Dynamic vehicle routing using an ABC-algorithm

R. Kroon, L.J.M. Rothkrantz

Delft University of Technology, Mekelweg 4, 2628 CD, Delft, The Netherlands

Abstract

In the past years the application of agent algorithms based on the natural behaviour of ants have shown to be successful in routing data through communication networks. Using the trail-laying abilities of ants the mobile agents are able to create well performing routing tables. In this paper an Ant Based Control algorithm is applied to the routing of road traffic trough a city. The algorithm is tested in a simulation environment that makes it possible to show the effect in different cities and circumstances. The agents do not move through a real city, but use a model of a city map. This model is supplemented with actual data from the traffic in the city. This enables the agents to divert traffic from congested routes, which improves travelling-times.

1. Introduction

Road traffic is getting busier and busier each year. Everyone is familiar with traffic congestion on highways and in the city. And everyone will admit that it is a problem that affects us both economically as well as mentally. Furthermore finding your way in an unknown city can be very difficult even with a map. Navigation systems like CARiN can help in such cases. These systems display the route to be followed when the user has entered his destination. The latest versions are also able to use congestion information to avoid trouble spots. But such information is only available for highways and not in a city.

This paper addresses the dynamic routing of traffic in a city. We want to set up a routing system for motor vehicles that guides them through the city using the shortest way in time, taking into account the load on the roads. Furthermore we want the routing system to be distributed, for more robustness and load distribution.

The routing system uses a routing algorithm based on earlier versions of Ant Based Control-algorithms. Exact routing algorithms like Dijkstra's algorithm only apply to central routing. And ant-based algorithms have proven to be superior to other distributed routing algorithms in [1,2]. In

[2] an ant-based algorithm was used for routing and load balancing in a telephony network. In [3] the algorithm is applied to packet switched networks with basic ideas taken from [1]. And now we will apply a variant of the algorithm to a traffic network in a city.

2. Ant-based control for network management

We can use the idea of emergent behaviour of natural ants to build routing tables in any network. We will apply it in a traffic network in a city, i.e. the composition of the roads and their intersections. This network is represented by a directed graph. Each node in the graph corresponds to an intersection. The links between them are the roads. Mobile agents, whose behaviour is modelled on the trail-laying abilities of natural ants, replace the ants. The agents move across the network between randomly chosen pairs of nodes. As they move, pheromone is deposited as a function of the time of their journey. That time is influenced by the congestion encountered on their journey. They select their path at each intermediate node according to the distribution of the simulated pheromone at each node. Each node in the network has a probability table for every possible final destination. The tables have entries for each neighbouring node that can be reached via one connecting link. The probabilities influence the agent's selection of the next node in their journey to the destination node. The probability of the agents choosing a certain next node is the same is the probability in the table.

The probability tables only contain local information and no global information on the best routes. Each time an agent visits a node the next step in the route is determined. This process is repeated until the agent reaches its destination. Thus, the entire route from a source node to a destination node is not determined beforehand.

Agents are launched at each node with regular time intervals with a random destination node. They travel around the network using the probabilities in the probability tables. The probabilities per destination are all filled with equal values for all nodes before the process begins.

3. Design

This section explains the design of the routing system.

3.1 Dynamic data

To route the traffic dynamically through a city we need dynamic data about the state of the traffic in the city. This can for example be directly from sensors in the road-surface. Such sensors can count vehicles and measure the speed of the vehicles. That information can be used to compute the time it takes to cover a part of the road. Another source can be the traffic information services. They can inform the system about congestion, diversions of the road, roadblocks and perhaps open bridges. And finally the vehicles themselves can provide the system with information about the path they followed and the time it took them to cover it. The current technology enables to fix the position of a vehicle with an accuracy of a few meters. That position can be communicated to the system along with the covered route.

For our routing system we will at first only use the latter type of information as dynamic data. But of course the model is open for additional types of dynamic data. The information from the vehicles is handled by a separate part of the routing system, called the timetable updating system. This subsystem takes care that the information is processed for use by the ant-based algorithm. This way one vehicle drives a certain route and sends its performance to the routing system. Another vehicle is able to use that information to choose the shortest route.

