OPERATIONS RESEARCH
Vol. 52, No. 2, March–April 2004, pp. 191–212
ISSN 0030-364X | EISSN 1526-5463 | 04 | 5202 | 0191

DOI 10.1287/opre.1030.0091

A Strategic Flow Model of Traffic Assignment in Static Capacitated Networks

Patrice Marcotte, Sang Nguyen, Alexandre Schoeb
DIRO and CRT, Université de Montréal, CP 6128, succursale Centre-Ville, Montréal, Québec, Canada H3C 3J7
{marcotte@iro.umontreal.ca, nguyens@iro.umontreal.ca, schoeb@crt.umontreal.ca}

This work pleads for the use of the concept of strategies, and their network-theoretic representation as hyperpaths, for modeling network assignment problems. While this concept describes adequately the behavior of users in transit systems, we show that it can apply as well to networks where arc capacities are rigid. This opens up a whole new field of research and raises several questions, from both the theoretical and computational points of view. These are investigated in the paper.

Subject classifications: equilibrium; traffic assignment; hyperpath; capacities; variational inequalities.
Area of review: Transportation.
History: Received February 2000; revisions received July 2001, July 2002; accepted March 2003.

1. Introduction

The traffic assignment problem consists in determining data or passenger flows that are compatible with the users' travel demand and routing behavior. Mathematical models of traffic assignment are usually based upon Wardrop's principle (Wardrop 1952), which states that at equilibrium, flows are assigned to shortest paths with respect to current (flow-dependent) travel delays. A variety of such models have been considered in the literature, incorporating such features as transportation modes, variable demand, user classes, stochasticity of travel delay, and more recently, the dynamic aspects of congestion. However, one feature of transportation networks that is frequently overlooked is the finiteness of arc capacities, which might prevent some users from traveling on their preselected path.

Our modeling solution to this problem is to assume that the user behavior is dictated by *strategies* that provide, at each node, alternative subpaths (from the current node to the destination) whenever the preferred outgoing arc is saturated. This concept of strategy (or "hyperpath") has been applied in the context of transit assignment by Nguyen and Pallottino (1989) and in the context of networks with stochastic, time-varying delays by Miller-Hooks (2001). However, in contrast with both these applications, we assume that the state of the system, at each node of the network, is flow dependent, and it follows that our model calls for entirely different analytical and algorithmical tools. The contribution of this work is fivefold:

- a variational inequality characterization of strategic equilibria;
- an analysis of the theoretical properties of the model;
- algorithms for deriving arc flows from the vectors of strategies;
- algorithms for computing the best response to a given strategic flow assignment;
- numerical implementation of algorithms for finding equilibrium solutions.

2. Motivation and Structure of the Paper

A simple way of preventing traffic flow from exceeding the physical capacity of a transportation network is to model delay through functions that become unbounded as flow approaches the capacity. A related approach, pursued by Hearn (1980), Larsson and Patriksson (1998), incorporates the capacity constraints within the model and sets the delay on an arc to the sum of the actual delay and the dual variable associated with the capacity constraint of that arc. From the modeling point of view, the dual variables can be interpreted as queueing delays occurring at the entrance of capacitated arcs (see Daganzo 1998).

A simple example will help to understand the differences between our strategic approach and the above "path approach." Consider a traffic assignment problem involving a single origin-destination couple and where demand is equal to one flow unit. Let us denote by P the set of origin-destination paths and by

$$x = \{x_p\}_{p\in P} \in X = \left\{ x\colon \sum_{p\in P} x_p = 1,\ x \geqslant 0 \right\},$$

a feasible flow pattern. Each path p is endowed with a capacity u_p, which corresponds to the minimum capacity of any arc in p. Let $F_p(x)$ be the travel time function along path p and let $F(x) = \{F_p(x)\}_{p\in P}$. If the capacities u_p are infinite, Wardrop's first principle states that an equilibrium is reached when travel costs are equal on all used paths,

and this common cost λ is less than the actual cost on any unused path. An equilibrium flow x is then a solution of the nonlinear complementarity problem:

$$x \in X,$$
$$F_p(x) - \lambda \geqslant 0,$$
$$x_p[F_p(x) - \lambda] = 0,$$

or, equivalently, of the variational inequality

$$x \in X,$$
$$\langle F(x), x - y \rangle \leqslant 0 \quad \forall y \in X.$$

Now, in the capacitated case, it might be required to include a queueing delay α_p to a saturated arc in order to achieve the above equilibrium conditions. This yields the complementarity system:

$$x \in X \cap \{x \leqslant u\},$$
$$F_p(x) + \alpha_p - \lambda \geqslant 0,$$
$$x_p[F_p(x) + \alpha_p - \lambda] = 0,$$
$$\alpha_p[x_p - u_p] = 0,$$
$$\alpha_p \geqslant 0,$$

which is equivalent to the variational inequality

$$x \in X \cap \{x \leqslant u\},$$
$$\langle F(x), x - y \rangle \geqslant 0 \quad \forall y \in X \cap \{x \leqslant u\}.$$

If the mapping F is the gradient of some potential function f, x is a solution of the previous variational inequality if and only if it is a first-order stationary point for the mathematical program

$$\min_{x \in X \cap \{x \leqslant u\}} f(x).$$

An advantage of this approach is that the above mathematical program can be solved quite easily. However, the use of a queueing delay resulting from transient overcapacity of the network is controversial, although its validity has been advocated by Daganzo (1998), using the concept of "point queues."

Our approach to capacities is entirely different. We assume that the users' behavior is dictated by travel strategies rather than path selection. More precisely, a strategy assigns to each node of the network an ordered set of successor nodes. At each node, a user selects the first element in his preference set whose associated outgoing arc is unsaturated; feasible strategies take into account the possibility that an arc be unavailable when the user reaches the tail node of that arc, and must consequently include alternatives. While strategies are deterministic, their realizations depend on arc availability, and are therefore stochastic and flow dependent. Users state their preferences but cannot be assured, from day to day, of the actual path they will travel.

The strategic approach has been applied successfully by Nguyen and Pallottino (1988) or Spiess and Florian (1989) to the modeling of urban transit networks. In this application users select, at a given transit node, a subset of the available lines and board the first incoming vehicle from a line belonging to that subset. In contrast with our definition, the sets of attractive lines in the transit application are unordered, and therefore ill adapted at modeling situations where users may incorporate into their strategies time-dependent information, such as public transportation schedules or online information to private vehicles. The price to pay for this added flexibility is an increase in theoretical and algorithmical complexity.

While our ultimate aim is the study of priority networks, where the use of strategies as fundamental decision variables settles some paradoxical situations that arise when only path flow variables are considered, this work focuses on the analysis and implementation of the nonpriority model introduced by Marcotte and Nguyen (1998). It is organized as follows. In §3 we give a detailed description of a simple instance where no equilibrium expressed solely in terms of path flow variables exist, thus justifying our notion of equilibrium in capacitated networks. In §§4 and 5 we analyze algorithms for obtaining path and arc flows from strategies, as well as for determining strategies that are optimal with respect to a given assignment. In §6 we formulate and analyze an equilibrium model with constant arc costs, i.e., where the sole congestion effects are induced by the arc capacities. In §7 we present algorithms for determining an equilibrium. The implementation of specific algorithms within this framework is discussed in §8, which also provides numerical results.

