On large size problems of dynamic network assignment and traffic equilibrium: Computational principles and application to Paris road network

Fabien Leurent, Vincent Aguiléra

HAL Id: hal-00348411

https://hal.science/hal-00348411

PAPER TITLE

On large size problems of dynamic network assignment and traffic equilibrium: Computational principles and application to Paris road network

AUTHORS' NAMES

Vincent AGUILERA, Fabien LEURENT

SUBMISSION DATE

July 31, 2008

Revised version: November 14, 2008

AUTHOR 1 (CORRESPONDING AUTHOR)

Prof. Fabien Leurent

Université Paris-Est, Laboratoire Ville Mobilité Transport

19 rue Alfred Nobel, Champs sur Marne, 77455 Marne la Vallée Cedex 2, France

Tel: (33) 164 152 111

Fax: (33) 164 152 140

e-mail: fabien.leurent@enpc.fr

AUTHOR 2

Vincent Aguiléra

Université Paris-Est, Laboratoire Ville Mobilité Transport

19 rue Alfred Nobel, Champs sur Marne, 77455 Marne la Vallée Cedex 2, France

Tel: (33) 164 153 681

Fax: (33) 164 152 140

e-mail: vincent.aguilera@enpc.fr

WORD COUNT: 5,987 + 6 FIGURES

On large size problems of dynamic network assignment and traffic equilibrium: Computational principles and application to Paris road network

Vincent Aguiléra, Fabien Leurent (Université Paris-Est, Lvmt)

Abstract

The paper reports on the algorithmic treatment and computer implementation of a macroscopic dynamic traffic assignment model called LADTA. The modelling assumptions and the mathematical analysis founding the model are first stated. Detailed descriptions of the main algorithms are given, together with the principles of the computer implementation. It is shown how the design of the software architecture allows for distributed computation of a traffic assignment. The practical ability of this implementation to tackle with large size networks is illustrated by an application to the Paris road network, which comprises around 1,300 zones and 39,000 links.

Keywords

Manuscript Text

INTRODUCTION

Dynamic Traffic Assignment (DTA) models have been the topic of continuous developments in recent years, on both the research side (*1* to *9*) and the implementation side. User friendly packages dedicated to network and traffic planning studies and including DTA have appeared on the market (*10,11,12*). Two main classes of DTA models are usually distinguished in the literature: on the one hand, models based on a microscopic or mesoscopic traffic simulators and on the other hand analytical macroscopic models.

Our model, called LADTA (for Lumped Analytical DTA) belongs to this second class. It has been designed as an extension of classical static assignment models (*13*), with special emphasis on the time-varying features including notably: dynamic network operation; traffic phenomena such as queuing; the time and space pattern of flow by origin-destination pair and user class; the choice of departure time.

The objective of this paper is to report on the algorithmic treatment and computer implementation of LADTA. Its ability to tackle with large size networks is illustrated on the Paris metropolitan area. This assignment problem has about 1,300 traffic assignment zones and 39,000 directional links. It is the authors' belief that the algorithmic approach may be of interest to the researchers as well as the software developers involved in traffic assignment; and that the computational performance and the application outcomes may be of interest to practitioners.

The paper comprises three sections, followed by a conclusion. It is structured as follows. Section 1 presents the modelling assumptions and the mathematical analysis that found LADTA. Section 2 focuses on the main algorithms and computer implementation. It also illustrates, through an example, how computations can be distributed. Section 3 reports on the application to the road network of Paris metropolitan area. The emphasis is put on (i) application data and results and (ii) runtime indicators during iterations. Some practical issues that make topics for further research are discussed in conclusion.

1.   LADTA MODEL AND SOLUTION SCHEME

1.1   Systems analysis of network traffic assignment

Network traffic assignment involves four cause-and-effect relationships, as depicted in Figure 1:

1) *the formation of travel services*: Informally, one can say that the user of a transportation network who traverses a route *r* to reach a destination, starting at a given departure time *h*, is using a travel service *(r,h)*. For each class of users, a travel service provides a certain quality of service and has a certain price. These depend on the quality of service and the price of the arcs traversed along the route. Solving the problem of service formation yields optimum services on the basis of the economic attributes of the arcs.

2) *user choice*: if the user of a transportation network behaves as a rational economic agent, he will choose the optimum service w.r.t. his/her own evaluation criteria. So for each user class the volume that is assigned to each service depends on the nature of the services and the a priori economic structure of the user class. Typically, this structure describes the distribution of the economic trade-offs between price and quality of service.

3) *volume loading*: the volume on each arc depends on the volumes on the paths which traverse the arc, and hence on the service volumes.

4) *traffic flowing*: for each arc the quality of service, in particular the traversal time, depends on both the arc entry volume and the arc flowing parameters, such as exit capacity and minimum traversal time. Solving the flowing problem yields the exit volume and the actual traversal time, on the basis of the entry volume and flowing parameters.
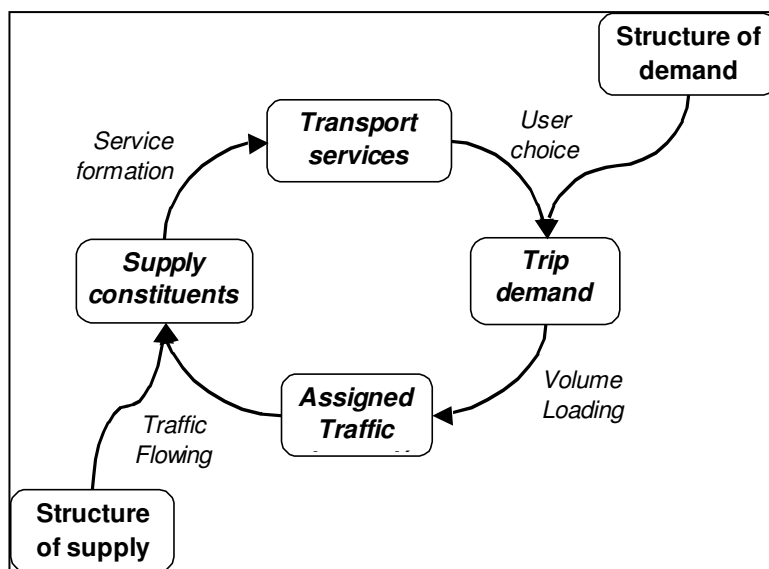

FIGURE 1: Graphical summary of the four causal relationships.

1.2   Modelling strategy in LADTA

LADTA extends Beckmann's static model (*14*) in the following ways:

- *Temporal profiles* are used to model volumes, times, prices and costs.

- *Service formation* is addressed using a dynamic route search algorithm which extends the classical shortest path algorithm used in the static case.

- *User choice* can include route choice and also departure time choice. Only the definition of the choice options differs from the static case.

- *Volume loading* is addressed by dynamic, recursive network propagation of volume "atoms" (time intervals with constant flow rate): this extends Dial's tree loading algorithm from static to dynamic case.

- *Traffic flowing* is performed at the arc level: inputs are the temporal profiles of exit capacity, minimum traversal time and cumulated entry volume. Outputs consist of the temporal profiles of the cumulated exit volume and the actual traversal time.

Analytical formulae together with efficient algorithms are available for all partial problems, which are assembled into an equilibrium scheme. The overall mathematical formulation amounts to a fixed point problem with respect to the arc volume temporal profiles. The equilibrium problem is addressed by a method of successive averages. A more detailed description can be found in *(6)*.

1.3   Sketch mathematical formulation

The clock time is a variable h whose values belong to the set H. The transportation network is modelled by a set N of nodes n, and a set A of directional arcs a, whose nodes belong to N. Arc a has a reference free-flow travel time t_{a0} and exit capacity $\kappa_a(h)$ in cumulated volume. In addition r denotes a route, which is defined as a continuous sequence of network arcs without redundancy (i.e. no arc is traversed more than once). We use $\{\,a \in r\,\}$ to denote the set of arcs which make up a route r, and $\{\,r \in a\,\}$ to denote the set of routes that traverse an arc a. $\{r > a\}$ denotes the sub-path of r which is strictly upstream of a. This subset is empty when $a \notin r$. $\{\,r < a\,\}$ denotes the sub-path of r which is strictly downstream of a.

