Road Traffic Management
Based on Ant System and Regulation Model

Cyrille Bertelle, Antoine Dutot, Sylvain Lerebourg and Damien Olivier
Laboratoire d'Informatique du Havre,
25 rue Philippe Lebon, BP 540
76058 Le Havre Cedex - FRANCE
E-mail : {Cyrille.Bertelle, Antoine.Dutot, Sylvain.Lerebourg, Damien.Olivier}@univ-lehavre.fr

KEYWORDS: Transport, distribution, road traffic, ant algorithm, multiagent systems, neural networks.

ABSTRACT

Decision support system for road traffic management can be used for freight transport, people transport but also for site evacuation. We deal with two aspects of the decision support system in a same global architecture: one for road users to choose the shortest path in time between two points and the other for road traffic management to regulate the traffic and to avoid jam. These two aspects interact. The cartography is represented by a weighted digraph. The weights evolve according to the traffic and the graph is therefore dynamic. The search of the shortest path is based on an ant algorithm because it is well suited for dynamic environment. The regulation system is based on a neural network.

1 CONTEXT

1.1 Road Traffic Management Necessity

The transport development must face up to many constraints like:

- Increase of traffic whereas urban infrastructures are not very evolutionary;
- Substructures realization and expansion limitation due to the available space and the costs;
- Problem of pollution which comes from transport. Reduction of the loud and atmospheric pollution for example;
- Deregulation and concurrency between the modes of transport.

So, it is necessary to find solutions to manage road traffic. Two aspects can be considered, in one hand Decision Support System (DSS) to help and to inform users and in the other hand regulation system based on control planning (Virtual Message Signs (VMS), traffic lights, etc).

1.2 Global Architecture

We propose a global architecture based on two main parts (see figure 1):

- The real world which is split in three elements:
  - the traffic which contains, in one hand, all mobile elements (cars, pedestrians, ...) described with different levels of autonomous behaviour and, in the other hand, spatio-temporal organizations which are predictable (school outs, ...) or not (jam, accident ...);
  - the environment which contains all the road infrastructure and logistic planning;
  - the control system which contains sensors (webcams, data traffic magnetic sensors, ...) and effectors (VMS, traffic lights, ...).
- The model which is split in the following elements:
  - information collection and processing in order to use them on the solving level;
  - a dynamic weighted digraph representing these informations and the traffic flow;
  - a regulation system based on this graph and managing the control system;
  - a DSS which use the dynamic graph and the regulation control. A multimodal interface informs and helps different users with respect to their profiles.

The information update and its adaptive treatment give the dynamic aspect of the global architecture as described in figure 1. So, it is typically a complex system model including retro-action phenomena. In this paper we develop two points of this architecture: the decision support system which suggests shortest paths obtained from a dynamic graph and regulation system based on multi-layer perceptron with backpropagation algorithm.

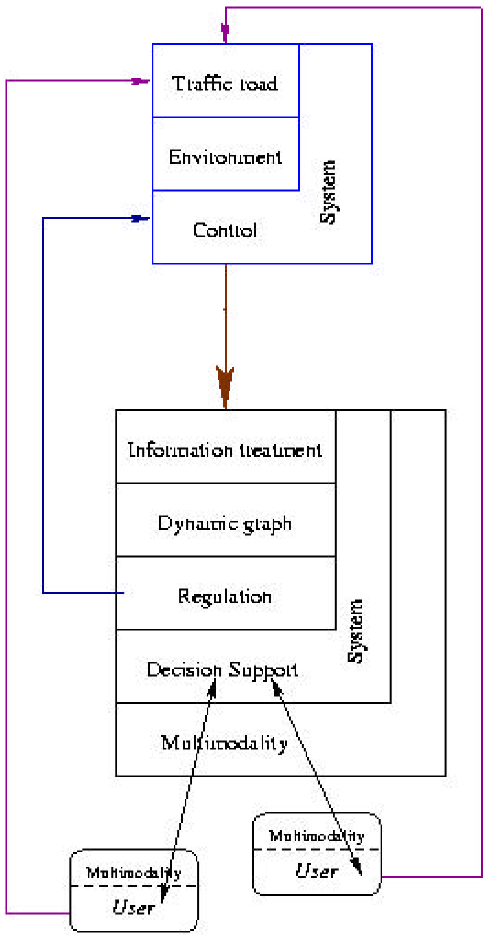


Figure 1: Global architecture for road traffic management

2 ANT ALGORITHM FOR CONSTRAINED PATHS COMPUTATION

2.1 Ant Algorithms

Ant algorithms are a class of meta-heuristics that can yield near-optimal solutions to hard optimization problems, where algorithms that yield exact solutions are not an issue. Ant algorithms maintain a population of agents that exhibit a cooperative behaviour (Langton, 1987) by continuously foraging their territories to find food (Gordon, 1995) using optimal paths, creating bridges, constructing nests, etc.