3. Strategic Equilibria in Capacitated Networks

Defining equilibrium meaningfully in capacitated transportation networks represents a nontrivial task. To gain some insight into this matter, let us consider the network illustrated in Figure 1, where each arc is endowed with a

Figure 1. A capacitated network.

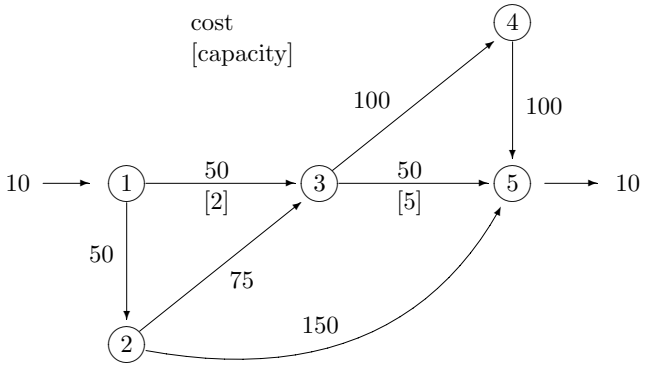


Table 1. Network paths for a small example.

Path Index	Path	Cost	Capacity
1	1–3–5	100	2
2	1–2–3–5	175	5
3	1–2–5	200	∞
4	1–3–4–5	250	2
5	1–2–3–4–5	325	∞

cost (shown next to the corresponding arc) and, possibly, a capacity (bracketed number). The paths from origin node 1 to destination node 5 are listed, together with their features, in Table 1.

Clearly, no Wardrop equilibrium is compatible with these data, because the shortest path 1–3–5 cannot accommodate all demand of 10 units. Quite naturally, one could settle for an equilibrium satisfying the following extended Wardrop principle:

> At equilibrium, the cost of a path with a positive residual capacity is larger or equal to the cost of any path carrying positive flow.

In our small example, any path flow vector of the form

$$x = (5 - t, t, 5, 0, 0)$$

with $t \in [3, 5]$ satisfies the above principle. However, whenever t is strictly greater than 3 and the users of the second path are endowed with the gift of prescience, it is tempting for them to switch to the less costly first path on which some capacity has been set free by *themselves*. In this sense, the equilibrium associated with the value $t = 3$ is more natural. Indeed this solution, associated with a queueing delay of 75 on arc $(1, 3)$ and 25 on arc $(3, 5)$, corresponds to a "Hearn-Larsson-Patriksson" equilibrium and is, notwithstanding the queueing delay, a system-optimal solution as well.

Let us now consider the strategic approach where a subset of strategies, represented as vectors whose elements consist of an ordered list of outgoing nodes, is given in Table 2. Some strategies, such as s_4, avoid capacitated arcs and correspond to ordinary paths. In contrast, a user adopting strategy s_1 could end up traveling on path 1, 2, 4, or 5, depending on the availability of the capacitated arcs $(1, 3)$ and $(3, 5)$.

Table 2. A set of strategies for the small example.

Node	1	2	3	4	5
s_1	[3, 2]	[3]	[5, 4]	[5]	[ ]
s_2	[3, 2]	[5]	[5, 4]	[5]	[ ]
s_3	[2]	[3]	[5, 4]	[5]	[ ]
s_4	[2]	[5]	[ ]	[ ]	[ ]
s_5	[3, 2]	[5, 3]	[5, 4]	[5]	[ ]
s_6	[3]	[ ]	[5, 4]	[5]	[ ]
s_7	[3, 2]	[3]	[4, 5]	[5]	[ ]

Table 3. Path access probabilities for the small example.

Path	Access Probability	Cost
1–3–5	$(2/10) \times (5/10) = 1/10$	100
1–2–3–5	$(8/10) \times (5/10) = 4/10$	175
1–2–5	0	200
1–3–4–5	$(2/10) \times (5/10) = 1/10$	250
1–2–3–4–5	$(8/10) \times (5/10) = 4/10$	325

Whenever the number v_a of users who want to access arc a exceeds its capacity u_a, we assume that the probability p of accessing the arc is equal to $p = u_a/v_a$. This is equivalent to assuming that the users are independently and uniformly distributed at the tail node of arc a. These *access probabilities* (or *access proportions*) allow us to compute the expected cost of strategies. For instance, if all 10 users adopt strategy s_1, the probability of accessing arc $(1, 3)$ is equal to $2/10$. The users, whether they access arc $(1, 3)$ or not, clash again at node 3, where the access probability of arc $(3, 5)$ is $5/10$. These numbers, which are required to compute the access probabilities associated with the paths of the network, are shown in Table 3.

The expected value of each user's *expected* delay is equal to the sum of the path costs, weighted by the respective path access probabilities, i.e.,

$$\left(\frac{1}{10} \times 100\right) + \left(\frac{4}{10} \times 175\right) + \left(\frac{0}{10} \times 200\right) + \left(\frac{1}{10} \times 250\right) + \left(\frac{4}{10} \times 325\right) = 235.$$

Access probabilities could also have been associated with the paths of the network; the reader is referred to the work of Marcotte and Nguyen (1998) for specific details, as well as a discussion of the relationship between strategies and hyperpaths defined on a suitably defined hypergraph.

A *strategic equilibrium* is reached when all users are assigned to strategies of minimal expected delays. This condition is violated if all commuters are assigned to strategy s_1, because strategy s_4, which corresponds to the uncapacitated path 1–2–5, is available at cost $200 < 235$. It is not too difficult to check that the unique equilibrium corresponds to the assignment of five users to strategy s_1 and of the remaining five to strategy s_2; the resulting strategic flows and path flows are shown in Table 4. One verifies that the expected delay of each strategy is equal to 185, which

Table 4. Equilibrium strategic path flows.

Path	Flow from s_1	Flow from s_2	Cost
1–3–5	5/6	5/6	100
1–2–3–5	20/6	0	175
1–2–5	0	4	200
1–3–4–5	1/6	1/6	250
1–2–3–4–5	4/6	0	325

is less than the expected delay of unused strategies, thus fulfilling the equilibrium conditions. In general, a vector x of *strategic flows* is a strategic equilibrium if and only if it is demand-feasible and satisfies the variational inequality

$$\langle C(x), x - y \rangle \leqslant 0 \tag{1}$$

for every demand-feasible strategic vector y, where the component $C^s(x)$ of $C(x)$ is the expected delay associated with strategy s and the strategic vector x. Unfortunately, the calculation of C requires the knowledge of total arc flow, which is not available in closed form, even for small networks. In standard traffic models, this information is readily available from path flows (via the arc-path incidence matrix), but in the strategic model, the dependence of the arc-hyperpath matrix on strategies complicates matters significantly. Indeed, the derivation of arc flows from strategies requires a specific algorithm which will be described in detail in §4.

We close this section with a list of the key mathematical symbols. Slightly abusing notation, we write "cost" and "flow" in place of "expected cost" and "expected flow," respectively. Other notations will be introduced as required.