The user classes are modelled by a set U of classes u. The origin-destination (O-D) pairs are modelled by a set I_u of pairs i for each user class u. A demand segment z consists of a category of users belonging to a class u, on an O-D pair i, with a temporal preference η and an economic trade-off β, which may pertain e.g. to the trade-off between travel time and price. The volume of segment z is denoted by δQ_z and $Q_z(h)$ stands for the demand volume of segments identical to z except perhaps the temporal preference, cumulated up to departure instant h.

A service $s = (r, h)$ is a route and departure time pair. Supply is characterized by the variables of price, p, traversal time, t, passage time, h, and the cumulated volumes X^+ at entry and X^- at exit (depending on passage time and at the level of arcs a or routes r). At the time h of entrance into a route r, a user of class u experiences a traversal price of $p_{ur}^+(h)$ and an actual traversal time of $t_{ur}^+(h)$, which cannot be less than the minimum traversal time $t_{ur0}^+(h)$ that would be experienced under free-flow condition.

To a user of segment $z = (u, i, \beta, \eta)$ the generalized cost of service $s = (r, h)$ is denoted by $G_z(s)$, for instance $G_z(r, h) = p_{ur}^+(h) + \beta_z\, t_{ur}^+(h) + \delta(h + t_{ur}^+(h), \eta)$ with β_z the segment unit cost of time and $\delta(\hbar, \eta)$ the delay cost of arriving at $\hbar = h + t_{ur}^+(h)$.

Service formation amounts to the recursive formulation of the service attributes (time, price etc) with respect to the destination and the exit time, $\hbar$:

$$t_{ur}^-(\hbar) = \sum_{a \in r} t_{ua}^-(\overline{H}_{u, r<a}(\hbar)),\tag{1a}$$

$$p_{ur}^-(\hbar) = \sum_{a \in r} p_{ua}^-(\overline{H}_{u, r<a}(\hbar)),\tag{1b}$$

in which $\overline{H}_{u,r<a}(\hbar)$ is the time of entry in the downstream sub-path "$r < a$" to exit r at $\hbar$: it is the inverse function of $h \mapsto H_{u,r<a}(h) = \hbar$, whereas $t_{ur}^-(\hbar) = t_{ur}^+(\overline{H}_r(\hbar))$ and $p_{ur}^-(\hbar) = p_{ur}^+(\overline{H}_r(\hbar))$.

About *user choice*, let us assume here for simplicity a cost-minimizing behaviour in a deterministic choice situation, under inelastic demand. The segment elementary volume δQ_z that corresponds to a departure time interval $[h, h+\delta h[$ is assigned to services *s* in elementary flows δX_{sz}^+ in the following way, in which S_z denotes the set of services available to segment z and μ_z denotes a dual variable:

$$\delta X_{sz}^+ \geq 0 \tag{2a}$$

$$\sum_{s \in S_z} \delta X_{sz}^+ = \delta Q_z \tag{2b}$$

$$G_z(s) \geq \mu_z, \ \forall s \in S_z \tag{2c}$$

$$(G_z(s) - \mu_z)\delta X_{sz}^+ = 0, \ \forall s \in S_z \tag{2d}$$

Volume loading is the problem of assigning the route volumes X_{ur}^+ to the network arcs *a*, yielding arc volumes X_{ua}^+. It involves also the travel times $t_{u,r>a}^+(h)$ on the sub-path of *r* upstream of *a* through the exit-entry function of time shift, $\overline{H}_{u,r>a}$. Stated formally,

$$X_{ua}^+(h) - X_{ua}^+(h_0) = \sum_{i \in I_u} \sum_{r \in R_{ui} \cap a} X_{ur}^+[\overline{H}_{u,r>a}(h)] - X_{ur}^+[\overline{H}_{u,r>a}(h_0)] \tag{3}$$

Traffic flowing is governed by two physical principles that limit the exit volume and one economic principle of time minimization. The first principle, called the arrival constraint, states that the exit volume, $X_{ua}^-(h)$, is limited to the entry volume, $X_{ua}^+(h)$, with a time lag consisting of the minimum traversal time t_{ua0}. The second principle, called the capacity constraint, states that the exit volume during interval $]h_0, h]$, $X_{ua}^-(h) - X_{ua}^-(h_0)$, is limited to the exit capacity, $\kappa_a(h) - \kappa_a(h_0)$. The third, economic principle called the time minimization principle, states that the volume flows as quickly as possible; capacity is used to the full when there is a queue at the exit. Thus

$$X_{ua}^-(h) = \inf_{h_{\min} \leq h-s \leq h} X_{ua}^+[h-s-t_{ua0}] + \kappa_a(h) - \kappa_a(h-s) \tag{4}$$

Thus $t_{ua}^+(h)$ is recovered as the solution argument of $\inf\{t : X_{ua}^-(h+t) > X_{ua}^+(h)\}$.

Supply-demand equilibrium is reached when the volumes and travel conditions (time and price) are functions of each other in the four problems of service formation, user choice, volume loading and traffic flowing. To formulate equilibrium we define generic functions on the basis of the problem equations established previously:

- $(t_{US}^+, p_{US}^+) = F_S(t_{UA}^+, p_{UA}^+)$ for *service formation*. Function F_S is defined by equations (1a, b).

- $X_{UR}^+ \in F_U(t_{US}^+, p_{US}^+)$ for *user choice*. Function F_U is defined by eqn (2). The use of a membership sign in place of an equals sign provides a reminder that F_U is multivalent.

- $X_{UA}^+ = F_V(X_{UR}^+)$ for *volume loading*. Function F_V is defined by eqn (3).

- $t_{UA}^+ = F_F(X_{UA}^+)$ for *traffic flowing*. Function F_F is defined by eqn (4).

Together, these four formal dependencies characterize the supply-demand equilibrium. Mathematically, this is a nonlinear complementarity problem in the variables X_{UR}^+. If we express the variables X_{UR}^+, t_{UA}^+ and p_{UA}^+ in terms of X_{UA}^+, we obtain an equivalent reduced formulation which is a fixed point problem:

$$X_{UA}^+ \in F_V \circ F_U \circ F_S[F_F(X_{UA}^+), p_{UA}^+] \qquad (5)$$

Equation (5) is the dynamic counterpart of Beckmann's static model (1956) and can be solved identically, by iteratively updating a vector of arc flows. The method for updating is to combine a current vector of class-arc entry flow profiles $X_{UA}^{+[k]}$ with an auxiliary vector $Y_{UA}^{+[k]}$ defined by $Y_{UA}^{+[k]} \in F_V \circ F_U \circ F_S[F_F(X_{UA}^{+[k]}), p_{UA}^+]$, weighted by fractions $1 - \zeta_k$ and ζ_k, respectively, to obtain the current vector of the next iteration $k+1$:

$$X_{UA}^{+[k+1]} = (1 - \zeta_k)X_{UA}^{+[k]} + \zeta_k Y_{UA}^{+[k]}$$

In this abstract formulation, the arc travel times t_{UA}^+ define an underlying chronology for flow propagation along the network. As the convex combination at the arc level cannot maintain the chronology, the arc flow profiles $X_{UA}^{+[k+1]}$ do not rely on a given chronology – at least, not until equilibrium is reached. This issue of chronological basis is addressed in (*8*).

2. ALGORITHMS AND IMPLEMENTATION

The LADTA model has been implemented targeting the application to problems of very large size, notably so to road networks for which the set of variables exceeds the amount of memory available on a single computer. This requirement led us to design distributable algorithms to tackle with the most computer intensive tasks in a reasonable amount of time.

This section provides in § 2.1 formal definition of *temporal profiles* (i.e. time dependant variables), in order to present three core algorithms: the arc-based multi-class traffic flowing algorithm MCPQTFLOW (in § 2.2), the dynamic least cost path algorithm DLCP (in § 2.3) and the volume loading algorithm RLOAD (in § 2.4). One distinctive feature of the last two algorithms is that they are destination-based. This allows for a simple and efficient distribution scheme, which is described in § 2.5, where the software architecture of the LTK is detailed.