This form of self-organization appears from interactions that can be either direct (e.g. mandibular, visual) or indirect. Indirect communications arise from individuals changing the environment and other responding to these changes: this is called stigmergy. For example, ants deposit signals named pheromones in the environment that influence others: the more pheromone on a path, the more ants tend to follow it. As pheromones evaporate, long paths tend to have less pheromone than short ones, and therefore are less used than others (binary bridge experiment).

Such an approach is robust and well supports parameter changes in the problem. Besides, it is intrinsically distributed and scalable. It uses only local informations (required for a continuously changing environment), and find near-optimal solutions. Ant algorithms has been applied successfully to various combinatorial optimization problems like the Travelling Salesman Problem (Dorigo and Gambardella, 1997), routing in networks (Caro and Dorigo, 1997), (White, 1997) or for distributed simulation (Bertelle et al., 2002b) but also to DNA sequencing (Bertelle et al., 2002a), graph partitioning (Kuntz et al., 1997) and clustering (Faieta and Lumer, 1994).

2.2 Graph

The cartography is represented by a weighted digraph $G = (V, E)$ where V is a set of vertices representing crossroads or any other significant information (school, town hall . . .) and $E = V{\times}V$ is a set of directed edges $e = (v_i , v_j)$. Thus each segment of a street that is between two adjacent vertices as defined previously is represented by either one or two directed edges. Two directed edges, one in either direction, are used if the street is two-way, and a single directed edge is used if it is a one-way segment. The edge weight w_{ij} between the vertices v_i and v_j is a dynamic factor which represents the time to cross the edge (v_i , v_j) and the traffic load.

2.3 Dynamic aspects

Weights evolve according to the traffic and the graph is therefore dynamic and we have to find paths in this graph. These changes are one of the major motivation for using ant algorithms. A monotonic approach is one way to search paths on a graph. It consists in regularly applying a computation on a frozen copy of the dynamic graph, then trying to use this information, though the real graph is still evolving. This approach is problematic: the graph can have changed during computation and results may not be usable any more, creating discrepancies between the real state and calculated paths. Furthermore, it is not incremental, each time the algorithm is performed anew.

Another way is to use an anytime algorithm. The dynamic graph is considered as a changing environment for computing entities that travel on the graph, taking into account the changes as they appear, and storing the solution directly in the graph, as an effect of their evolution. Ant algorithms are well suited for that kind of task as it has been shown in (Dorigo et al., 1996).

This approach is almost implicitly distributed. This would not create many communications since the algorithm only uses local informations and stores results directly in the graph (that is, directly in the computing resources local memory).

2.4 Algorithm

We search in the graph some paths between two vertices v_0 and v_n . The resolution method is distributed and based on auto-organization mechanisms. We continually release numerical ants on the dynamic graph, and allow them to find routes between pairs of vertices. The ants deposit numerical pheromones on edges. The amounts of pheromone deposited is a function of the length and congestion of paths. Ants are attracted by weights of edges and pheromones. The evaporation allows to forget bad paths. The ants tend to converge on paths which are the fastest.

To be able to distribute the computation, we have divided the algorithm in two parts and for each we have a specific time.

- The environment. It is represented by the dynamic graph. Its major role is to manage the ant population, evaporation phenomenon and simulation of weights on the edges. We store also in the vertex v_n the shortest path which comes from v_0 , the minimal global cost W_{0n} of the path from v_0 to v_n . Due to the dynamic change of weights the duration of the shortest path may change when another ant covers the path crossing the same vertices and we note t_{0n} the instant where the ant has found the same path. For a given step, we have:

```
tenv = discrete time of the environment
BEGIN
  birth of ants on the vertex v0
  pheromone evaporation (see (2))
  weights update
  IF t0n << tenv // This depends on path
    // length. No ants have used the path
    // since a long time
  THEN W0n = + ∞
  ENDIF
  tenv = tenv + 1
END
```