$G = (N, A)$ network with node set N and arc set A
$j^+ = \{k : (j,k) \in A\}$ forward star of node j
d_{qr} demand from origin node q to destination node r
u_{jk} capacity of arc (j,k)
$\bar{u}_{jk}$ residual capacity of arc (j,k)
S set of strategies
$q(s)$ origin node of strategy $s \in S$
E_j^s ordered list of nodes in the forward star j^+
$s = \{E_j^s\}_{j \in N}$ strategy
x^s strategic flow
$x = \{x^s\}_{s \in S}$ vector of strategic flows
z_j^s flow from strategy s having reached node j
τ_j^s probability of accessing node j using strategy s
$\pi_{jk}^s(x)$ probability of accessing node k from node j using strategy s
$C^s(x)$ cost of strategy s
$C(x) = \{C^s(x)\}_{s \in S}$ vector of strategic costs
C_j^s cost of strategy s from node j to the destination
W working set (set of strategies considered at a given iteration).

4. The Loading Mechanism

In this section, we describe a procedure for recovering arc flows from strategic flows. This loading procedure, which is required to produce the costs of paths and strategies, is akin to a deterministic simulation. Throughout the paper we make the assumption that this operation is feasible, i.e., there exist arc flows that are compatible with the current strategies and capacities. This condition can be enforced by introducing an artificial arc with arbitrarily high cost (a "walk arc") between every two nodes of the network.

In the small example considered in §3, the preference sets of all users were identical at all tail nodes of capacitated arcs. This resulted in an intuitive and simple loading process based on the ratio between capacity and flow. The generalization of this process to more complex situations, however, can lead to distinct arc flows, depending on the queue discipline implemented, i.e., the way one manages the simultaneous assignment of flows to the capacitated arcs of the network. We consider two such disciplines (while the analogy is made with queues involving discrete customers, the loading procedure acts on a continuum of vehicles):

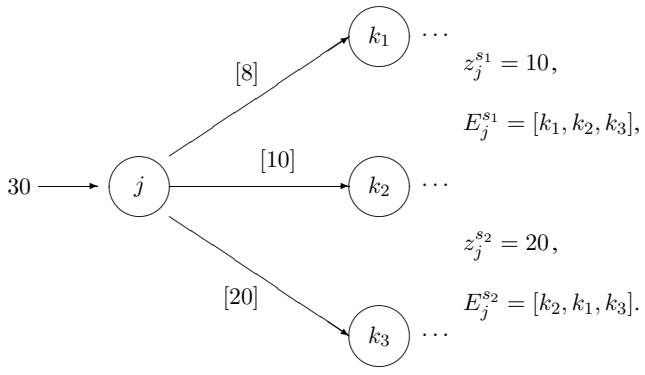
single queue processing (SQP)—Users are randomly and uniformly distributed in a single vertical queue (storage space is unlimited) and no preemption is allowed, i.e., the first-in-first-out (FIFO) condition is enforced;

parallel queue processing (PQP)—Users are randomly and uniformly distributed in a single vertical queue. A user is allowed to access an arc if its residual capacity is positive and if no user with the same current *first* choice stands in front of him in the queue. In other words, one may preempt another user who expects to access a higher priority arc (with positive residual capacity) within his own strategy. However, the preempted user retains its "priority number" when competing for an outgoing arc. This rule implicitly assumes that the users' behavior is dictated by the "status" of a capacitated arc (saturated or unsaturated) and not by its residual capacity. Equivalently, one could say that users are unaware of the number of competitors lying ahead in the queue.

The discipline PQP allows the representation of bottlenecks that originate at *head nodes* of outgoing arcs. Single queue processing, in contrast, is better adapted at modeling situations where FIFO holds and possesses better theoretical properties. For both these reasons, it has been adopted in our study.

Consider the subnetwork illustrated in Figure 2. At the first iteration of the loading process, the strategic flows x^{s_1} and x^{s_2} are assigned to their preferred node at a rate proportional to their respective size. Because the ratio $10/20$ is less than the ratio $8/10$, the arc (j, k_2) gets saturated

Figure 2. The loading process at node j.



first, after half the flow (10 units) of strategy s_2 has been assigned to it. At that stage, and in accordance with the definition of SQP, half the flow (five units) of strategy s_1 must have reached its preferred node k_1 as well. Next, node k_2 is removed from the preference sets, the arc capacities are replaced by their residual capacities and the process is repeated. At the second iteration, 5 units of flow from s_1 and 10 from s_2 compete for the residual capacity (3) of arc (j, k_1). According to our proportionality assumption, one flow unit from s_1 reaches node k_1 while two flow units from s_2 reach k_1. A third iteration is required to load the unassigned flow to the uncapacitated arc (j, k_3). The arc access probabilities are then obtained by computing, a posteriori, the ratio of the successful flow over the corresponding arc capacity. These calculations are summarized in Table 5.

Although the loading procedure is based on intuitive concepts, its generalization to complex topologies is not straightforward. Actually, for the process to be well defined, there must exist a topological ordering of the network nodes. The loading is then performed in accordance with this topological order. Because the incoming flows at a node only depend on the flows originating from nodes located at lower levels of any Hasse diagram of the network, the procedure is independent of the topological ordering adopted, as it should be.

If the application considered admits an embedded acyclic network—for example, in the case of a time-space expanded network—then a natural topological ordering is immediately at hand; otherwise, a network preprocessing phase must precede the loading procedure. For any given destination, it is reasonable to assume that utilized paths do not include cycle. This is akin to applying Dial's concept of "efficient" path (see Dial 1971) from each origin to a given destination. A path is efficient if every arc in it has its head node closer to the destination than its tail node. For the considered destination, the working subnetwork that includes only efficient arcs is acyclic. The loading procedure can then be applied to the subnetwork, one destination at a time. This implies further that the final solution algorithm can resort to a decomposition scheme by destination.

Table 5. Single queue loading at node j.

Iteration	Arc	(j, k_1)	(j, k_2)	(j, k_3)
1	Residual capacity	8	10	20
	Flow	$5(s_1)$	$10(s_2)$	0
2	Residual capacity	3	0	20
	Flow	$5(s_1)+1(s_1)+2(s_2)$	$10(s_2)$	0
3	Residual capacity	0	0	20
	Flow	$6(s_1)+2(s_2)$	$10(s_2)$	$4(s_1)+8(s_2)$

Access probabilities: $\pi_{jk_1}^{s_1} = 6/10$, $\pi_{jk_3}^{s_1} = 4/10$, $\pi_{jk_1}^{s_2} = 2/20$, $\pi_{jk_2}^{s_2} = 10/20$, and $\pi_{jk_3}^{s_2} = 8/20$.

Such schemes are frequently embedded in existing traffic assignment methods (see, for instance, Patriksson 1994). This issue is discussed shortly in the appendix.

The loading procedure is performed on the current subset of strategies (the working set W), which is enriched at each iteration of the master algorithm (see §7), and would be rather straightforward were it not for strategies having zero flow. While, at first sight, these strategies might be discarded, they play a central role in algorithm CAPSHORT of §5, which computes the optimal user reaction to given network conditions.