2.1 Temporal profiles

The basic variables in the LADTA model are functions of time. In the algorithms presented in the sequel, those functions are represented by piece-wise linear functions of time called *temporal profiles*. Formally, a temporal profile is defined as follows:

Definition 1, Temporal Profile: *a temporal profile* P *is a (finite, non empty) set of n real numbers triples* $\{(h_i, p_i, \dot{p}_i), i = 1 \dots n\}$, *such that* $\dot{p}_1 = 0$ *and* $(h_i)_{i=1\dots n}$ *is a strictly increasing sequence. It defines a piece-wise linear function* $P(h)$ *as follows:*

$$P(h) = \begin{cases} p_1 & if \quad h \in \left[-\infty; h_1\right] \\ p_i + \dot{p}_i(h - h_i) & if \quad h \in \left]h_{i-1}; h_i\right] \\ p_n & if \quad h \in \left]h_n; +\infty\right] \end{cases}$$

The set of operators defined on profiles includes basic ones (e.g. linear combination) and some others among which two deserve mention at this stage: **shift** and **hcomp**. Both are binary operators on profiles that take one argument in a particularly important class of profiles called *traversal time profiles*.

Definition 2, Traversal Time Profile: *A temporal profile* T *is a traversal time profile if* $T(h)$ *is strictly positive and satisfies also the First In – First Out (FIFO) property:*

$$\forall (h_1, h_2), (h_1 < h_2) \implies h_1 + T(h_1) \le h_2 + T(h_2)$$

Definition 3, shift: *Let* X *be a profile and let* T *be a traversal time profile.* $Y = \text{shift}(X, T)$ *is a profile such that* $Y(h + T(h)) = X(h)$.

Definition 4, hcomp: *Let* X *be a profile and let* T *be a traversal time profile.* $Y = \text{hcomp}(X, T)$ *is a profile such that* $Y(h) = X(h + T(h))$.

2.2 Arc-based multi-class traffic flowing

For the reasons exposed in section 1, LADTA models the dynamicity of traffic phenomena in a simplified way: on each arc, flows from various user classes interact only at capacity bounded bottlenecks called *point queues*, located at the exit of the arc. The point queue model is briefly introduced in § 2.2.1. It forms the basis of the arc-based multi-class traffic flowing algorithm, exposed in § 2.2.2.

2.2.1 The point queue model

Let define a *volume profile* and a *capacity profile* as follows:

Definition 5, Volume Profile: *a temporal profile* $X = (h_i, x_i, \dot{x}_i)_{i=1\dots n}$ *is a volume profile if* $X(h_1) = 0$ *and* $X(h)$ *is increasing.*

Definition 6, Capacity Profile: *a temporal profile* $K = \{(h_i, k_i, \dot{k}_i), i = 1 \dots n\}$ *is a capacity (rate) profile if* $K(h) \ge 0$ *for all* h, $\dot{k}_i = 0, i = 1 \dots n$ *and* $k_n > 0$.

Let now consider a point-wise queue q, having a capacity profile K. The time to flow n vehicles out of the queue, starting at instant h, is given by:

$$\text{tflow}(n, h, K) = [\min\{h' : h' > h, \int_h^{h'} K(h)dh \geq n\}] - h$$

Let $X = (h_i, x_i, \dot{x}_i)_{i=1...n}$ be a volume profile entering queue q. The queue length (i.e. the number of vehicles present in q) at an instant h is denoted $N_q(h)$. Let h_q be the first instant, if it exists, such that (i) $N_q(h_q) = 0$ and (ii) the entry flow (i.e. the instant variation of the entry volume) exceeds the exit capacity rate $K(h_q)$. Starting from h_q, N_q will begin by increasing, then remain positive until queue vanishes at instant h_{end} such that the volume having entered the queue from h_q to h_{end}, $X(h_{end}) - X(h_q)$, is matched by the volume having exited the queue, $\int_{h_q}^{h_{end}} K(h)dh$. From the definitions of a volume profile and a capacity profile, N_q is obviously a profile. The corresponding *time in queue* profile T_q, valid for $h \in [h_q; h_{end}]$, is such that:

$$T_q(h) = \text{tflow}(N_q(h), h, K)$$

Given these definitions, the PQTFLOW algorithm computes the time-in-queue profile for all h, taking as inputs:

- $X = \{(h_i, x_i, \dot{x}_i), i = 1...n\}$: a volume profile entering the point queue.

- K: a capacity profile.

2.2.2 Multi-class traffic flowing

Let now consider:

- a network $G = (N, A)$

- a set of user classes U

- a set of input volume profiles $X_{A,U}^+ = \{X_{a,u}^+\}_{a \in A, u \in U}$

- a set of free flow traversal time profiles $T_{0,A,U} = \{T_{0,a,u}\}_{a \in A, u \in U}$

- a set of exit capacity profiles $K_A = \{K_a\}_{a \in A}$

The purpose of the arc-based multi-class traffic flowing algorithm MCPQTFLOW is to establish a set of traversal time profiles, $T_{A,U} = \{T_{a,u}\}_{a \in A, u \in U}$, assuming that the flows from various user classes interact only at point queues located at the exit of the arcs.

Algorithm 1: MCPQTFLOW

Main loop:

    for each $a \in A$ **do**

        Let $X_{q,a}^+ \leftarrow \sum_{u \in U} \text{shift}(X_{a,u}^+, T_{0,a,u})$

        Let $T_{q,a} \leftarrow \text{PQTFLOW}(X_{q,a}^+, K_a)$

        for each $u \in U$ **do**

$$T_{a,u} \leftarrow T_{0,a,u} + \text{hcomp}(T_{q,a}, T_{0,a,u})$$

end for

end for

2.3 Dynamic least cost path algorithm

Let define a traversal cost profile as follows:

Definition 7, Traversal Cost Profile: *a temporal profile* $C = (h_i, c_i, \dot{c}_i)_{i=1...n}$ *is a traversal cost profile if* $C(h)$ *is strictly positive.*

Given a directed graph $G = (N, A)$, two distinguished nodes o and d in N, a set of arc traversal time profiles $T_A = \{T_a\}_{a \in A}$, a set of arcs *traversal cost profiles* $C_A = \{C_a\}_{a \in A}$, r a finite, possibly cyclic, path from o to d, a the first arc in r, and $r < a$ the sub-path of r downstream of a, then the cost of the path r when leaving o at instant h can be defined inductively as follows:

$$C_r(h) = C_a(h) + C_{r<a}(h + t_a(h))$$

If $R_{o,d}$ denotes the set of paths from o to d, then we define a *dynamic least cost path* from o to d as a mapping that associates each departure time h to a path $r \in R_{o,d}$ such that:

$$C_r(h) = \min\{C_{r'}(h) : r' \in R_{o,d}\}$$

The purpose of the dynamic least cost path algorithm DLCP is to compute the dynamic least cost paths to a destination d, from all nodes in N and for all departure times. As a prerequisite, let us define a *routing profile*.

Definition 8, Routing Profile: *let* $G = (N, A)$ *be a directed graph,* d *a distinguished node in* G, $n \neq d$ *a node in N,* Idx *a one-to-one mapping between* A *and* $[1; \#A]$. *A temporal profile* $R_{n,d} = (h_i, r_i, \dot{r}_i)_{i=1...m}$ *is a routing profile from* n *to* d *if for all* h, $R_{n,d}(h) \in \{\text{Idx}(a) : a = (n, j) \in A\}$.

The DLCP algorithm yields a set $C_{N,d} = \{C_{n,d}\}_{n \in N}$ of path traversal cost profiles to d, and a set $R_{N,d} = \{R_{n,d}\}_{n \in N}$ of routing profiles to d such that, for all h and all $i \in N$, $i \neq d$, and for all $a \in A$, the following constraint holds – a Bellman condition:

$$[R_{i,d}(h) = \text{Idx}(a)] \Rightarrow [C_{i,d}(h) = \min\{C_{a'}(h) + C_{j'}(h + T_{a'}(h)) : a' = (i, j') \in A\}]$$

Informally, the DLCP algorithm proceeds almost like the Dijkstra algorithm: starting from the destination, nodes that violate this constraint are successively relaxed. Costs are initially set to infinite values for all nodes except the destination of which the cost is zero. When a node is relaxed, all instants are processed jointly. If it violates the Bellman condition, its predecessors are put in a list waiting for being processed. One crucial difference with the classical Dijkstra algorithm is that a given node may appear more than once in the waiting list, since arcs traversal costs vary with time. So the order in which nodes are taken out of the waiting list is a key feature: processing a

node "too early" leads to unnecessary relaxation of its predecessors. Intuitively, if a distance can be defined between the destination and any other node in the graph, the idea is to take out of the list at first the nodes closest to the destination according to that distance. This distance must reflect the evolution of the cost of paths towards the destination. For instance, if nodes in the graph are assigned xy coordinates, and if arcs traversal costs are related to the length of the arcs, then the Euclidian distance is a good candidate. A more formal mathematical analysis is provided in (*15*), but with no specification of the computer implementation.

