Agent-Oriented Urban Traffic Control Simulation

Mingshu Li[1,2] Ken Wee Chong[2] Stanley Chan[2] John Hallam[2]

[1]Institute of Software, Chinese Academy of Sciences
P.O.Box 8718, Beijing 100080, P.R.China

[2]Department of Artificial Intelligence, University of Edinburgh
5 Forrest Hill, Edinburgh EH1 2QL, U.K

Abstract

Urban traffic problems are very complex and highly interactive. The Multi-agent systems (MAS) approach may however provide a new hopeful direction. Currently, a simulation system is needed to understand and explore the difficulties in a MAS-based traffic control. This paper discusses some of the issues in an agent-oriented urban traffic control simulation, including a multi-agent paradigm for modelling the urban traffic problem and relationships among the traffic agents. A multi-agent urban traffic control simulation system and its testbed are also described.

Keywords

Urban traffic problems, traffic control simulation, agent, multi-agent systems.

Introduction

As the number of road users increases in every major city of the world, people have to face traffic problems everyday. Generally speaking, a traffic problem consists of all the activities and installations concerned with traffic affairs. This paper will only introduce one of the many urban traffic problems [1]: how to let vehicles drive safely and efficiently according to some reasonable routes from their starting points to their destinations. The task of driving in traffic alone is extremely complex and highly interactive, and most of the traditional methods cannot fully solve it. With the development of computation and communication technologies, distributed artificial intelligence (DAI), especially the MAS approach, provides a new hopeful direction.

Due to the problem complexity, it is not possible to give a complete intelligent solution for the real traffic world at once. Hence a simulation system is needed to understand and explore the difficulties in a MAS-based traffic control.

A Multi-Agent Paradigm for Modelling the Urban Traffic Problem

The notion of agents

An agent is the central computational entity [2], which serves as an explicit model for all entities participating in cooperation. It can be decomposed into the following three components: namely *body*, *head* and *communicator* which implement the functionality, the observable behaviour and message-transmission respectively.

A multi-agent paradigm can be taken as a subfield of DAI technologies to solve problems collaboratively by coordinating the knowledge, goals and plans of semi-autonomous intelligent agents. It offers certain advantages of [3]: faster response, increased flexibility, robustness, resource sharing, graceful degradation, and better adaptability of integrating pre-existing and stand-alone systems.

Using agents to model the urban traffic control simulation problem

If each road user could be embedded into a network of telecommunication links that spread all over the urban region and connected every relevant passive or active entity (agent), then significant improvements are conceivable. Such a traffic world can be viewed as a network consisting of nodes, arcs and moving entities. The streets or roads are the arcs, the junctions and the traffic lights are the nodes which link the roads together, while the moving entities are the vehicles. Thus, each road user can plan a route, select a means of transport and calculate the estimated time needed based on the information given by the network. If necessary, the driver can also replan and come up with an alternative, more viable plan.

A multi-agent model for the urban traffic problem may be viewed as three layers [1]: the highest layer is the global urban traffic agent, executing the global planning; the middle layer is composed of some traffic flow control agents, executing group planning; in the lowest layer, there are a lot of individual traffic agents, executing individual planning. The most important individual traffic agents in this model are vehicles, streets, junctions, and traffic lights. In this paper, only the simulation for individual traffic agents will be discussed [4] [5].

A vehicle (mainly cars in this paper) agent consists of four components: the *communicator* consists of the communication channels, the agent's internal commitment and event queues; the *driver* defines the type of intelligent driving controlling the vehicle; the *controller* is the knowledge-based decision-making component; and the *vehicle* consists of all the attributes and functionalities of a vehicle that enable motion in the simulated world. The communicator can be regarded as the agent's communication unit, the driver and controller as the head and the vehicle as the body. The other three traffic agents have a similar design to the vehicle agent.

Currently, each singleton traffic light is used to control one traffic direction and the set of traffic lights at a junction can be grouped and controlled by a *traffic light manager* consistently. The traffic light manager takes the role of a traffic light agent and is attached to the junction agent.