It is clear that the access probabilities corresponding to zero-flow strategies cannot be computed from flow ratios of null flows. To bypass this problem, we base our calculations on the proportion ρ_j^s of strategic flow from strategy s that reached node j. At the origin node of strategy s, this proportion is set to 1 and the arc access probabilities π_{jk}^s are computed iteratively according to the formula

$$\pi_{jk}^s \leftarrow \pi_{jk}^s + \beta\rho_j^s,$$

where β is a local variable set to the access probability at node k. Next, the residual proportion is updated from $\rho_j^s \leftarrow \rho_j^s(1-\beta)$ and the process is halted whenever ρ_j^s vanishes. Mathematically, the same results would be achieved by increasing the values of null flows to a small number ϵ, and then letting ϵ tend to zero.

As shown in a previous example, the loading operation is a greedy process where the flows associated with the preferred outgoing node of each active strategy are processed until either (i) the flow from some strategy is entirely processed, or (ii) the residual capacity of some arc becomes null. In both cases, the sets $\overline{E}_j^s$ representing the preference orders, as well as the unprocessed strategic flows $\bar{z}_j^s$, are updated, and the process is repeated. In the SQP case, the loading process terminates simultaneously for all flows, i.e., when the proportion of residual flows processed is equal to 1 ($\beta = 1$) for all strategies. The situation is different in the PQP case, where some flow could be entirely processed at the first iteration, for instance if the associated preferred outgoing arc has infinite capacity.

The pseudocode performing the assignment of a strategic flow vector $x = \{x^s\}_{s\in W\subset S}$ under SQP is given below. In this pseudocode, the strategy set is partitioned into subsets having identical preferred outgoing node. Next, one sets β, the proportion of residual flow yet to be assigned, to the minimum ratio of residual demand over the corresponding arc capacity, whenever this minimum is less than one. Otherwise, β is set to one, i.e., all residual flows are assigned to their current preferred choices, and the algorithm terminates. Note that, generically, a single node is removed from the preference sets at a given major iteration of the procedure. However, the algorithm addresses the general situation where outgoing arcs could saturate simultaneously.

PROCEDURE CAPLOAD (x)
input: $x = \{x^s\}_{s\in W}$ [*strategic flow vector*]
output: $\pi = \{\pi^s_{jk}\}_{(j,k)\in A,\, s\in W}$ [*arc access probabilities*]
$C = \{C^s\}_{s\in W}$ [*strategic cost vectors*]

INITIALIZATION
for $s \in W$ **do**
  $C^s \leftarrow 0$ [*cost of strategy s*]
  $z^s_{q(s)} \leftarrow x_s$ [*flow at origin node $q(s)$*]
  $\tau^s_{q(s)} \leftarrow 1$ [*probability of accessing origin node $q(s)$*]
  for $j \in N(j \neq q(s))$ **do**
    $z^s_j \leftarrow 0$ [*strategic flow at node j*]
    $\tau^s_j \leftarrow 0$ [*probability of accessing node j under strategy s*]
    for $k \in E_{js}$ **do**
      $\bar{v}^s_{jk} \leftarrow 0$ [*strategic flow on arc (j,k)*]
      $\pi^s_{jk} \leftarrow 0$ [*conditional probability of accessing arc (j,k) under strategy s*]
    endfor
  endfor
endfor

ASSIGNMENT PHASE
for $j \in N$ (according to topological order) **do**
  $\overline{W} \leftarrow \{s \in W : E_{js} \neq \varnothing\}$ [*set of active strategies*]
  for $s \in \overline{W}$ **do**
    $\bar{E}^s_j \leftarrow E_{js}$ [*updated preference order $\bar{E}^s_j$*]
    $\bar{z}^s_j \leftarrow z^s_j$ [*residual strategic flow*]
    $\rho^s_j \leftarrow 1$ [*residual probability*]
  endfor
  for $k \in j^+$ **do**
    $\bar{u}_{jk} \leftarrow u_{jk}$ [*residual capacity*]
  endfor
  while $\overline{W} \neq \varnothing$ **do**
    $K \leftarrow \varnothing$ [*set of residual first choices*]
    for $k \in j^+$ **do**
      $d_k \leftarrow 0$ [*demand for node k*]
      $S_k \leftarrow \varnothing$ [*set of strategies having node k as first residual choice*]
    endfor
    for $s \in \overline{W}$ **do**
      $k \leftarrow \bar{E}^s_j(1)$ [*first residual choice of strategy s*]
      $K \leftarrow K \cup \{k\}$ [*construction of set K*]
      $S_k \leftarrow S_k \cup \{s\}$ [*construction of S_k*]
      $d_k \leftarrow d_k + \bar{z}^s_j$ [*construction of d_k, current demand for node k*]
    endfor
    $\beta \leftarrow \min\{1, \min\{\bar{u}_{jk}/d_k\}\}$ [*proportion of residual flow assigned to current first choice*]
    for $k \in K$ **do**
      for $s \in S_k$ **do**
        $z^s_k \leftarrow z^s_k + \beta\bar{z}^s_j$ [*updating flow z^s_k*]
        $\bar{v}^s_{jk} \leftarrow \bar{v}^s_{jk} + \beta\bar{z}^s_j$ [*updating arc flow $\bar{v}^s_{jk}$*]
        $\bar{z}^s_j \leftarrow \bar{z}^s_j - \beta\bar{z}^s_j$ [*updating residual flow $\bar{z}^s_j$*]
        $\pi^s_{jk} \leftarrow \pi^s_{jk} + \beta\rho^s_j$ [*updating probability π^s_{jk}*]
        $\rho^s_j \leftarrow (1-\beta)\rho^s_j$ [*updating proportion ρ^s_j, yet to be processed*]
      endfor
      $\bar{u}_{jk} \leftarrow \bar{u}_{jk} - \beta d_k$ [*updating residual capacity $\bar{u}_{jk}$*]
      if $\bar{u}_{jk} = 0$ **then**
        for $s \in \overline{W}$ **do**
          $\bar{E}^s_j \leftarrow \bar{E}^s_j - \{k\}$ [*removing node k from $\bar{E}^s_j$*]
        endfor
      endif
    endfor
    if $\beta = 1$ **then**
      $\overline{W} \leftarrow \varnothing$ [*end of loading*]
    endif
  endwhile
  for $s \in W$ **do**
    for $k \in E_{js}$ **do**
      $C^s \leftarrow C^s + c_{jk}\tau^s_j\pi^s_{jk}$ [*update of C^s*]
      $\tau^s_k \leftarrow \tau^s_k + \tau^s_j\pi^s_{jk}$ [*update of probability τ^s_k*]
    endfor
  endfor
endfor

At each step within the **while** structure of the loading algorithm, at least one arc becomes saturated. Therefore, this loop is performed at most $|j^+|$ times for each node j, where j^+ is the forward star of node j. It follows that the loop is executed at most $\sum_{j\in N}|j^+| = |A|$ times and that the total running time of the algorithm is polynomial.