Algorithm 2: DLCP

Inputs:

$G = (N, A)$: a directed graph.

d : a distinguished node in N .

$T_A = \{T_a\}_{a \in A}$: a set of traversal time profiles.

$C_A = \{C_a\}_{a \in A}$: a set of traversal cost profiles.

dist : a mapping that associates any node n in N with a number $\text{dist}(n) > 0$.

Outputs:

$R_{N,d} = \{R_{n,d}\}_{n \in N}$: a set of routing profiles to destination d .

$C_{N,d} = \{C_{n,d}\}_{n \in N}$: a set of cost profiles to the destination d . When leaving from node n at instant h , $C_{n,d}(h)$ is the cost to reach d along route $R_{n,d}(h)$.

Local variables:

Q : a list of nodes waiting for treatment.

Initialisation

    for each n in N **do**

        $C_{n,d} \leftarrow \{(0, +\infty, 0)\}$

        $R_{n,d} \leftarrow \{(0, 0, 0)\}$

    end for

    $Q \leftarrow \{d\}$

    $C_{d,d} \leftarrow \{(0, 0, 0)\}$

    $R_{d,d} \leftarrow \{(0, 0, 0)\}$

Main loop

    while $Q \neq \varnothing$ **do**

        Let $j \in Q$ such that $\text{dist}(j) = \min\{\text{dist}(n) : n \in Q\}$

        $Q \leftarrow Q \setminus \{j\}$

        for each arc $a = (i, j)$ **do**

$$\text{Let } C = C_a + \text{hcomp}(C_{j,d}, T_a)$$

$$\text{Let } H = \{h : C(h) < C_{i,d}(h)\}$$

if $H \neq \emptyset$

Let C' a cost profile such that

$$C'(h) = \begin{cases} C(h) & \text{if } h \in H \\ C_{i,d}(h) & \text{otherwise} \end{cases}$$

$$C_i \leftarrow C'$$

Let $R'_{i,d}$ a routing profile such that:

$$R'_{i,d}(h) = \begin{cases} \text{Idx}(a) & \text{if } h \in H \\ R_{i,d}(h) & \text{otherwise} \end{cases}$$

$$R_{i,d} \leftarrow R'_{i,d}$$

$$Q \leftarrow Q \cup \{i\}$$

end if

end for

end while

2.4   Volume Loading

Given a set of routing profiles to a destination, demand profiles associated to the origins, and arcs traversal time profiles, the RLOAD algorithm computes the entry volume profile of each arc, starting from the origins, and following the routing profiles according to the arcs traversal times. This is the dynamic, destination-based version of Dial's cascade algorithm (*16*); it is analogous to some dynamic network loading algorithms (*3*).

Algorithm 3: RLOAD

<u>Inputs:</u>

$G = (N, A)$: a directed graph.

d : a distinguished node in N .

$T_A = \{T_a\}_{a \in A}$: a set of traversal time profiles.

$R_{N,d} = \{R_{n,d}\}_{n \in N}$: a set of routing profiles to the destination d .

$X_{N,d} = \{X_{n,d}\}_{n \in N}$: a set of volume profiles. $X_{N,d}$ describes the demand to the destination d . $X_{d,d}$ must be null.

dist : a mapping that associates each node n in N with a number $\text{dist}(n) > 0$

<u>Output:</u>

$$X_{A,d} = \{X_{a,d}\}_{a \in A} : \text{a set of volume profiles.}$$

Local variables:

Q : a list of nodes waiting for treatment.

Initialisation

for each $a \in A$ **do** $X_{a,d} \leftarrow 0$

for each node $n \in N$ **such that** $X_{n,d} \neq 0$ **do** $Q \leftarrow Q \cup \{n\}$

Main loop

while $Q \neq \varnothing$ **do**

Let $i \in Q$ such that $\mathrm{dist}(i) = \max\{\mathrm{dist}(n) : n \in Q\}$

$Q \leftarrow Q \setminus \{i\}$

for each arc $a = (i, j)$ **do**

Let $H_a = \{h : R_{i,d}(h) = Idx(a)\}$

Let X'_a a volume profile such that $X'_a(h) = \displaystyle\int_{t \in H_a \cap]-\infty; h]} \mathrm{d}X_{i,d}(t)$

$X_{a,d} \leftarrow X_{a,d} + X'_a$

$X_{j,d} \leftarrow X_{j,d} + \mathrm{shift}(X'_a, T_a)$

if $j \neq d$

$Q \leftarrow Q \cup \{j\}$

end if

end for

$X_{i,d} \leftarrow 0$

end while

2.5   Software architecture of the LTK

The computer implementation of the LADTA model is called the LADTA ToolKit (LTK for short). The block diagram in Figure 2 gives an overall picture of the software architecture of the LTK.

The LADTA C API implements computer intensive algorithms, basic operations on profiles, and efficient data I/O routines to manage large profile sets. It is compiled as a shared library, using the GNU C compiler `gcc`. The LTK itself contains a set of command line programs to realise the various tasks needed while computing a DTA. This allows for driving a simulation using a scripting language (e.g. Tcl, Python) as glue, while tools included in the LTK are used as building blocks.
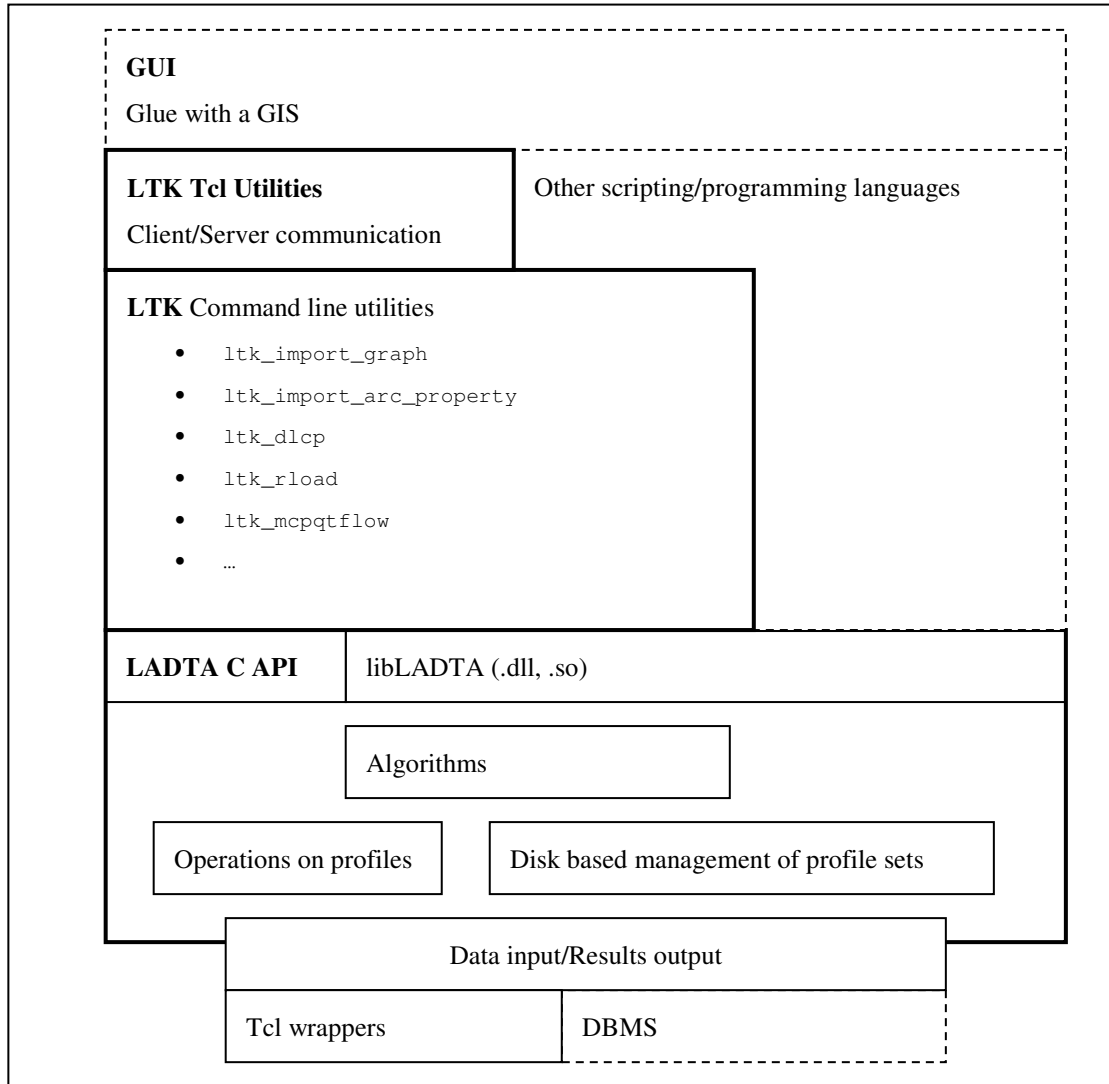
Large size problems of dynamic network assignment and traffic equilibrium