3.2 Architecture

We will now explain the structure of the system from the viewpoint of the vehicle and its driver. A vehicle is driving through a city and it wants to know the way. The driver enters the address where he wants to go and expects a routing system to tell him where to go. Besides the destination the routing system needs to know the location where the vehicle is at the moment. Therefore the vehicle sends a request to a satellite of the GPS (Global Positioning System). This is shown by arrow A in figure 1. GPS is a system that can determine a position of the sender with an accuracy of a few meters. So the GPS-satellite answers the vehicle with its current position (arrow B). This position is measured in latitude/longitude co-ordinates. In the vehicle these co-ordinates are translated in a position on a certain road with the aid of a digital map of the city. Now the vehicle has enough information to request the routing system what route to follow. The vehicle sends its position and its desired destination along with the request for the route to the routing system (arrow D). Arrow E is the answer from the routing system that contains the route that the vehicle should follow. These steps are pretty obvious, but we have skipped arrow C. This arrow indicates that the vehicle provides the routing system with information about the route is has followed since the previous time. The information consists of (1) the location and time at the moment of the previous update, (2) the location and time at this moment and (3) the route that the vehicle has followed in between these times and locations. Table 1 shows a detailed enumeration of the information that is send along the indicated arrows.

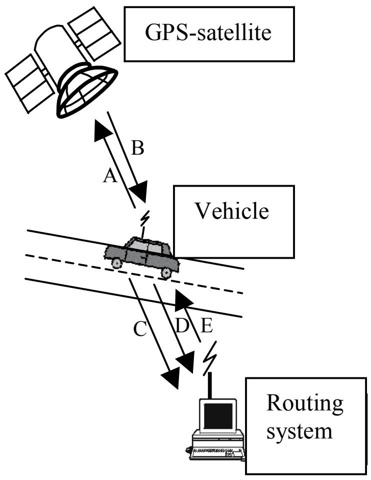

Figure 1:Communication of the vehicle

	From	**To**	**Data**
A	Vehicle	GPS-satellite	REQUEST_POSITION
B	GPS-satellite	Vehicle	ANSWER_POSITION, latitude/longitude co-ordinates
C	Vehicle	Routing system	UPDATE, previous time/position, covered road A, covered road B, covered road C, …, current time/position
D	Vehicle	Routing system	REQUEST_ROUTE, current position, destination
E	Routing system	Vehicle	ANSWER_ROUTE, road A, road B, road C, …

Table 1: Communicated data between the different objects

3.3 Routing problem

The most important problem of this research is solved by the *timetable updating system* and the *route finding system*. These two subsystems together form the *routing system*. The function of the *route finding system* will be clear: we are building a system to route vehicles. The reason why we need the *timetable updating system* is the following. The *route finding system* needs information about the state of the network. A static route finding system could use a fixed set of data, but we will use a dynamic route finding system that needs dynamic data. Those data are provided by the *timetable updating system*. That information can be for example the load of the parts of the network but a more direct and therefore more practical type of information is the time it takes to cover a road. Vehicles send information about their covered route to the *timetable updating* *system*. From that information this system computes the travelling-times for all roads and stores it in the timetable in the *memory*. Besides the timetable also a history of measurements is stored in the memory. The *route finding system* uses the information in the timetable to compute the shortest routes for the vehicles. When a vehicle requests route information, the *route finding system* sends this information back to the vehicle.