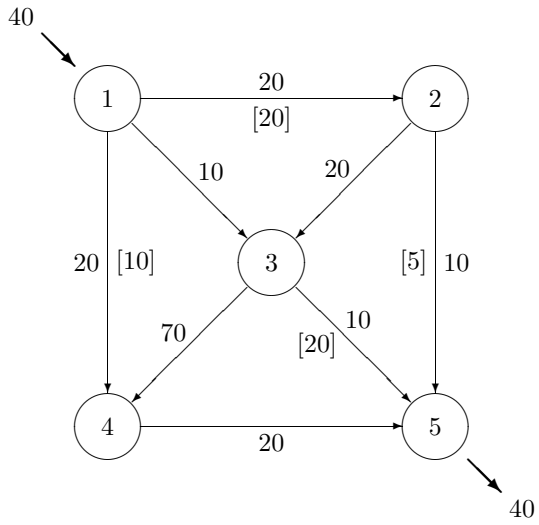
5. Computing a Best Strategy

Most algorithms for solving standard traffic assignment problems rely on a shortest path procedure. When dealing with strategic flows, things become significantly more complex because the cost function is not available in closed form. The challenge lies in pricing out strategies that are *not* elements of the working set W. Before stating the algorithm, we illustrate the procedure on the simple example of Figure 3, where a demand of 40 units has to be assigned from node 1 to node 5. Next to each arc is indicated its cost and, within square brackets, its capacity. Initially, the entire flow is assigned to the strategy

$$s_1 = ([2, 4, 3], [5, 3], [5, 4], [5], [\,]).$$

Under both queue disciplines, 20 units reach node 2 from node 1. Among the remaining 20 units, 10 units reach

Figure 3. Computing a best strategy: A simple example.



node 4, and the remaining 10 reach node 3. At node 2, 20 units are assigned to arcs $(2,3)$ and $(2,5)$ in respective proportions $3/4$ and $1/4$. Thus 25 (i.e., $10+15$) units of flow access node 3. Only one-fifth of these units cannot access arc $(3,5)$ directly and must transit through node 4. The flows reaching every node of the network are given in Table 6; the arc flows are given in Table 7 and the expected cost, obtained by averaging the total arc costs, is equal to

$$\frac{1}{40}(20\cdot 20+10\cdot 10+10\cdot 20+15\cdot 20+5\cdot 10+5\cdot 70 +20\cdot 10+15\cdot 20)=47\frac{1}{2}.$$

Now let us compute an optimal strategic answer $\bar{s}$ (not to be confused with an equilibrium solution) in the SQP case. Starting from the destination node, we set $C_5^{\bar{s}}=0$ and $E_5^{\bar{s}}=[\,]$. At node 4, the optimal preference order is trivially $E_4^{\bar{s}}=[5]$, and $C_4^{\bar{s}}=20$. At node 3, the choice of $(3,5)$ as the preferred arc is optimal and $E_3^{\bar{s}}=[5,4]$. The loading of strategy $\bar{s}$ at node 3 yields the probabilities $\pi_{35}^{\bar{s}}=4/5$ and $\pi_{34}^{\bar{s}}=1/5$. Working backwards at node 3, we obtain

$$C_3^{\bar{s}}=\frac{4}{5}(10+C_5^{\bar{s}})+\frac{1}{5}(70+C_4^{\bar{s}})=26.$$

The optimal preference order at node 2 is clearly $E_2^{\bar{s}}=[5,3]$. The access probabilities, already computed for strategy s_1, are $\pi_{25}^{\bar{s}}=1/4$ and $\pi_{23}^{\bar{s}}=3/4$, respectively. We obtain

$$C_2^{\bar{s}}=\frac{1}{4}(10+C_5^{\bar{s}})+\frac{3}{4}(20+C_3^{\bar{s}})=37.$$

Table 6. Node flows corresponding to the loading of strategy s_1.

Node	1	2	3	4	5
Flow	40	20	25	15	40

Table 7. Arc flows corresponding to the loading of strategy s_1.

Arc	$(1,2)$	$(1,3)$	$(1,4)$	$(2,3)$	$(2,5)$	$(3,4)$	$(3,5)$	$(4,5)$
Flow	20	10	10	15	5	5	20	15

The process terminates at the origin node, where the preference order is dependent on the value of the strategic flow x_{s_1}, and the optimal ordering $E_1^{\bar{s}}=[3]$ minimizes the expected cost

$$C_1^s=\pi_{12}^s(20+C_2^{\bar{s}})+\pi_{13}^s(10+C_3^{\bar{s}})+\pi_{14}^s(20+C_4^{\bar{s}}) =57\pi_{12}^s+36\pi_{13}^s+40\pi_{14}^s.$$

Indeed, because arc $(1,3)$ has infinite capacity, we have $\pi_{13}^{\bar{s}}=1$. The corresponding optimal strategy $\bar{s}=([3],[5,3],[5,4],[5],[\,])$, with expected cost 36, is displayed in Table 8, together with the cost-to-go at every node of the network.

Now, for a given strategic flow vector x inducing arc access probabilities $\pi_{jk}^s(x)$, the cost of an optimal reactive strategy $\bar{s}$, characterized by its preference sets $\{E_j^{\bar{s}}\}_{j\in N}$, is obtained by scanning the network in reverse topological order, starting at the destination node r:

$$C_j^{\bar{s}}=\begin{cases}\infty & \text{if } j>r,\\ 0 & \text{if } j=r,\\ \displaystyle\sum_{k\in E_j^{\bar{s}}}(c_{jk}+C_k^{\bar{s}})\pi_{jk}^{\bar{s}}(x) & \text{if } j<r.\end{cases} \tag{2}$$

Note that a similar recursion has been used by Nguyen and Pallottino (1989) to evaluate the cost of shortest hyperpaths in a transit assignment model, the crucial simplification being that in the transit case, access probabilities are *not* flow dependent.

A procedure that returns, for a given strategic flow and origin-destination pair (q,r), an optimal reactive strategy together with its expected cost, is listed below for the SQP queue discipline. This procedure does not assume that the "best" strategy is a member of the working set W, i.e., its flow could be zero, hence the importance of being able to load such strategies (MICRO-LOADING phase). The algorithm has two main components:

1. a sort operation for defining the preference order of the required strategy $\bar{s}$, at each node of the network whose topological index is less than the index of the destination node (the other nodes are irrelevant);

Table 8. The optimal strategy.

Node j	$E_j^{\bar{s}}$	$C_j^{\bar{s}}$
5	[ ]	0
4	[5]	20
3	[5, 4]	26
2	[5, 3]	37
1	[3]	36

2. a loading procedure that includes $\bar{s}$ in the working set. This procedure is similar to CAPLOAD, but dispenses with some bookkeeping operations that have already been performed by CAPLOAD, such as the computation of the access probabilities of strategies other than $\bar{s}$.