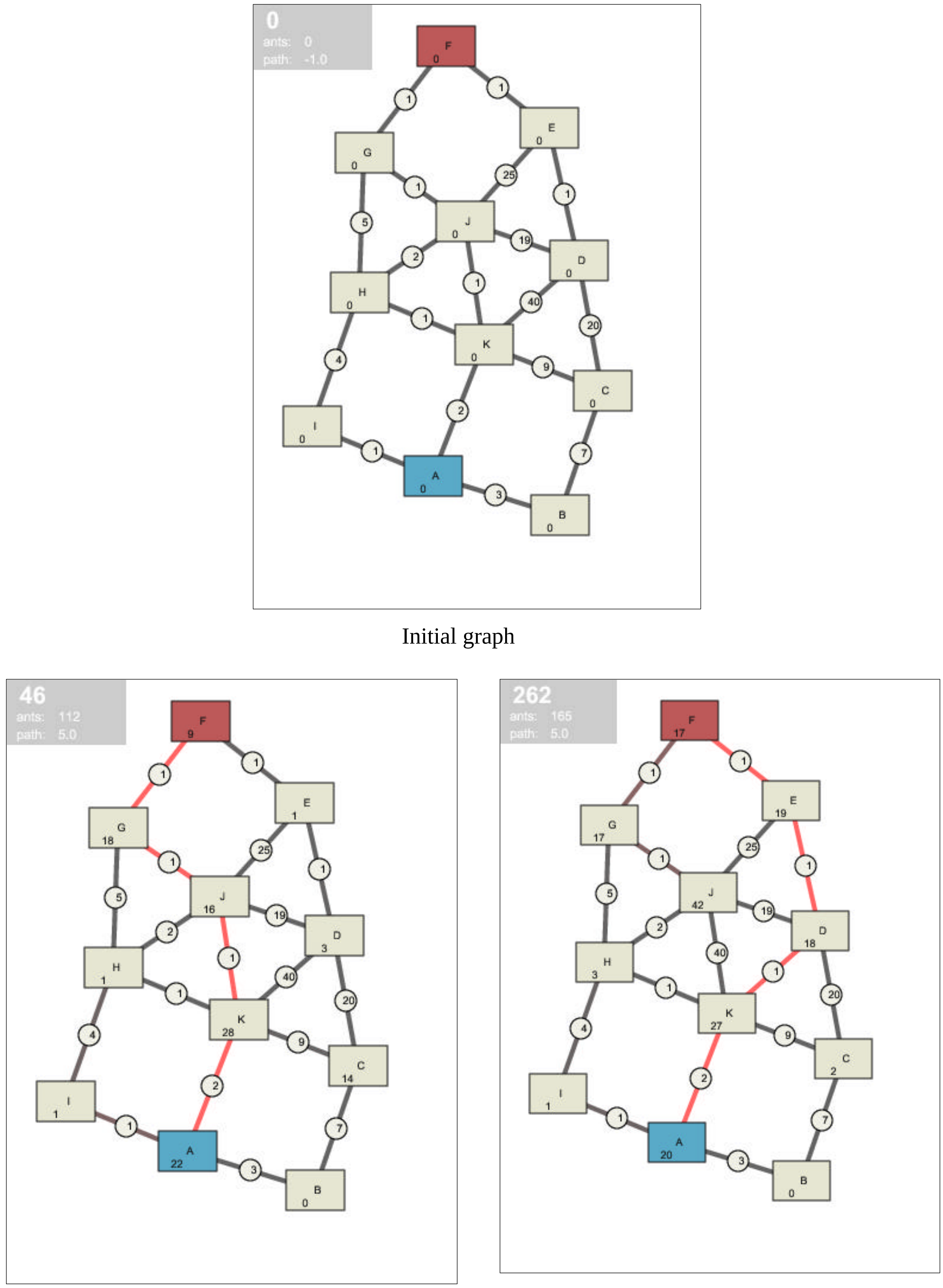
- The ants. Ants try to go from the vertex v_0 to the another vertex v_n . Ants manage their displacements according to times and pheromones. They also drop pheromones on edges. Three states are possible for an ant looking for food, reaching the final vertex v_n, and coming back to the source. For one ant located on v_i, we have :

```
tant = discrete time for the ant
vertex = vi
ant_state ∈ {search, arrived, go_back}
BEGIN
  IF ant_state == search
    THEN
    //The ant must choose an adjacent vertex
    //to i
    Ai = set of the adjacent vertices of i
         Which have not been traversed yet
         by the ant
    FORALL j ∈ Ai DO
      Compute the probability pij (see (1))
        that the ant chooses to hop from
        the vertex i to j
    ENDFOR
    Select the next vertex vk
      according to the probability pij
    Wait during the time wik - 1
    vertex = vk // Move to k
    IF vk == vn
      THEN ant_state = arrived
    ENDIF
  ENDIF
  IF ant_state == arrived
    THEN
    Update if necessary shortest path
      and times t0n
    ant_state = go_back
  ENDIF
  IF ant_state == go_back
    THEN
    Deposit pheromone on path
      used by the ant (see (4))
    death of the ant
  ENDIF
  tant = tant + 1
END
```