**FIGURE 2: Multilayer software architecture of the LADTA Tool Kit.
Dashed frames represent possible extensions.**

On top of the LTK, the LTK Tcl Utilities contain a set of Tcl scripts that (i) ease the use of the LTK and (ii) manage client/server communication via TCP/IP sockets when the computation of a DTA is distributed. A user-friendly GUI is under development, using the free GIS called GRASS.

In what follows, we illustrate through an example how to use the LTK for solving a one-class, fixed-demand DTA, using distributed computations and the following resolution scheme:

$$FC(P_A, T_{A,k-1}) \rightarrow C_{A,k} = P_A + \beta \cdot T_{A,k-1}$$

$$RC(C_{A,k}, T_{A,k-1}, Q_I) \rightarrow R_{I,k}$$

$$VL(T_{A,k-1}, Q_I, R_{I,k}) \rightarrow X_{A,k}^+ = \zeta_k \cdot Y_{A,k}^+ + (1 - \zeta_k) \cdot X_{A,k-1}^+$$

$$TF(X_{A,k}^+, K_A, T_{0,A}) \rightarrow T_{A,k}$$

where P_A is a set of arc toll price profiles and β a unit cost of time.

Large size problems of dynamic network assignment and traffic equilibrium

STEP 0: Initialization

We have at our disposal n+1 computers connected over a network. n computers are servers, named `S1` to `Sn,` and one is the client. The client is the computer that drives the computation: it sends requests to servers and takes care of synchronisation. Servers run the LTK Server (`ltkd`) and wait for requests from the client. The servers and the client share a common directory (`LTK_DATA`) for data exchange, and they all store their private data in a local directory `ltk_data`.

Inputs are supposed to be initially stored in `LTK_DATA`. At start, this directory contains the following files:

- `G`: stores a directed graph $G = (N, A)$

- `KA`: stores the set of arc capacity profiles $K_A = \{K_a\}_{a \in A}$

- `T0A`: stores the set of free flow traversal time profiles $T_{0,A} = \{T_{0,a}\}_{a \in A}$

- `PA`: stores the toll price profiles $P_A = \{P_a\}_{a \in A}$)

- `Q,d`: stores demand volume profiles to destination `d`, taken from the dynamic OD trip table $Q_I = \{Q_i\}_{i \in I}$.

During the initialization, the client:

- makes a local copy of `G, KA, T0A` and `PA`.

- makes a local copy of `T0A`, named `TA,0`.

- assigns a set of destinations to each server, so that each server is assigned a comparable number of destinations. It then requests the servers to copy their assigned `Q,d` from `LTK_DATA` to their local `ltk_data`, and waits for completion.

- requests the servers to make a local copy of `G`, and waits for completion.

Once the initialization step has been completed, the client performs a number of iterations. Iteration `k`, starting at `k=1`, is divided into five successive steps.

STEP 1, FC (Formation of Costs): $FC(P_A, T_{A,k-1}) \rightarrow C_{A,k} = P_A + \beta \cdot T_{A,k-1}$

During this step, some arc properties are used to compute the arcs traversal costs at iteration `k`. In the general case, which arc properties are used and how they are combined is left to the modeller as long as the resulting profile set contains arc traversal cost profiles. For the purpose of our example, we compute a generalised cost using arc traversal times $T_{A,k-1}$ at previous iteration, arc toll price profiles P_A and a unit cost of time β.

To complete this step, the client:

- creates the local file `CA,k` by executing the `ltk_lincomb` tool with the appropriate arguments (`1 P beta TA,k-1`)

- copies `TA,k` and `CA,k` to `LTK_DATA`.

- requests the servers to make a local copy of `TA,k` and `CA,k`, and waits for completion.

Large size problems of dynamic network assignment and traffic equilibrium

STEP 2, RC (Route Choice): $\mathrm{RC}(C_{A,k}, T_{A,k-1}, Q_I) \rightarrow R_{I,k}$

Depending on the arcs traversal costs and times, the route choice step updates the set of routes for every OD pair and departure time. Typically, the DLCP algorithm is used here to compute an optimal route per OD and departure time.

To complete this step, the client:

- requests each server `Si` to compute the set of routing profiles `R,k,d`, for all destinations `d` assigned to `Si` during initialisation. Server `Si` does so by executing the `ltk_dlcp` tool, with the appropriate arguments (`G d CA,k TA,k`). `R,k,d` is stored in `Si`'s `ltk_data`.

- waits for completion by all servers.

STEP 3, Volume Loading (VL):

$$\mathrm{VL}(T_{A,k-1}, Q_I, R_{I,k}) \rightarrow X_{A,k}^+ = w_k \cdot Y_{A,k}^+ + (1-w_k) \cdot X_{A,k-1}^+$$

Routes computed in the RC step are used, jointly with the demand, to compute the entry volume profiles per arc at iteration $X_{A,k}^+$. The auxiliary state $Y_{A,k}^+$ is obtained by summing on the client side partial loads computed by the servers using the RLOAD algorithm (see Algorithm 3).

STEP 4, Traffic Flowing (TF): $\mathrm{TF}(X_{A,k}^+, K_A, T_{0,A}) \rightarrow T_{A,k}$

Arc entry volumes are "flowed" to compute the profiles of arc traversal time. If using the MCPQTFLOW algorithm, the client runs the `ltk_mcpqtflow` tool to compute `TA,k`.

STEP 5: Stopping criterion

The client decides on whether to stop the computation or to do one more iteration, based on a fixed criterion (e.g. a maximum number of iterations has been reached), or the value of a convergence indicator.
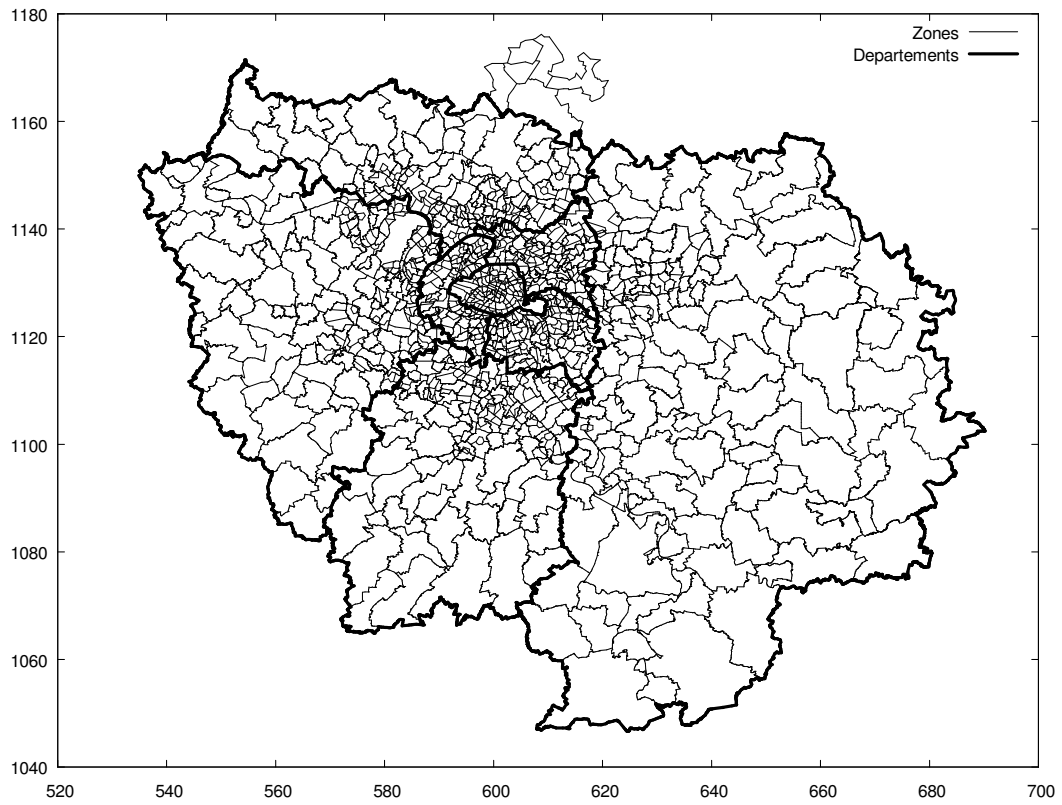
3.   APPLICATION TO THE PARIS METROPOLITAN AREA