PROCEDURE CAPSHORT $(z, (q, r))$ (SQP rule)
input: $z = \{z_j^s\}_{s \in W,\, j \in N}$ [*strategic flow vector at the nodes*]
$(q, r) \in L$ [*O-D pair*]
output: $\bar{s}$, $C^{\bar{s}}$ [*optimal strategy and its cost*]

INITIALIZATION
for $j \in N$ **do**
  if $j = r$ **then** $C_j^{\bar{s}} \leftarrow 0$ [*cost of optimal strategy leaving node j*]
    else $C_j^{\bar{s}} \leftarrow \infty$
  endif
endfor

MICRO-LOADING (loading of a zero flow strategy)
for $j \in N$ $j < r$ (in reverse topological order) **do**
  $\overline{E}_j^s \leftarrow \text{sort}(j^+)$ [*sort in increasing order the set $\{c_{jk} + C_k^{\bar{s}}\}_{k \in j^+}$*]
  for $k \in \overline{E}_j^s$ **do**
    $\bar{u}_{jk} \leftarrow u_{jk}$ [*residual capacity of arc (j, k)*]
    $\pi_{jk}^{\bar{s}} \leftarrow 0$ [*conditional access probability to arc (j, k) under strategy $\bar{s}$*]
  endfor
  $\overline{W} \leftarrow \{s \in W : E_{js} \neq \varnothing\} \cup \{\bar{s}\}$ [*working set for O-D pair (q, r)*]
  for $s \in \overline{W}$ **do**
    $\overline{E}_j^s \leftarrow E_{js}$ [*residual preference order*]
    $\bar{z}_j^s \leftarrow z_j^s$ [*residual strategic flow*]
    $\rho_j^s \leftarrow 1$ [*proportion of residual strategic flow yet to be assigned*]
  endfor
  while $\rho_j^{\bar{s}} \neq 0$ **do**
    $K \leftarrow \varnothing$ [*set of residual first choices*]
    for $k \in j^+$ **do**
      $d_k \leftarrow 0$ [*demand for node k*]
      $S_k \leftarrow \varnothing$ [*set of strategies having node k as first residual choice*]
    endfor
    for $s \in \overline{W}$ **do**
      $k \leftarrow \overline{E}_j^s(1)$ [*first residual choice of strategy s*]
      $K \leftarrow K \cup \{k\}$ [*building the set of preferred outgoing nodes*]
      $S_k \leftarrow S_k \cup \{s\}$ [*building the set of strategies with preferred node k*]
      $d_k \leftarrow d_k + \bar{z}_j^s$ [*construction of d_k, current demand for node k*]
    endfor
    $\nu \leftarrow \max\{d_k / \bar{u}_{jk} : k \in K\}$
    $\beta \leftarrow \min\{1/\nu, 1\}$ [*proportion of assigned residual flow*]
    for $k \in K$ **do**
      for $s \in S_k$ **do**
        if $s = \bar{s}$ **then**
          $\pi_{jk}^s \leftarrow \pi_{jk}^s + \beta \rho_j^s$ [*updating access probability $\pi_{jk}^{\bar{s}}$*]
          $\rho_j^s \leftarrow (1 - \beta)\rho_j^s$ [*updating proportion $\rho_j^{\bar{s}}$*]
        else
          $\bar{z}_j^s \leftarrow \bar{z}_j^s - \beta \bar{z}_j^s$ [*update of the residual flow $\bar{z}_j^s$*]
          $\rho_j^s \leftarrow (1 - \beta)\rho_j^s$ [*updating the proportion ρ_j^s*]
        endif
      endfor
      $\bar{u}_{jk} \leftarrow \bar{u}_{jk} - \beta d_k$ [*update of the residual capacity $\bar{u}_{jk}$*]
      if $\bar{u}_{jk} = 0$ **then**
        for $s \in \overline{W}$ **do**
          $\overline{E}_j^s \leftarrow \overline{E}_j^s - \{k\}$ [*updating the residual order $\overline{E}_j^s$*]
        endfor
      endif
    endfor
    if $\beta = 1$ **then** [*end of micro loading*]
      $\rho_j^{\bar{s}} = 0$
    endif
  endwhile
  for $k \in E_j^{\bar{s}}$ **do**
    $C_j^{\bar{s}} \leftarrow C_j^{\bar{s}} + \pi_{jk}^{\bar{s}}(c_{jk} + C_k^{\bar{s}})$ [*update of cost $C_j^{\bar{s}}$*]
  endfor
  if $j = q$ **then** $C^{\bar{s}} \leftarrow C_q^{\bar{s}}$ [*cost of optimal strategy*]
  endif
endfor

6. The Equilibrium Model

By definition, a strategic vector x^* is in equilibrium if it lies in the set X of demand-feasible vectors and satisfies the variational inequality VI(C, X)

$$\langle C(x^*), x^* - x\rangle \leqslant 0 \quad \forall y \in X.$$

Even though the model is driven by simple parameters, namely travel costs and link capacities, the resulting variational inequality is both nonlinear and asymmetric! In this section we will analyze some of its properties and nonproperties as well.

6.1. A "Braess Paradox"

In a standard model of traffic assignment with nonconstant arc costs, it might occur that an improvement to the network, e.g., the addition of an arc, results in an increase of the travel times for *all* users of the network. This situation is known in the literature as the "Braess paradox" (Braess 1968).

A similar situation may arise in the strategic model. Indeed, reconsider the capacitated network of Figure 1, where the equilibrium cost was equal to 185. If arc $(2, 3)$ is deleted from the network, the entire equilibrium flow is redirected toward strategy $s_2 = ([3, 2], [5], [5, 4], [5], [\,])$, whose expected cost $(2/10)100 + (8/10)200 = 180$ is less than 185. This shows that a decrease in arc capacity may lead to an improvement in travel time for *all* users. This is not surprising, in the view that equilibrium and optimality are two different concepts.

6.2. Integrability

To show that C may fail to be a gradient mapping, consider the network of Figure 4. If the sum $x_1 + x_2$ of the strategic flows is greater than the capacity of the first arc, one has, in both the single queue and parallel cases:

$$C_1(x_1, x_2) = \frac{c_1}{x_1 + x_2} + \left(1 - \frac{1}{x_1 + x_2}\right)c_2,$$
$$C_2(x_1, x_2) = \frac{c_1}{x_1 + x_2} + \left(1 - \frac{1}{x_1 + x_2}\right)c_3.$$

Figure 4. Nonintegrability of the cost mapping.

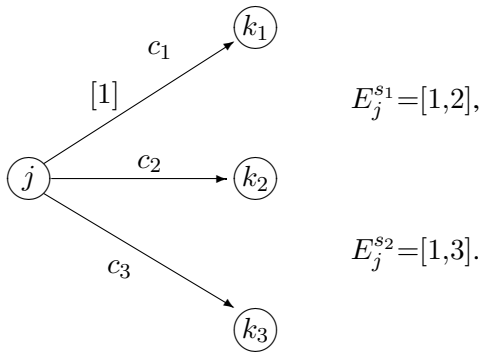
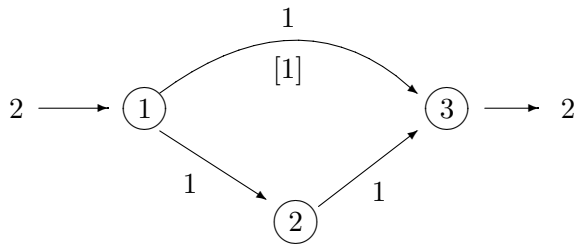


Figure 5. Nondifferentiability of the cost mapping.