3.3.1 Route finding system

This system uses the earlier mentioned ant-based control algorithm (ABC-algorithm). This algorithm makes use of forward and backward agents. The forward agents collect the data and the backward agents update the corresponding probability tables in the associated direction. The algorithm consists of the following steps:

- At regular time intervals from every network node **s**, a forward agent is launched with a random destination **d**: F_{sd}. This agent has a memory that is updated with new information at every node **k** that it visits. The identifier **k** of the visited node and the time it took the agent to get from the previous node to this node (according to the timetable) is added to the memory. This results in a list of (**k**, $\mathbf{t}_k$)-pairs in the memory of the agent. Note that the agent can move faster than the time in the timetable.
- Each travelling agent selects the link to the next node using the probabilities in the probability table. The probabilities for the nodes that have already been visited by this agent are filtered out for this agent. Then a copy of the remaining probabilities is made for this agent and these probabilities are normalized to 1. Only this agent uses this temporary probability distribution to choose a next node. So the probability table is not updated yet.
- If an agent has no other option than going back to a previously visited node, the arising cycle is deleted from the memory of the agent.
- When the destination node **d** is reached, the agent F_{sd} generates a backward B_{ds}. The forward agent transfers all its memory to the backward agent and then destroys itself.
- The backward agent travels from destination node **d** to the source node **s** along the same path as the forward agent, but in the opposite direction. It uses its memory instead of the probability tables to find its way.

- The backward agent with previous node **f** updates the probability table in the current node **k**. The probability p_{df} associated with node **f** and destination node **d** is incremented. The other probabilities, associated with the same destination node **d** but another neighbouring node, are decremented. The used formulas are given below.

The probability of the entry corresponding to the node f from which the backward agent has just arrived is increased using the following formula:

$$P_{new,f} = \frac{P_{old,f} + \Delta P}{1 + \Delta P} \tag{3}$$

Here, Pnew,i is the new probability, Pold,i the old probability and ΔP the probability increase. ΔP should be inversely proportional to the age of the forward agent. The formula we use is:

$$\Delta P = \frac{a}{t} + b \tag{4}$$

Where a and b are constants and t is the trip-time of the forward agent from this node to the destination node. This trip-time is the sum of the trip-times from this node to the destination node of the forward agent. We do not take into account that the conditions of the traffic network can change from the moment that the node is visited by the forward agent and the updating of the backward agent.

The other entries in the probability table with the same destination but other neighbouring nodes are decreased using the formula:

$$P_{new,i} = \frac{P_{old,i}}{1 + \Delta P}, \quad \forall\, i \neq f \tag{5}$$

4. Experiments

As a proof of concept we run an experiment in a traffic network as displayed in Figure 4. For vehicles from the road between intersections 5 and 6 or 6 and 7 there is only one reasonable path to their destination. The alternative paths, which go via intersections 3, 4 and 8, will hardly ever be more attractive for those vehicles. For the vehicles from the roads between intersection 1 and 2 and intersection 9 and 10 there are two reasonable options. They can take the northern path via intersections 3, 4 and 8, or they can take the southern path via intersection 6. The length of both routes is different. Taking the length and maximum speed into account, the path for a vehicle from 1/2 to 9/10 can be covered in 75 seconds when passing intersections 2, 6 and 9. The alternative via intersections 2, 3, 4, 8 and 9 will cost at least 82 seconds. So all vehicles from 1/2 to 9/10 and vice versa will initially take the southern route. But intersection 6 of the southern route is a point where many vehicles from different roads join and cross each other's path. Therefore it is controlled by traffic lights. These traffic lights make the crossing safer and the priority for vehicles from different roads is distributed more fairly. On the other hand the traffic might perceive a considerable delay at this intersection because of the heavy load and the traffic lights. This will cause the northern path to be more attractive for vehicles from intersection 1/2 to 9/10 and vice versa. So we expect the vehicles that follow the advice of the Routing system to drive via the northern roads, where there are no delaying intersections. This should result in faster routes for these vehicles as opposed to the vehicles that do not use the Routing system.

4.1.1 Results

The first run of this experiment provided the following graphs (Figure 2 and Figure 3). These graphs show the differences in the average travel time of standard and smart vehicles over period of 40 minutes (2400 seconds). The standard vehicles do not use the Routing system, the smart vehicles do. The value at the end, after 2400 seconds, is the average of all measured travel times since the start of the experiment until the end.