Let q_{ij} be the amount of pheromone trail deposited on the edge connecting i and j, w_{ij} the weight of the edges which depends on the time of the traffic flow to connect the location i and j, it is a dynamic variable. The probability that an ant when it is located on i choose j is:

$$p_{ij} = \frac{(q_{ij})^a \left(\frac{1}{w_{ij}}\right)^b}{\sum_{k \in A_i} (q_{ik})^a \left(\frac{1}{w_{ik}}\right)^b} \qquad (1)$$

Where A_i is the set of adjacent vertices of i which have not been traversed yet by the ant. The amount of pheromone q_{ij} on the edge (v_i , v_j) is modified by the environment and by the ants. The environment regularly updates this pheromone quantity using an evaporation rate, noted (1 - ρ):



Initial graph

Shortest path

Shortest path after a modification of weight

Figure 2 : Search of a shortest path on a simple dynamic

$$q_{ij}^{new} = r \quad q_{ij}^{old} \qquad (2)$$

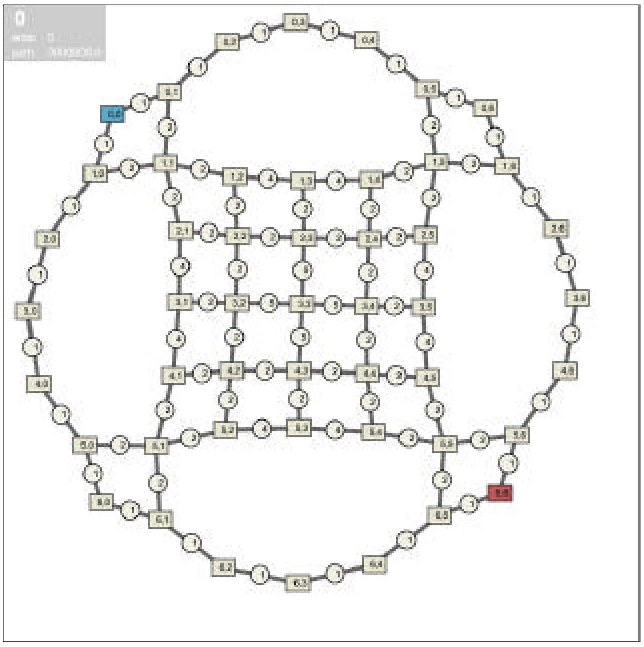
where $0 \leq \rho \leq 1$ and q_{ij}^{old} and q_{ij}^{new} are respectively the pheromone quantity before and after the update.

An ant which has found a path between the two vertices v_0 and v_n and so come back to start vertex, modify the pheromone quantity by reinforcement rate, noted Δq :

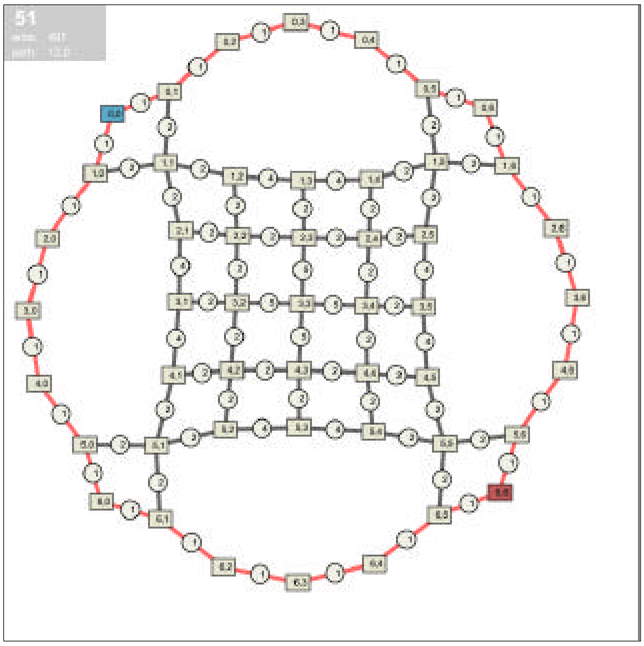
$$\Delta q = \frac{K}{W_{ij}} \qquad (3)$$

where K is a constant and W_{ij} the global cost of the path between i and j.