The cross partial derivatives of C are given by the expressions

$$\frac{\partial C_1}{\partial x_2}(x_1, x_2) = \frac{c_2 - c_1}{(x_1 + x_2)^2},$$
$$\frac{\partial C_2}{\partial x_1}(x_1, x_2) = \frac{c_3 - c_1}{(x_1 + x_2)^2},$$

and are clearly distinct if $c_3 \neq c_2$. Hence, the Jacobian of C is asymmetric and the variational inequality VI(C, X) cannot be reformulated as a "standard" optimization problem over the feasible set X.

6.3. Differentiability

Let us consider the situation illustrated in Figure 5. The two strategies of interest are $s_1 = ([3, 2], [3], [\,])$ and $s_2 = ([2], [3], [\,])$. It is not difficult to check that the cost function is

$$C(x_1, x_2) = (\max\{1, 2 - (1/x_1)\}, 2),$$

whose first partial derivative with respect to x_1 does not exist when $x_1 = 1$ (the left derivative is zero while the right derivative is equal to 1). Note that in this counterexample, the single queue and parallel processings yield identical flows and access probabilities, so that the example is valid for both queue disciplines.

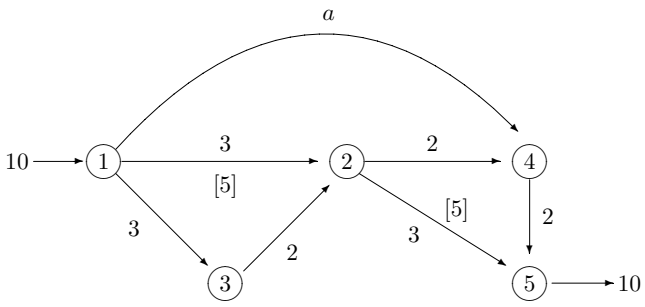
6.4. Continuity

Under the PQP rule, it can be shown that the mapping is not always continuous. In the SQP case, the mapping C is continuously dependent on the access probabilities π_{ij}^{kl}, which are defined as ratios of residual capacities over flows. Whenever the degenerate situation where arcs get saturated simultaneously does not occur, these probabilities are continuous functions of the strategic flow vectors. In the SQP case, we conjecture that C is continuous in the degenerate case as well.

6.5. Monotonicity, Existence, Uniqueness, and Convexity

Monotonicity is a key functional property that is required to establish the convergence of most iterative algorithms for solving equilibrium problems modeled as variational

Figure 6. A nonmonotone cost function.



inequalities. A function F from a convex subset X of R^n into R^n is monotone on X if, for every x and y in X, there holds

$$\langle F(x) - F(y), x - y \rangle \geqslant 0.$$

Unfortunately, the function C of the strategic model may fail to be monotone. Consider the situation illustrated in Figure 6, together with the strategies $s_1 = ([2,4],[5,4],[\,],[5],[\,])$, $s_2 = ([3],[5,4],[2],[5],[\,])$, and the strategic flow vectors of the form $x = (x_1, x_2)$. Because both strategies have the same preference order at the unique node where they compete, the single queue and parallel loadings are identical. For $x^1 = (4,6)$, $\pi_{12}^{s_1}(x) = 1$ and we have

$$C^{s_1}(x) = 3 + \frac{1}{2} \times 3 + \frac{1}{2} \times 4 = 6\frac{1}{2},$$
$$C^{s_2}(x) = 3 + 2 + \frac{1}{2} \times 3 + \frac{1}{2} \times 4 = 8\frac{1}{2}.$$

For $x^2 = (8,2)$ we obtain

$$C^{s_1}(x) = \frac{3}{8} \times (a+2) + \frac{5}{8}\left(3 + \frac{5}{7} \times 3 + \frac{2}{7} \times 4\right) = \frac{21a + 262}{56},$$
$$C^{s_2}(x) = 3 + 2 + \frac{5}{7} \times 3 + \frac{2}{7} \times 4 = \frac{58}{7}.$$

Now,

$$\langle C(x^1) - C(x^2), x^1 - x^2 \rangle = \frac{1}{56}\langle (102 - 21a, 12), (-4, 4) \rangle,$$

which is negative if $a < 90/21$. This shows that the cost function C is not monotone.

Under the assumption that the degenerate case does not occur, however, existence of at least one solution follows from the continuity of the mapping C and the compactness of the feasible set X. As in linear programming, multiple solutions can coexist and form a convex set. Under the SQP rule, we conjecture that the set of equilibria is indeed convex, a property not shared by the PQP rule (see Marcotte and Nguyen 1998 for a counterexample).

6.6. Monotonicity of the Preference Order

To efficiently compute the best strategic response to a given strategic flow vector x and conditional access probabilities $\pi(x)$, one must, at each node of the transportation network, determine an optimal preference order for each strategy. (Actually, the strategies having identical destination nodes will have identical preference orders.) The optimal order E_j^s should minimize the expression

$$\sum_{k \in E_j^s} \pi_{jk}^s (c_{jk} + C_k^s).$$

A greedy way for determining a "good" order consists in sorting the nodes of the forward star of node j upward with respect to the costs $c_{jk_i} + C_{k_i}^s$. This yields

$$E_j^{\tilde{s}} = [k_1, k_2, \ldots, k_l],$$

where

$$c_{jk_i} + C_{k_i}^s \leqslant c_{jk_{i+1}} + C_{k_{i+1}}^s, \quad i = 1, 2, \ldots, |j^+| - 1. \tag{3}$$

While this choice may be suboptimal under PQP, the next proposition shows that it yields the optimal ordering under the SQP rule.

PROPOSITION 1. *In an optimal strategic answer to a given strategic flow vector x and its associated conditional access probability vectors $\pi(x)$, let the preference sets E_j^s be ordered upward with respect to the quantities $\{c_{jk_i} + C_k^s\}_{k \in j^+}$. This choice is optimal in the single queue case.*

PROOF. Let us consider an arbitrary preference order

$$E_j^s = [k_1, k_2, \ldots, k_l]$$

with $l = |j^+|$. With each node $k \in j^+$ we associate a *weight* $e_k = c_{jk} + C_k^s$ and define an *adjacent switch* as the permutation of two adjacent nodes in E_j^s. The adjacent switch of nodes $E_j^s(i)$ and $E_j^s(i+1)$ is *increasing* if the weight of the first node is greater or equal to that of the second node. We will show that an increasing adjacent switch on E_j^s can only decrease the value of C_j^s.

Let $E_j^{\tilde{s}}$ denote the preference order resulting from an increasing adjacent switch performed on an index i of E_j^s, i.e.,

$$E_j^{\tilde{s}} = [k_1, k_2, \ldots, k_{i-1}, k_{i+1}, k_i, \ldots, k_l],$$

and let us summarize the micro-loading phase (loading of a zero flow associated with the optimal strategy currently built) of strategies s and $\tilde{s}$ at node j. After arc (j, k_{i-1}) gets saturated, an *equal* proportion β of the residual flows of strategies s and $\tilde{s}$ is assigned to nodes k_i and k_{i+1}, respectively. At this point, both strategies share the same residual flow proportion ρ. We consider three cases:

Case 1. Arc (j, k_i) gets saturated first. When arc (j, k_i) becomes saturated, an equal proportion β of the residual

Table 9. Conditional access probabilities to arcs (j, k_i) and (j, k_{i+1}) for s and $\tilde{s}$.