This section reports on an application of the LADTA Tool Kit to the road network of the Paris metropolitan area. The simulation inputs are introduced in § 3.1, including a dynamic OD trip table. Some assignment results are depicted and commented on in § 3.2. Lastly, computational aspects including the distribution scheme and the run time per iteration are presented in § 3.3.
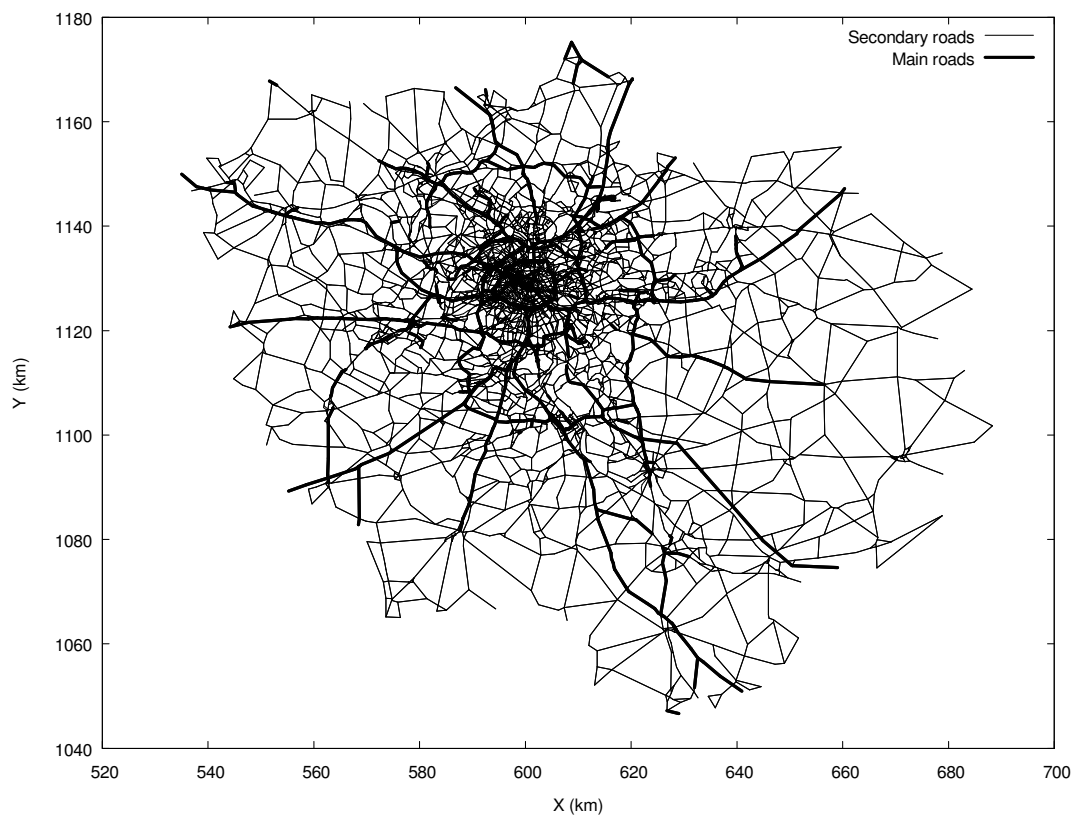
3.1   Application data for the Paris case

The administrative region that covers Paris metropolitan area is called "Région Ile de France". Its geographic extent is around 140 km from west to east, and 100 km from south to north. It is divided into eight counties called "départements", as illustrated in bold line on Figure 3(a); the City of Paris is the central department. The zoning system as provided by the State Department of Transport for Paris area (Dreif) covers the eight counties in the Paris metropolitan area plus some extra area in the northern part. It comprises 1,277 zones and its level of detail is varied with the density of population. The Dreif supplied us with a geocoded file of the Paris metropolitan area

road network. Figure 3(b) depicts this network. It comprises 39,137 directed arcs and 18,048 nodes. Roads having a capacity higher 2,500 pcu/h are drawn with thick lines.

(a)

Large size problems of dynamic network assignment and traffic equilibrium

(b)

FIGURE 3 : (a) Traffic zones and administrative departments in Paris area (b) Paris metropolitan road network.

For the purpose of our experiment, we had to set up a dynamic OD trip table. To this end we used two data sources:

- an OD trip table containing the averaged flow $Q_{od}|_{5\,\text{pm}}^{7\,\text{pm}}$ between each OD pair of zones (o,d) during the evening peak hour (from 5 pm to 7 pm), in pcu/h.

- a household travel survey made over the Paris metropolitan area in 2001 and called the EGT. The EGT contains data on about 80,000 individual trips, 36,000 out of them being recorded as "made by car". The departure time of each surveyed trip is known, as well as its statistical weight in the set of trips made during an average weekday.

We derived a dynamic OD matrix, aggregated by county to county pair and by periods of 30 minutes. Precisely, a demand volume profile $Q_{o,d}(h)$ was evaluated by inter-zone OD pair (o,d), knowing:

- the county O to which zone o belongs (resp. D, d),

- the trip flow from o to d, during the evening peak period, $Q_{od}|_{5\,\text{pm}}^{7\,\text{pm}}$,.

- the profile of trip volume by county to county OD pair (O,D), $Q_{OD}(h)$, hence the volume from county to county D at the evening peak, $Q_{OD}|_{5\text{pm}}^{7\,\text{pm}}$.

Based on these inputs, the temporal profile by elemental OD pair (o,d) was inferred as follows:

$$Q_{od}(h) = \frac{Q_{od}|_{5\,\text{pm}}^{7\,\text{pm}}}{Q_{OD}|_{5\,\text{pm}}^{7\,\text{pm}}} Q_{OD}(h)$$

3.2   Assignment results

The results after 50 iterations were taken as sufficiently close to equilibrium. It is quite hard to summarize the outcomes of an all day long dynamic assignment. Certainly the most evocative outcome would be a video of the instantaneous traffic state on a network map, but on such a large size problem the film would have to be focused on some sub-network (either by location or by road type) to achieve legibility. Before providing a network map, let us comment on some aggregate indicators and their variations over the day of assignment.

Figure 4 depicts the number of queued vehicles, in the whole network and during the day. It starts to increase shortly after 6:00 am. It then exceeds 100,000 veh during the morning peak, between 7:30 am and 9:00 am. From 10:00am to 4:00pm it varies between 10,000veh and 40,000 veh. The evening peak starts at 4:00pm. Between 4:00pm and 5:00pm, it grows rapidly up to 170,000veh, and remains around this level until 8:00pm. Three distinct peaks can be observed between 4:00pm and 8:00pm, after what the number of queued vehicles decreases to reach a few thousands after 9:00pm.

These variations are consistent with those of the Instantaneous Mean Speed (IMS) computed along the main roads (i.e. roads having a capacity higher 2,500 pcu/h), which is plotted in Figure 5. In order to ease comparison with real world data, the background of the plot in Figure 5 is taken from the Sytadin real time traffic information system. Sytadin provides a mean speed indicator, updated every 6 minutes, on the Paris area main roads network. The dark curve in the background plot shows the evolution of Sytadin's mean speed indicator between 5:00am and 12:00am, November 14th, 2008. The red area in the background plot represents values that are exceptionally taken by this indicator, while the yellow area can be interpreted as its range of variations under usual traffic conditions. Those information are publicly available at http://www.sytadin.tm.fr.