The most important components in the street and junction simulation are *lanes* and *areas*(Fig 1). Conceptually, a street or a junction can be viewed as being made up of a number of adjacent lanes with each lane consisting of a collection of area components stitched together. Each lane of a street contains a queue of vehicles. The street has all the information of all the vehicles on each of its lanes: their speed, acceleration, distance travelled, the areas currently occupied and predicted to occupy. To avoid the problem of different dimensions in length and width, all the vehicles in a street in the simulated world are assumed to occupy a standard area component of the same length called the *unit car length* (ucl). This value will include the space to separate two vehicles in the same lane. As with the street agent, a junction consists of a number of adjacent lanes. However, the lane of a junction is basically the sequence of the junction area components that a vehicle will occupy while crossing the junction. Vehicles crossing a junction in different directions will take different lanes and pass different areas of the junctions. As an example in Fig 1, a vehicle travelling from *George IV Bridge* to *Bank Street* will take *lane a* of *George IV Bridge*, crossing area sequence $\{5, 1\}$ of the junction, then enter into *lane a* of *Bank Street*; whereas, a vehicle turning right from *George IV Bridge* to *Lawnmaket* will occupy junction areas $\{6, 2, 3, 4\}$. Therefore, the area sequence $\{5, 1\}$ is the junction lane for going straight ahead from *George IV Bridge* to *Bank Street*; and the sequence $\{6, 2, 3, 4\}$ is the junction lane for turning right from *George IV Bridge* to *Lawnmaket*.

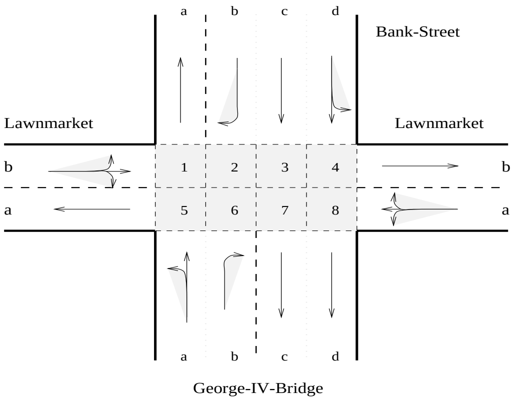

Figure 1: Lanes and areas in street & junction agents

Relationships among the Traffic Agents

The four traffic agents mentioned above can be divided into two major types: mobile agents (vehicles) and immobile agents (streets, junctions and traffic lights). Vehicle agents are dynamic in nature, they initiate actions and post requests to street agents, junction agents and traffic light agents to execute their intentions. On the other hand, street agents and junction agents are static. They provide street/junction traffic information and various facilities to vehicle agents. Traffic light agents can be taken between dynamic and static, i.e., they are dynamic means they can change their status dynamically to control oncoming traffic; they are static means they do not have intentions of their own and can be considered as a medium for traffic control between mobile agents and their carriers, immobile agents (streets and junctions).

We can use a *client/sever* analogy to describe the relationships among these agents. The client (vehicles) requests traffic services from the servers (streets, junctions and traffic lights). Occasionally, a junction agent will act as a client to a street agent in order to gather street traffic statistics. Also, a traffic light agent can be taken as a client to a junction agent to calculate an optimal light sequence.

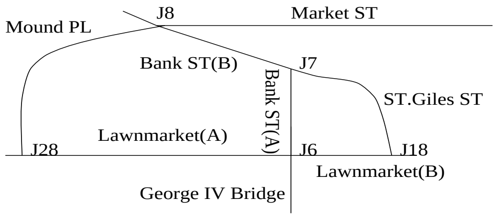

Figure 2: Part of Edinburgh urban traffic sketch map

Fig 2 is a partial sketch map of Edinburgh [1]. Suppose a car named *CAR-1* is approaching junction *J6* from *George IV Bridge* and the other two cars, *CAR-2* and *CAR-3* are travelling in *Bank Street (B)* and at junction *J7* respectively. Fig 3 introduces the relationships among the four traffic agents.

As shown in Fig 3, a street agent connects to another street agent via a junction agent; a traffic light agent is attached to a junction agent; vehicle agents communicate to street and junction agents to exchange traffic information and are controlled by traffic light agents at each junction.