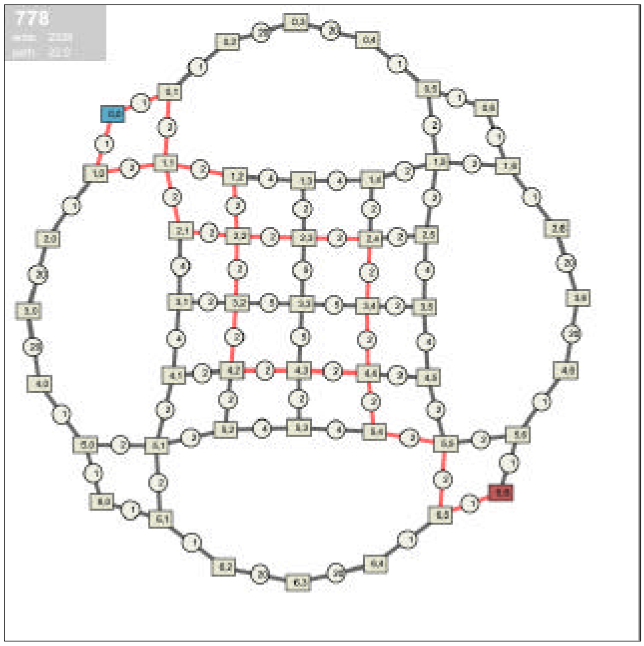
$$q_{ij}^{new} = q_{ij}^{old} + \Delta q \qquad (4)$$



Initial graph



Shortest path



Shortest path after a modification of weight

Figure 3 : Search of a shortest path on a simple dynamic Manhattan

2.5 Results

We show two examples of the algorithm execution. The first one is a very simple graph (see figure 2), the first node is A and the last node is F . We search the shortest path while we make vary the weight of an edge (K, J) located on the shortest path, during the calculation. The result is presented on the third graph. In this example, at each environment time step, 4 ants are released, $\alpha = 3$, $\beta = 1$, $\rho = 0.6$ and K = 1.

The second one is based on a Manhattan representation with a ring road (see figure 3). The first graph shows the initial situation, the ring road is the fastest way then we jam it, so a new path is detected by the ants. The second graph shows the shortest path obtained which takes the ring road, the last one is the solution when the ring road is jamed. In this example, at each environment time step, 10 ants are released, $\alpha = 3$, $\beta = 1$, $\rho = 0.9$ and K = 2.

3 REGULATION MODELLING

We deal now with road traffic regulation. We use for this purpose a model, using an agent-based representation for road traffic and a neuronal model for the regulation.

This study (Foote, 2002) presented in the following will look at traffic flow on a Manhattan-style road grid. At each crossroad, there is a traffic lights system deciding which cars are going to cross. Cars enter the grid from the outside and decide which direction they wish to use at each crossroad. We use the Madkit package (Gutknecht and Ferber, 1997) to manage the agent world. The neural network is a multi-layer perceptron implementing a backpropogation algorithm.

3.1 Controlling multi-agent system

With the rise of multi-agents system, computer scientists are coming across the control problem concerning such virtual world regulation. In general, we cannot predict how a population of many agents, acting on each other, will evolve. Since multi-agent systems often contain hundreds or thousands entities, we can no longer control each of these elements precisely as we don't know how it might evolve. We have to develop a system to control the situation globally.

The generic agent organization used in this work is based on theoretical study of A. Cardon (Lesage et al., 1999). He describes how a multi-agent system can be divided into three types of agents:

- *Aspectual Agents* are the basic agents that represent the target population. In our case, they represent cars and traffic lights in a town;

- *Morphological Agents* deal with aspectual agents measurements. They collect only some informations which lead to describe evolutive and adaptive organizational aspects. It is a kind of projection of all agent characteristics onto a smaller dimensional space. In our case, a morphological agent plays a statistic collection service, taking into account for example, cars position and information about their displacements.