The IMS is computed as follows:

$$IMS(h) = \frac{\sum_{a \in A'(h)} L_a V_a(h)}{\sum_{a \in A'(h)} L_a}$$

with :

- $A'(h)$ the set of arcs having a capacity greater than 2500veh/h and a strictly positive input flow at instant h.

- $V_a(h) = \dfrac{L_a}{T_a(h)}$

- L_a is the length of arc a.

The maximum value of the IMS, which is around 90 km/h, is attained at 5:00am. From then, the average speed decreases down to 55 km/h just before 8:00am. It next increases until 10:00am, and then slowly decreases while remaining above 80 km/h until the evening peak arises. It appears like the evening peak lasts longer than the morning peak.

Interestingly, the variations of the IMS compare favourably with those of the mean speed indicator provided by Sytadin.
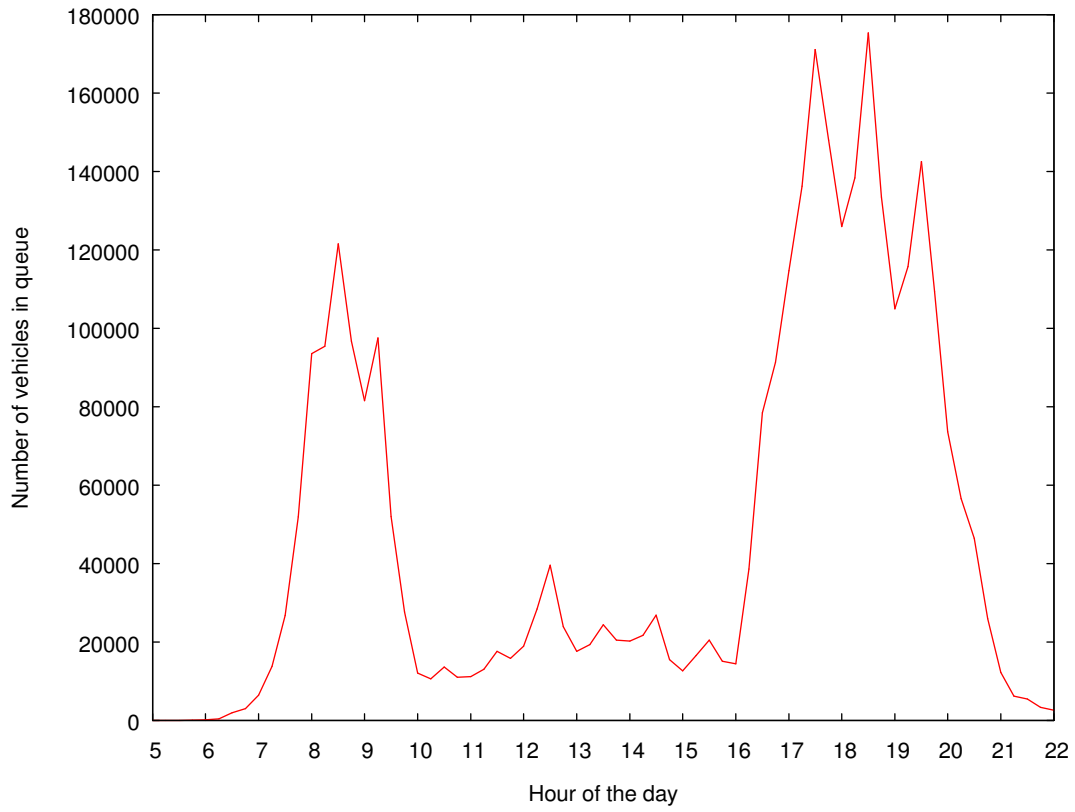
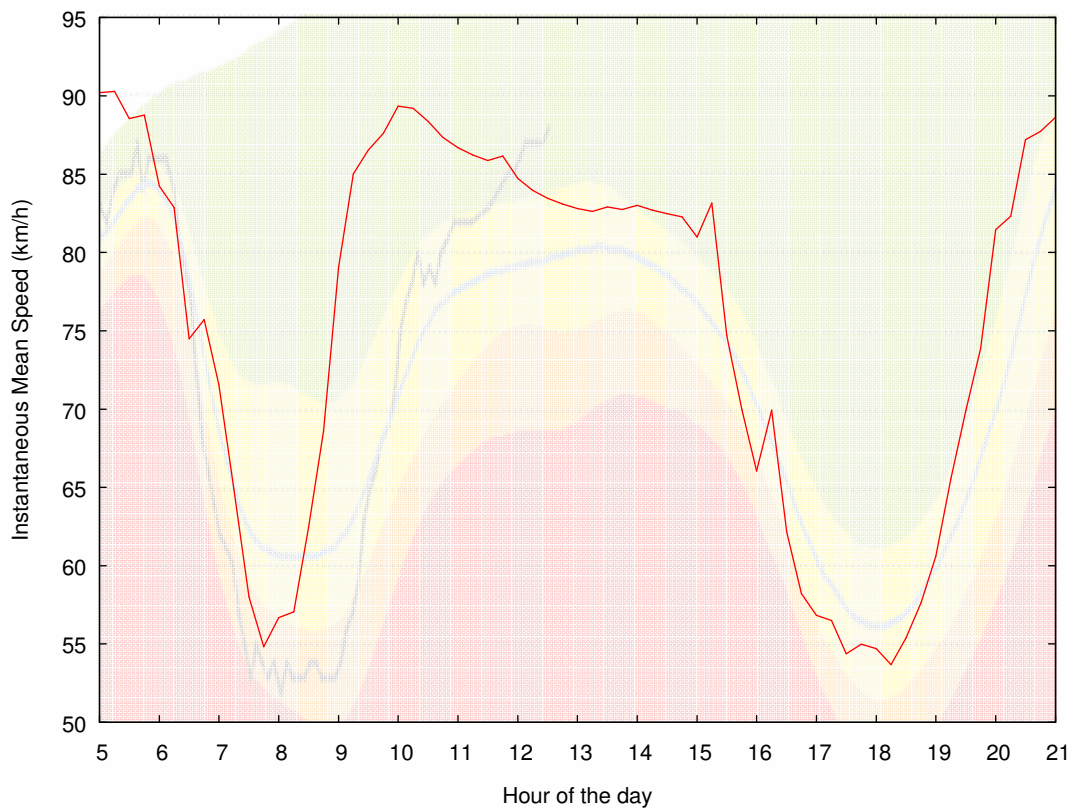**FIGURE 4: Number of queued vehicles throughout the Paris network**

FIGURE 5: Instantaneous Mean Speed on main roads, compared to measured data.

Large size problems of dynamic network assignment and traffic equilibrium

Let us finally come to the network map of congestion: out of many link indicators in terms of vehicle number, number of queued vehicles, proportion of queued vehicles, mean speed, ratio of mean speed to free-flow speed and so on, our preferred indicator is the total duration of the congestion period across the day. This is depicted in Figure 6 by using two thresholds of, respectively, 1 and 4 hours per day. Almost all of the motorways are queued longer than one hour per day. Links queued longer than four hours per day include (a) the main north-south arterials in the Centre of Paris; (b) the first Paris ring road "Le Boulevard Périphérique"; (c) some parts of the second ring road "A 86"; (d) some parts of the third ring road "La Francilienne"; (e) some radial motorways running south; (f) an intermediary ring road used to access the major Business Centre at "La Défense".