When a vehicle travels from one street to another (say *CAR-2* travels along *Bank Street* via junction *J7*), it has to cooperate with the current street agent (*Bank ST (B)*), the destination street agent (*Bank ST (A)*) and the junction agent connecting them (*J7*). In addition, it also has to coordinate with the surrounding vehicle agent(s) (*CAR-3*). It will perform reactive actions towards the change occurred in its environment (i.e., the surrounding vehicles, the traffic light status, etc.). In such

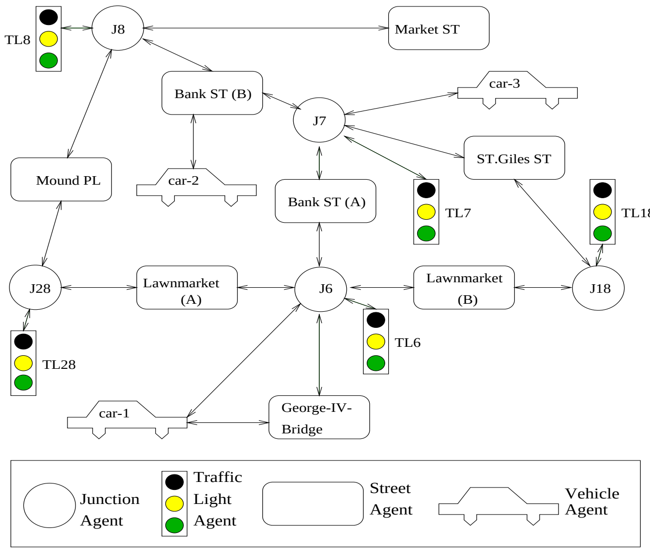

Figure 3: Relationships among the traffic agents

a system, each agent has its own roles and objectives. All of them will interact with each other to achieve their goals. Hence, the cooperation and coordination are the key point for a successful traffic simulation system.

A Testbed for Traffic Simulation

According to the multi-agent model for urban traffic problems, a cooperative traffic scheduling and controlling system (COTSCS) has been designed [1]. The overall scheme of the system is composed of three layers: the central manager (Scheduling & Planning); the traffic control points (Communicating & Coordinating); and the individual traffic entities (Controlling & Executing). Also, only individual traffic entities will be simulated in this paper. Such a testbed is needed to provide a realistic environment for analysing and evaluating the appropriateness of the system design. The testbed is implemented using the *Agent-0* architecture [6], where *Agent-0* is a specific implementation of the general purpose agent loop for the interpretation of messages passed among the agents. It is a way to organise the program structure such that multiple agents can communicate with each other within a closed world.

The agent interpreter loop

Fig 4 and Fig 5 show the internal agent loop architecture of the traffic agents where the numbered arrows show the order of the execution within the loop. The *message dispatcher* is an inter-agent message delivery function.

The agent interpreter loop (for Streets & Junctions) will read the messages in the queue, update internal agent states and make commitments to the requests; and service the current commitments, which resulting outgoing messages to the requesters. In Fig 4, the *event handler* is a group of functions that provide agents services to the clients. All external requests will firstly be validated and the internal agent states are subsequently updated if necessary. The *commitment handler* is a group of functions that return the serviced results to the requesters.

The agent interpreter loop (for Vehicles & Traffic Lights) will execute any planned action, initiate

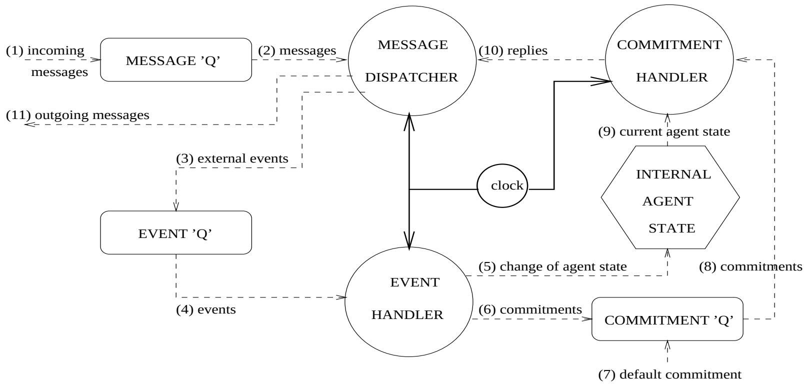

Figure 4: The agent interpreter loop (for Streets & Junctions)

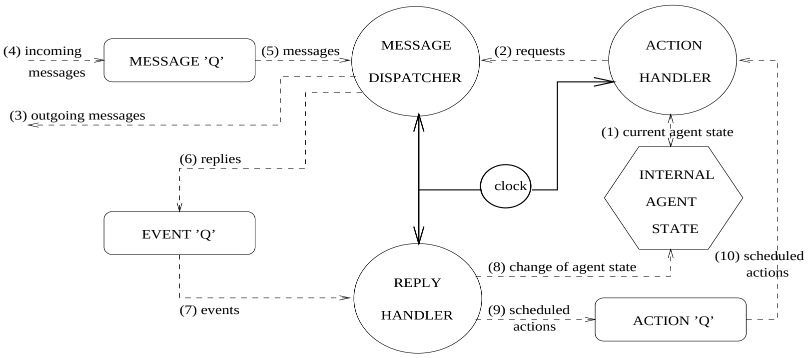

Figure 5: The agent interpreter loop (for Vehicles & Traffic Lights)