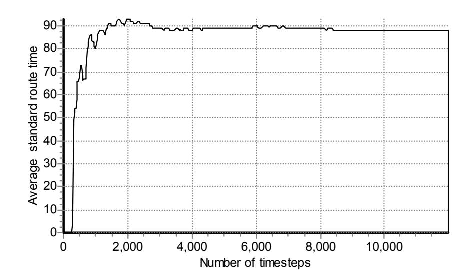

Figure 2: Average standard route time

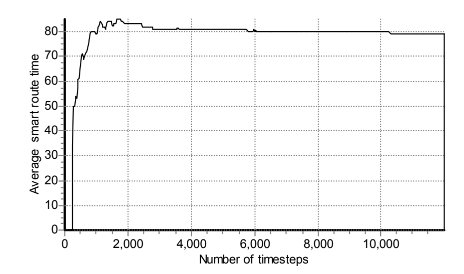

Figure 3 Average smart route time

When we zoom in on the graphs (Figure 2 and Figure 3) we see that the (rounded) value for the standard vehicles is 88 seconds and the value for the smart vehicles is 79 seconds. So the overall profit for the smart vehicles is 10 % on average as opposed to the standard vehicles, which do not use the Routing system.

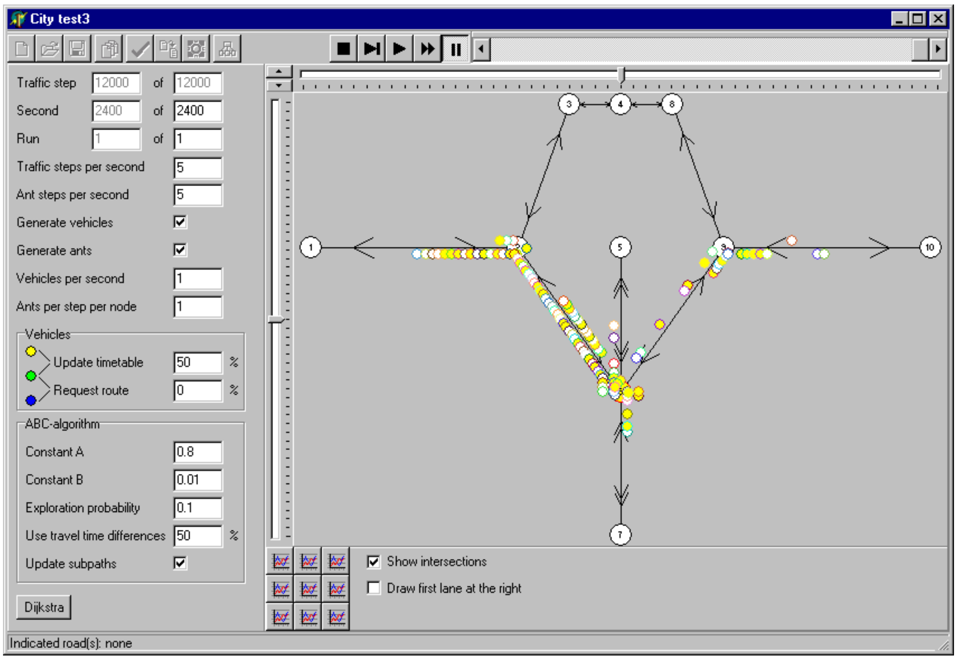

Figure 4: Screenshot of the City program after 2400 second without Routing system

References

[1] G. Di Caro and M. Dorigo. *AntNet: distributed stigmergetic control for communication networks*. Journal of Artificial Intelligence Research (JAIR), 9, 317-365, 1998

[2] R. Schoonderwoerd, O. Holland, J. Bruten, L.J.M. Rothkrantz. *Load balancing in telecommunication networks*, Adaptive Behaviour, 5, 2 , 1997.

[3] L.J.M. Rothkrantz, J.C. Wojdel, A. Wojdel, H. Knibbe. *Ant based routing algorithms*, Neural Network World, Vol. 10, No.3, July 2000

[4] H. Dibowski, L.J.M. Rothkrantz *Hierarchical routing system using Ant Based Control*, ITS-report, July 2003