- *Analytical Agents* are some kind of rulers of our agent population, looking at the statistics provided by the morphology agents, and then acting on them to control the global behaviour of the system. The analytical agents do not directly modify the behaviour of any particular agent, but rather, indirectly shape the evolution of the aspectual agents as a whole.

Here, we present an application of this theoretical model, based on a neural network. Our problem is as follows: how can we maximise the flow of traffic through a road network? The input layer of our neural network will process information from the morphological space of the aspectual agents and then give an output figure which represents the global state of the network. This figure can then be used to decide on the action to be taken to increase traffic flow. Since in past situations we know how the system evolved, we can also train the neural network to anticipate the evolution of the system.

3.2 Neuronal Approaches Based Models

The regulation model uses an agent-based description which is analysed by a neural network based on multi-layer perceptron.

Agent-based description

The simulator is decomposed in three main parts:

- The *environment* is composed of a dynamical graph, as described in section 2.2. A more accurate representation is used, each road has its own length and width and is implemented in bidimensional grid where cars evolve from one position to another;

- The *traffic lights* manage the cars circulation at each crossroad. Each one finds out the identities of its neighbours, it looks for cars which arrive at its crossroad and knows the direction that each car want to go. The lights have two possible behaviours. The *dumb light* acts in automatic mode. It sends requests to its neighbours to know which space is available on their crossroads. It lets cars go to its choosen direction if there is space available at the target crossroad. In *cooperative light* mode, all traffic lights proceed sorting queues that they manage. The longest queue is first managed and the associated light lets cars go to their choosen direction if space is available, else the second-longest queue acts and so on ...

- The *cars* can be in one the three following states. They are in the state *moving* when they have to go to one crossroad (graph node) to another if there is no car in front of it. When a car reaches its choosen crossroad without having other cars in front of it, it changes its state to *atLight* one. In this state, it sends a message to the light telling which way it wants to go. It then waits until the light gives it permission to move. If a car has other ones in front of it, during its move, it changes its state to the *waiting* one.

Moreover, the simulation manages input and output fluxes between the simulated town (as Manhattan-style road grid) and the exterior.

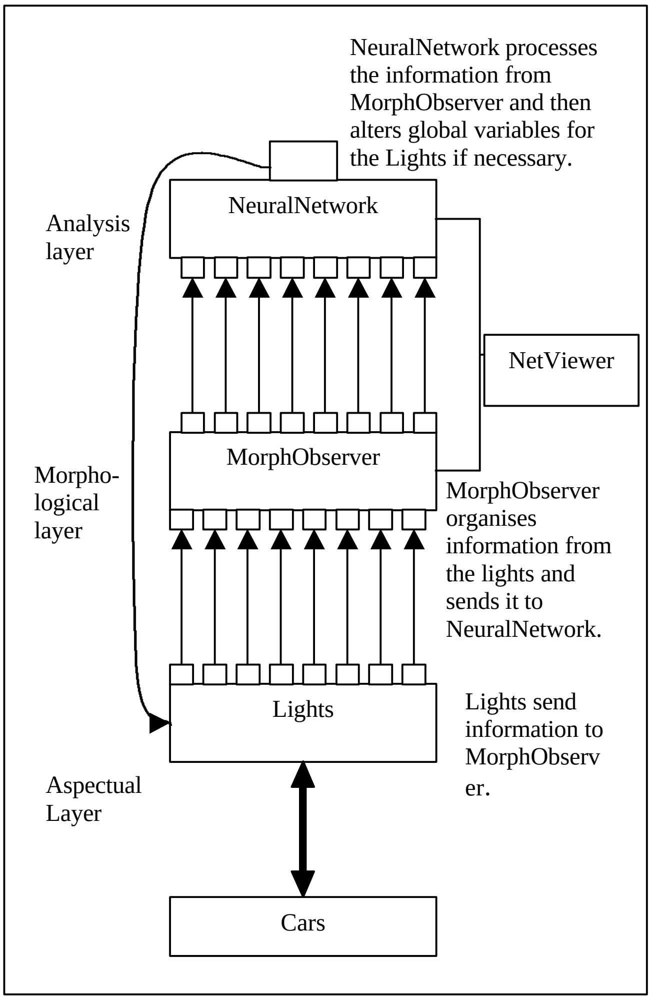


Figure 4 : Neural network based regulation

Multi-Layer Perceptron Regulation

The neural network used for the regulation is a multi-layer perceptron. It is the analysis layer of the generic agent organisation described previously (see figure 4).