requests to the server agents; and process the replies and update the internal agent states, plan for next requests/actions. In Fig 5, the *action handler* is a group of functions that are in charge of posting requests to the agent servers and carry out any scheduled action that might change the internal agent states. The *reply handler* is a group of functions or rules that process the replies from the agent servers. The internal agent states will be updated here. Further planning and control will be evaluated before the messages are passed over to other modules.

This design is independent of whether a single or multiple process architecture is used. The current system is implemented under a single process environment, so the order of agent activation in the main simulator loop is crucial: (1) vehicle agents initiate actions/requests; (2) traffic light agents initiate actions/requests; (3) junction agents service external requests and make commitments; (4) street agents service external requests and make commitments; (5) junction agents service commitments; (6) street agents service commitments; (7) traffic light agents process replies and plan for next actions/requests; (8) vehicle agents process replies and plan for next actions/requests.

The simulation clock

Central to the operation of the simulation system is the *simulation clock*. The role of the clock is to initiate the actions of the simulation loop at the regular intervals. Like an area, the length of one cycle is an abstraction of a *unit of time* (ut) and inherently, the agents are not "real-time". The unit of abstraction can be chosen to suit the chosen platform used to run a real-time system. Furthermore, it is possible to map the real time in the real world to the simulation time in the simulated world.

Traffic simulation scenarios

From the view of vehicles, various traffic scenarios that occur in our everyday traffic world have been successfully simulated in this testbed: (1) moving along a street or a junction; (2) approaching and moving off at a junction; (3) leaving a street and entering a junction; (4) keeping a safe distance; (5) give way at a junction; (6) accident.

Conclusion

Comparing to the single process Truckworld system [7] where only one truck can be simulated, the system in this paper can simulate a world consisting of any number of vehicles, streets, junctions and traffic lights. The current system follows Truckworld's update and action mechanism but it has a much more complicated modelling mechanism where actions are not instantaneous. Unlike the command driven mechanism in Truckworld, the simulation is automated in the sense that once it has been parameterised and initialised, the simulation is continuous with no interruption.

A multi-agent version Truckworld [8] is fully distributed in nature but a truck in it does not have any notion of the rest of the world except for what it learns through its sensors. In contrast, in this system, the street and junction agents act as the vehicle agent's sensors and provide the vehicle with the information of the other vehicles in its surrounding.

Although there are some simplifications used to simulate the traffic world, the system as a whole, possesses the necessary communication and coordination facilities for the agents to interact among themselves towards cooperating and solving problems in the domain of an urban traffic scenario. The system is designed in a modular and extensible way so that addition facilities can be integrated to the system without the need to restructure the design or architecture of the system.

References

[1] M.Li, et al., "A Cooperative Intelligent System for Urban Traffic Problems", in 1996 IEEE Symp. on Intelligent Control, also Research Paper No.822, Department of Artificial Intelligence, University of Edinburgh, July 1996.

[2] A.Lux and D.Steiner, "Understanding Cooperation: an Agent's Perspective", Proc. of 1st Int. Conf. on Multi-Agent Systems, 1995, pp261-268.

[3] N.V.Findler and G.D.Elder, "Multiagent Coordination and Cooperation in a Distributed Dynamic Environment with Limited Resources", Artificial Intelligence in Engineering, 9(1995), pp229-238.

[4] Stanley Chan, Multi-Agent Traffic Simulation – Vehicle, MSc. Thesis, Department of Artificial Intelligence, University of Edinburgh, 1996.

[5] Ken Wee Chong, Multi-Agent Traffic Simulation – Street, Junction & Traffic Light, MSc. Thesis, Department of Artificial Intelligence, University of Edinburgh, 1996.

[6] Y.Shoham, "Agent-Oriented Programming", Artificial Intelligence, 60(1993), pp51-92.

[7] R.J.Firby and S.Hanks, The Simulator Manual, Technical Report YALEU/CSD/RR 563, Department of Computer Science, Yale University, 1987.

[8] C.Thomas, et al., Multi-Agent Use of the Truckworld Simulator, Technical Report Dept. of CS & E, FR-35, University of Washington, 1994.