vicinity of the designed average speed, the result we have obtained is true for any value of signal duration in this test range, which certainly makes the previous result stronger. The plot in Figure 10 depicts a trivial information regarding optimization of the maximum cumulative waiting time by use of adaptive timer based traffic signal system.

In the adaptive timer model, since it was observed that the normal model checking method of NuXmv is taking longer time (more than an hour) in this model as compared to the time it took in fixed coordinated model, few alternate methods of model checking are explored using the same tool, with an intent of a comparative study of verification time taken by the different methods. However, to reduce the total run time of these methods without loss of generality, the signal duration has been reduced from 60 to 10. First, a bounded model checking was carried out with a depth bound of 50 for the validation properties as well as the verification property. This depth bound was guessed based on the summation of all Vehicle_transit_time_i_j (i ϵ (0,1,2,3),j ϵ (1,2,3,4)) for each road segment, all maximum possible EW_queue_i of each intersection and the signal duration. IC3 based model checking was performed subsequently for the verification property, followed by K Liveness based model checking for all the validation and verification properties. A table comparing consumed time by various model checking methods is presented in the table below. The command files that are used for different methods of model checking are included in the project binder as files named cmd_file_bmc, cmd_file_ic3 and cmd_file_ic3_klive.



Method	Time for Validation (in secs)	Time for Verification_One iteration (Max_queue = 5, in secs)
Normal	>1200 (only 8 properties validated)	9
BMC (with depth bound 50)	119	9
IC3	Not checked	80 (time noted while satisfiability at depth 50 was confirmed)
IC3 K Liveness	780 (for validation of 26 properties) 117 (for validation of 25 properties i.e. excluding phi)	2

Total nos. of properties for model validation: 26
phi = Vehicle enable -> (F (G (Vehicle_crossed_signals = 4)))

5. Conclusion

Albeit the result of verification of the Fixed Coordinated Timer Model yields a fruitful result only within a narrow range of vehicle speeds, the result is still important because most of the vehicle runs at an average speed within similar narrow variance around the designed average speed. To maximize the benefit of implementation of fixed coordinated timer model in real life, conducting a practical survey is recommended to arrive at a judicious value of the designed average speed. By summarizing the multiple results obtained in Figure 8, Figure 9 Figure 10, an optimum mode of operation of traffic signal system, for the traffic scenario considered in this project, can be concluded. For the given traffic scenario, in morning and evening, when the EW road lanes are equally busy as compared to the NW road lanes, operating the traffic signal with adaptive timer will not add any value, since the intersections will always seek its Timer_R to be adapted/reinitialized by the maximum possible signal time due to quick accumulation of traffic queue in the EW lanes. Instead, if the traffic signal is operated in a coordinated timer mode, with time delays of operation of traffic signals at different intersections coordinated from one end of the NS road, the maximum cumulative waiting time can be reduced by far (by more than 50%) for the vehicles with average speed of 70% - 90% of the designed average speed. When the office-home bound traffic rush is not present, the traffic signals can be operated in adaptive timer mode which can ensure the maximum possible flow efficiency in the NS road. To this end, it is established that by an appropriate scheduling of the mode of operation of traffic signal system, keeping in view the characteristics of daily traffic flow pattern, a traffic flow efficiency of a road of priority can be improved manifold.

By analysis of the verification time taken by different model checking methods supported by NuXmv, it is observed that the verification time can be significantly minimized by bounded model checking with an appropriate depth bound for search, over the time a normal depth first verification method in NuXmv takes. This is evident since bounded model checking algorithm searches the state space along each execution path up to the given bound of depth, in a breadth first way. However, bounded model checking is incomplete since it confirms satisfiability of a property up to a given search depth only, which narrows down its domain of application. Although the time consumed by IC3 is longer than that in bounded model checking in this specific case of a single invariant property verification property of the model, IC3 can outperform the other verification methods in terms of time efficiency, while verifying a large number of invariant properties of a complex finite model. For model checking of LTL properties, IC3 with K Liveness turns out to be the most optimal trade off between time efficiency and reliability of verification, which is evident from the time duration it took for verification of both validation and verification properties. This is possible since K Liveness with IC3 tackles the problem by proving that an accepting condition can be visited at most K times and leverage its tight integration with IC3. Another advantage of using this method is that the algorithm doesnt require any depth bound to be set by its user. This is extremely advantageous particularly in case of very complex models, where guessing the search depth manually is cumbersome and time-taking.

To this end, a considerable scope of future work extending this project in different ways, has been envisaged. One can extend the adaptive timer model to include a pedestrian crossing request module at the intersections and verify how the performance of the model is impacted due to this inclusion. The pedestrian crossing request feature, however, is not much relevant for the fixed coordinated model since the pedestrians can cross a road during the time when signal of that road is red. There is also an option of extending the model to verify properties of traffic signal system involving more complex structure of intersections (e.g. intersection of 6 ways or higher). Another dimension of future work (an overoptimistic one) could be to estimate the expected value of waiting time given a distribution of a vehicles starting instance, rather than estimating the maximum waiting time, which is verified in this project. Model to minimize the expected value of waiting time and verification of the same will definitely have more practical relevance and hence can be useful to establish the advantage of the proposed model in a stronger and more holistic way.

6. Bibliography

1. K. T. K. Teo, W. Y. Kow, and Y. K. Chin, "Optimization of traffic flow within an urban traffic light intersection with genetic algorithm" in Computational Intelligence, Modelling and Simulation (CIMSiM), 2010 Second International Conference, September 2010, pp. 172-177.

2. X. Z. Binbin Zhou, Jiannong Cao and H.Wu, "Adaptive traffic light control in wireless sensor network-based intelligent transportation system" in Vehicular Technology Conference Fall (VTC 2010-Fall),IEEE 72nd, Septem- ber 2010, pp. 1-5.

3.Y. W. Cheng Hu, "A novel intelligent traffic light control scheme" in Grid and Cooperative Computing (GCC), 2010 9th International Conference on Digital Object Identifier, November 2010, pp. 372-376.

4.Samuel Coogan, Murat Arcak, and Calin Belta, "Finite State Abstraction and Formal Methods for Traffic Flow Networks" in 2016 American Control Conference (ACC, Boston Marriott Copley Place, July 2016, pp. 864-879.

5.Armin Biere, Alessandro Cimatti, Edmund M. Clarke, Ofer Strichman, Yunshan Zhu, "Bounded Model Checking" in Vol 58 of Advances in Computers, 2003.

6.Kristine Y. Rozier, "Linear Temporal Logic Symbolic Model Checking", Computer Science Review, 2010.

7.Aaron R. Bradley, Fabio Somenzi, Zyad Hassan, Yan Zhang, "An Incremental Approach to Model Checking Progress Properties", Proceeding FMCAD '11 Proceedings of the International Conference on Formal Methods in Computer-Aided Design, 2011, pp. 144-153.

8.Zyad Hassan, Aaron R. Bradley, and Fabio Somenzi, "Incremental, Inductive CTL Model Checking",Proceeding CAV'12 Proceedings of the 24th international conference on Computer Aided Verification, 2012, pp. 532-547.

9. Marco Bozzano, Roberto Cavada, Alessandro Cimatti, Michele Dorigatti, Alberto Griggio, Alessandro Mariotti, Andrea Micheli, Sergio Mover, Marco Roveri, Stefano Tonetta, "nuXmv 1.1.1 User Manual"