The network computes global variables to reduce traffic jams. The three states of its output are: *clear*, *busy* and *getting blocked* corresponding to no danger of gridlock, slight danger and danger of gridlock. So the retro-action of this analysis layer on the aspectual layer consists in altering the following variables:

- *waitTime* corresponds to the delay between sending batches of cars through the lights;

- *carDispersion* corresponds to the authorized cars number able to come into the town from the exterior.

3.3 Experimentations

We show in figure 5, two windows of the visual interface desktop of the simulator which represent respectively a schematic view of the traffic and a traffic observer curve. This last information gives the number of cars which are moving at each step of the simulation. In this example, the regulation leads to the preservation of the fluidity, the global number of moving cars is preserved between 100 and 150 units.

3.4 Regulation Feed-Back on Dynamic Graph Modelling

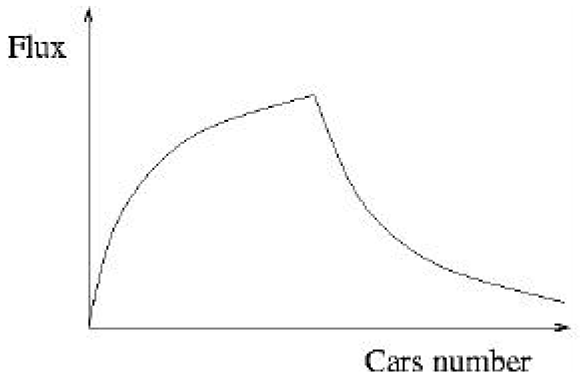


Figure 6: Flux model for road traffic

The regulation system which acts on *waitTime* variable, is able to give N_{ij} the cars number on each edge of the graph modelling the road traffic. Taking into account some physical characteristics of each road modeled with edge, we use a classical model to compute the traffic flux (number of cars by second) associated with each edge (see figure 6) and noted F_{ij} . This model is based on the definition of a critical number of cars under which the fluidity is maintained and over which