These results must be taken as an under statement of traffic congestion in the Paris metropolitan area, because of our simplified model of congestion on the basis of point-wise queues: thus congestion is under estimated both in the unqueued state where medium-range flow has unimpaired speed and in the queued state where the capacity rate is kept at maximum value, notwithstanding the lack of queue backward propagation.
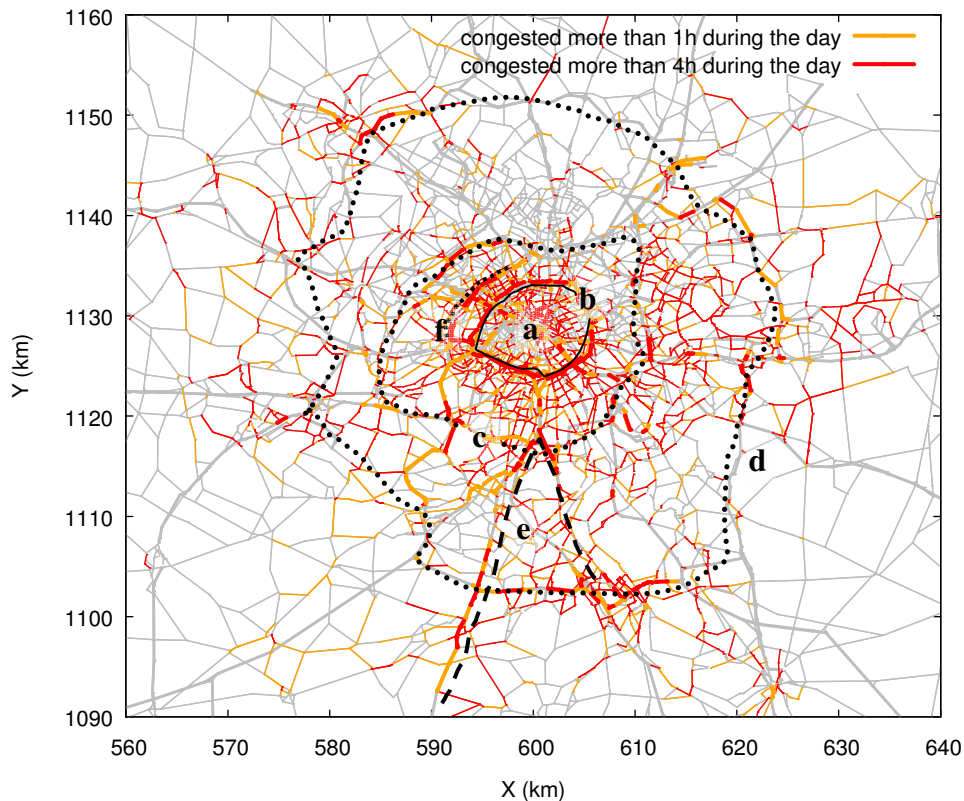

FIGURE 7 Location of severe traffic congestion along the Paris road network

3.3   Computation log book

The computation was distributed among one client computer and six servers, using the distribution facilities of the LTK, and following the resolution scheme exposed in §2.5, with arc toll price profiles set to 0, and arc traversal costs taken equal to arc traversal times. A total of 4 computers, each equipped with bi-core processors and 2Gb of RAM, were used. One computer ran the client application. The other three ran

two servers each. Each server handled about 214 destinations. A total of 50 iterations were performed, yielding a total run time of roughly 5h30.

3.3.1 Run time per iteration

The run time per iteration, as well as the time taken by each step within an iteration, were measured along 50 iterations. The time by iteration includes the time needed for data exchange between the client and the servers. The run time of the Formation Cost step (FC) is almost unnoticeable. Anyway, this gives a rough approximate of the cost of data exchange since in the FC step at iteration k, the client basically requests the server to get a local copy of the traversal time profiles computed in iteration k-1.

Not surprisingly, the two most time consuming steps are Route Choice (RC) and Volume Loading (VL). The run time of the Route Choice step (RC), during which the DLCP algorithm is performed, increases rapidly during the very first iterations, and then tends towards 2 minutes per iteration. Volume loading evolves similarly. The Traffic Flowing step (TF), done at the client level, is almost negligible during the first 10 iterations, then increases slowly to about 10 seconds per iteration.

These elements enable one to predict the effect of any number of processors.

3.3.2 Flow balance at nodes

Our computation scheme relaxes the constraint that, at every instant, the total flow coming in a given node should be equal to the total flow going out of the node. Indeed, at traffic equilibrium (if any) this property must hold at each node. So a necessary condition for convergence is to estimate the flow balance at nodes. Arbitrarily, we considered a node to be unbalanced if

$$\int_h \left| X_n^+(h) - X_n^-(h) \right| dh > \frac{1}{100} X_n^+(+\infty)$$

In the evolution of the number of unbalanced nodes with the iteration number along the assignment run, from iteration 2 at criterion value of 7% a continued decreasing is observed, at a quick and steady pace up to iterations 12-15 at criterion value of 2%, then with diminishing returns owing to the step size in the convex combination step.

CONCLUSION

The LADTA Tool Kit was applied to the Paris metropolitan road network for an equilibrium dynamic traffic assignment (DTA) in continuous time all day long. The computational performance was satisfactory, thus demonstrating the feasibility of equilibrium, macroscopic DTA on problems of very large size.

The application results showed that many major links are subject to queued traffic during a significant period in the day, which makes obvious the need to use dynamic rather than static assignment both to analyze and to manage the network. A side requirement pertains to assignment indicators in order to track queued traffic states and the effects of congestion at a scalable level of detail – including by origin-destination pair of traffic zones. Traffic planners at the Dreif are all the more interested in our model as its application data are mainly the same as their data for static assignment, apart from the dynamic OD matrix.

The computational approach handles time in a continuous way, with selected instants that separate affine pieces of the main traffic and travel variables. The assignment

software has a modular and multilayer architecture, which allows for distributed computation where a client computer asks servers for specific tasks. The software user is supplied with many commands and controls on the computation process at every layer in the architecture.

In the authors' opinion, these implementation principles are robust and will make easier the further developments in the LADTA model. In the current state of implementation, the dynamic features in LADTA include: (i) the route propagation of time, flow and travel cost; (ii) in the users' choice of route and/or departure time, a perfect information about travel times and costs as expected under equilibrium; (iii) dynamic operation of arc exit capacity and reference travel time; (iv) point-wise queuing at arc exit. On-going development is targeted to: (v) information and traffic orientation; (vi) congestion in the unqueued state of traffic; (vii) backward propagation of queues. Further model development will be aimed at: (viii) the location of parking capacity and its availability by time of day; (ix) the choice of travel mode.

REFERENCES

(1) Kuwahara M & Akamatsu T (1997) Decomposition of the reactive dynamic assignments with queues for a many-to-many origin-destination pattern. *Transportation Research B* **31**/1, 1-10.

(2) Kuwahara, M., Akamatsu, T., (2001). Dynamic user optimal assignment with physical queues for many-to-many OD pattern. *Transportation Research Part B* 35, 461–479.

(3) Wu, J.H., Chan,, Y. and Florian, M., 1998. The continuous dynamic network loading problem: a mathematical formulation and solution method. *Transportation Research B* 32, pp. 173–187.

(4) Bellei G, Gentile G and Papola N, A within-day dynamic traffic assignment model for urban road networks, *Transportation Research Part B* **39** (2005), pp. 1–29.

(5) Bellei G, Gentile G, Meschini L, Papola N: A demand model with departure time choice for within-day dynamic traffic assignment. *European Journal of Operational Research* 175(3): 1557-1576 (2006)

(6) Leurent F (2003) On network assignment and supply-demand equilibrium: an analysis framework and a simple dynamic model. *Paper presented at the 2003 European Transport Conference* (CD Rom edition).

(7) Leurent F (2004) The multiclass flowing problem in a dynamic traffic assignment model. *Paper presented at the 2004 European Transport Conference* (CD Rom edition).

(8) Leurent F., Aguiléra V., Mai H.D. (2007) Convergence Criteria and Computational Efficiency of a Dynamic Traffic Assignment Model. Proceedings of the World Conference on Transport Research (WCTR) 2007, first CD Rom edition, Berkeley, juin 2007, 22p.

(9) Peeta S & Ziliaskopoulos A (2001) *Foundations of dynamic traffic assignment: The past, the present and the future*. Network and Spatial Economics **1** (3/4), 233-266.

(10) http://www.caliper.co.il/transcad.htm

(11) http://www.inro.ca/en/products/dynameq/index.php

(12) http://www.omnitrans-international.com/pages/software/packs_traffic_dynamic.htm

(13) Sheffi Y (1984) *Urban Transportation Networks.* Prentice Hall, Englewood Cliffs, New Jersey.

(14) Beckmann MJ, McGuire CB & Winsten CB (1956) *Studies in the Economics of Transportation*. New Haven: Yale University Press.

(15) Minoux, M (1976) Structures algébriques généralisées des problèmes de cheminement dans les graphes : théorèmes, algorithmes et applications. *RAIRO revue verte*, Vol. 10 (6), pp. 33-62.

(16) Dial R (1969) Algorithm 360: shortest path forest with topological ordering. *Communications of ACM* Vol. 12, pp. 632-633.