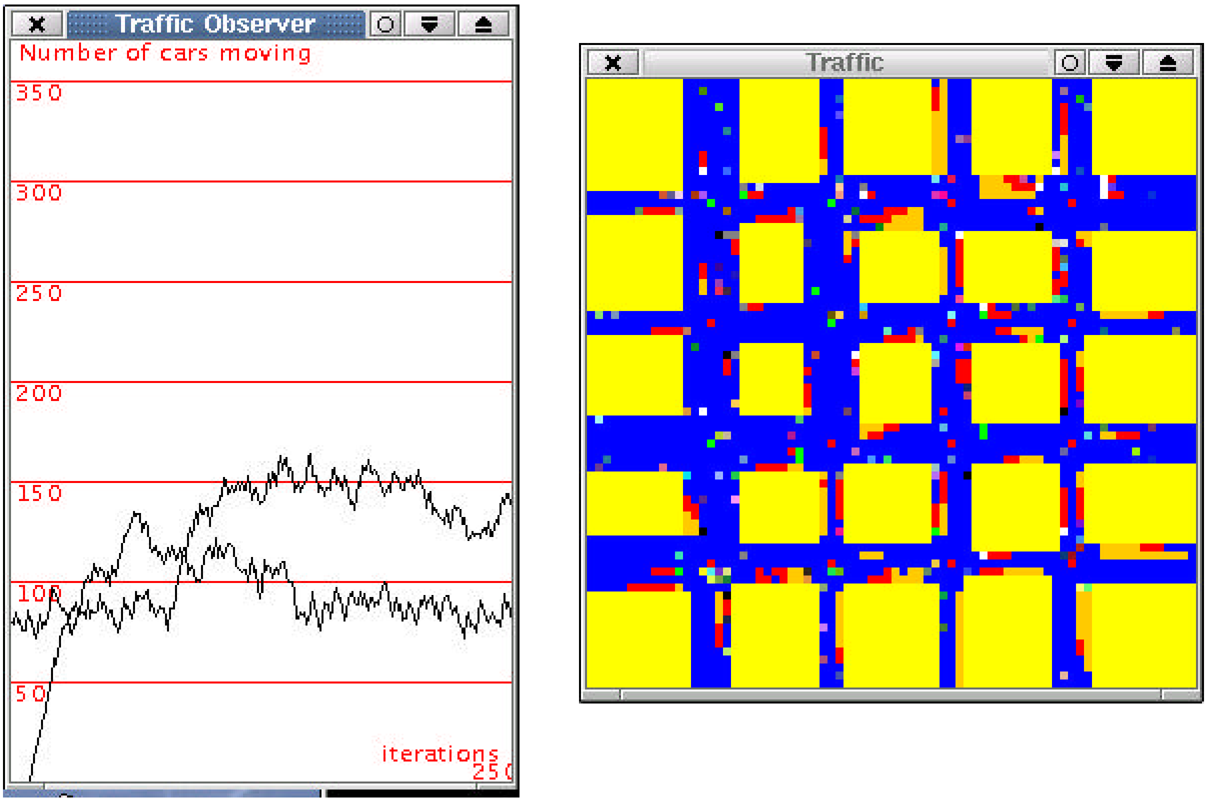


Figure 5 : Regulation experimentation

the fluidity quickly falls down. We then compute a characteristic fludity-based time, expressed as:

$$d_{ij} = \left(\frac{F_{ij}}{N_{ij}} \right)^{-1} \qquad (5)$$

This characteristic time contributes to the regulation feed-back of the regulation system on the dynamic graph modelling the road traffic. In fact, each edge weight is the sum of an observed time for crossing the road, noted r_{ij} and the characteristic fluidity-based time defined above:

$$w_{ij} = r_{ij} + d_{ij} \qquad (6)$$

4 CONCLUSION

We are developing an architecture of both a regulation system and a decision support system based on a dynamic graph. Ant algorithms are used and well suited for adaptive aspects and anytime approaches of dynamic traffic flow. Neural networks are used and well suited for traffic flow regulation. We actually work on future development concerning management of heterogeneous informations flows from any kind of sources (satellites, webcams, sensors) and multimodal interfaces for the different users. We are searching to extract the most important and urgent informations using organizations of cooperatives/antagonists agents. Multi-agent systems are adapted to find emergent evolutionary solutions in dynamic problems.

References

Bertelle, C. ; Dutot, A. ; Guinand, F. and Olivier, D. (2002a). Dimants: a distributed multi-castes ant system for dna sequencing by hybridization. In *NETTABS 2002, AAMAS 2002 Conf*, Bologna (Italy).

Bertelle, C. ; Dutot, A. ; Guinand, F. and Olivier, D. (2002b). Distribution of agent based simulation with colored ant algorithm. In *ESS2002*, pages 39-- 43, Dresden (Germany).

Di Caro, G. and Dorigo, M. (1997). *Antnet: A mobile agents approach to adaptive routing*. Technical report, IRIDIA, Université libre de Bruxelles, Belgium.

Dorigo, M. and Gambardella, L. (1997). Ant colony system: A cooperative learning approach to the traveling saleman problem. *IEEE Transactions on Evolutionary* Computation, 1(1):53--66.

Dorigo, M. ; Maniezzo, V. and Colorni, A. (1996). The ant system: optimization by a colony of cooperating agents. *IEEE Trans. Systems Man Cybernet*., 26:29--41.

Faieta, B. and Lumer, E. (1994). Diversity and adaptation in populations of clustering ants. In *Conference on Simulation of Adaptive Behaviour*, Brighton.

Foote, A. (2002). *Controlling the global behaviour of a massive multi-agent landscape. A practical example using neural network*. Technical report, LIH.

Gordon, D. (1995). The expandable network of ant exploration. *Animal Behaviour*, 50:995--1007.

Gutknecht, O. and Ferber, J. (1997). *Madkit: Organizing heterogeneity with groups in a platform for multiple multi-agents systems*. Technical report, LIRMM, Montpellier University, http://www.madkit.org.

Kuntz, P. ; Layzell, P. and Snyers, D. (1997). A colony of ant-like agents for partitioning in VLSI technology. In *Fourth European Conference on Artificial Life*, pages 417--424, Cambridge, MA:MIT Press.

Langton, C. editor (1987). *Artificial Life*. Addison Wesley.

Lesage, F. ; Tranouez, P. and Cardon, A. (1999). A multiagent prediction of the evolution of knowledge with multiple points of view. In *KAW'99*, Banff, Canada.

White, T. (1997). *Routing with swarm intelligence*. Technical Report SCE-97-15.