	Case 1		Case 2		Case 3	
	(j,k_i)	(j,k_{i+1})	(j,k_i)	(j,k_{i+1})	(j,k_i)	(j,k_{i+1})
s	$\beta\rho$	$\beta'\rho'$	$\beta\rho+\beta'\rho'$	0	ρ	0
$\tilde{s}$	0	$\beta\rho+\beta'\rho'$	$\beta'\rho'$	$\beta\rho$	0	ρ

flow of s and $\tilde{s}$ is assigned to nodes k_i and k_{i+1} and the node k_i is deleted from the residual preference order of strategy s. At this point, strategies s and $\tilde{s}$ have the same residual preference order and residual flow proportion $\rho' = \rho(1-\beta)$. It follows that the micro-loading phase terminates identically for both strategies. We denote by β' the proportion of each strategic flow that accesses node k_{i+1}.

Case 2. Arc (j, k_{i+1}) gets saturated first. The analysis is symmetric with that of Case 1. We let β denote the proportion of residual flow from s and $\tilde{s}$ assigned to nodes k_i and k_{i+1} before arc (j, k_{i+1}) gets saturated, and we let β' denote the proportion of residual flow assigned to node k_i.

Case 3. No arc gets saturated. This constitutes a limiting case of Case 1, where $\beta = 1$.

The output of the micro-loading of strategies s and $\tilde{s}$ is displayed in Table 9. As a general rule, we obtain the relationships

$$\pi^s_{jk_i} + \pi^s_{jk_{i+1}} = \pi^{\tilde{s}}_{jk_i} + \pi^{\tilde{s}}_{jk_{i+1}},$$
$$\pi^{\tilde{s}}_{jk_{i+1}} \geqslant \pi^s_{jk_{i+1}},$$

from which we deduce

$$C^s_j - C^{\tilde{s}}_j = (e_{k_{i+1}} - e_{k_i})(\pi^s_{jk_{i+1}} - \pi^{\tilde{s}}_{jk_{i+1}}) \geqslant 0.$$

Thus an increasing adjacent switch does not deteriorate the preference order E^s_j, and it is always possible, by performing a finite number of increasing adjacent switches, to transform an arbitrary optimal preference order into the preference order $E^{\tilde{s}}_j$, which must therefore be optimal. □

7. Solution Algorithms

7.1. General Considerations

The quest for a strategic equilibrium comes down to finding a vector x^* that solves the variational inequality VI(C, X), i.e.,

$$\langle C(x^*), x^* - x\rangle \leqslant 0 \quad \forall x \in X.$$

The challenge in solving this mathematical program is threefold:

- In contrast with the uncapacitated case, strategic flows, of which there is an exponential number, have to be kept explicitly in order to derive the cost function C.
- The cost function is not available in closed form. This rules out methods using sophisticated linesearch techniques.
- The cost function cannot be assumed to be either integrable (a gradient mapping), differentiable nor monotone.

The first "nonproperty" limits the algorithmic choice to restriction strategies. Within the restriction framework, we developed variants of the "Frank-Wolfe" and projection methods. Because the cost function C is nonmonotone, convergence of the iterates towards an equilibrium solution x^* is not guaranteed. In this context, the algorithms must be viewed as "natural" but heuristic procedures.

7.2. A Stopping Criterion

"Gap functions" provide a computable criterion for deciding whether the current iterate is in near-equilibrium. A nonnegative function g is a gap function if its zeroes coincide with the solutions of the variational inequality. A natural member of this class, defined as

$$g_P(x) = \max_{y \in X} \langle C(x), x - y\rangle, \tag{4}$$

is a byproduct of the computation of a best strategy via the CAPSHORT procedure. In this work we make use of a relative gap function, which is simply a scaled version of g_P:

$$g_R(x) = \frac{g_P(x)}{\langle C(x), x\rangle}. \tag{5}$$

The procedure that evaluates the gap function is listed below.

PROCEDURE GAP (x)
input: $x = \{x_s\}_{s \in W}$ [*strategic flow vector*]
output: $g_R(x)$ [*value of relative gap at* x]
$\bar{x}$ [*extremal solution of* $\min_{y \in X}\langle C(x), y\rangle$]

INITIALIZATION
for $s \in W$ **do**
$\bar{x}_s \leftarrow 0$ [*optimal flow assigned to strategy* s]
endfor
$C^T = 0$ [*total cost* $\langle C(x), x\rangle$]
$C^* = 0$ [*optimal value of linear subproblem*]

COMPUTATION OF TOTAL COST
CAPLOAD$(x) \rightarrow z, C$ [*loading of* x]
for $s \in W$ **do**
$C^T \leftarrow C^T + x_s C^s$ [*total cost update*]
endfor

COMPUTATION OF OPTIMAL TOTAL COST
for $(q, r) \in L$ **do**
  CAPSHORT$(z, (q, r)) \to \bar{s}, C^{\bar{s}}$ [*computing an optimal strategy*]
  $C^* \leftarrow C^* + \bar{x}_{\bar{s}} C^{\bar{s}}$ [*optimal total cost update*]
endfor
$g_R(x) \leftarrow (C^T - C^*)/C^T$ [*gap evaluation*]

7.3. A Linearization Approach

Each iteration of the Frank-Wolfe (FW) linearization approach (Frank and Wolfe 1956) method for minimizing a differentiable function f over a compact and convex set X takes the form

$$x^{k+1} = x^k + \theta_k(\bar{x}^k - x^k),$$

where $\bar{x}^k \in \arg\min_{y \in X} \langle \nabla f(x^k), y \rangle$ and $\theta_k \in [0, 1]$ is some suitable stepsize. The popularity of this approach for solving traffic assignment problems is due to its efficient handling of network structures: Only shortest paths have to be computed for each origin. In the algorithmic scheme STRATEQ1, the FW strategy is adapted to variational inequalities in conjunction with a stepsize rule based on the harmonic sequence $\theta_k = 1/(k+1)$.

STRATEQ1
input: strategic flow vector x, tolerance factor ϵ, working set W
output: strategic equilibrium x^*

INITIALIZATION
GAP$(x) \to \bar{x}, g_R(x)$
$k \leftarrow 1$

MAIN LOOP
while $g_R(x) \geqslant \epsilon$ **do**
  $x \leftarrow (1 - \theta_k)x + \theta_k \bar{x}$
  GAP$(x) \to \bar{x}, g_R(x)$
  $k \leftarrow k + 1$
  $W \leftarrow W \cup \{\bar{x}\}$
endwhile

The initial vector x is computed with respect to free-flow travel times. Although the cost of its origin-destination components is equal to that of shortest O-D paths, all the relevant strategic information is yet produced by the CAPSHORT procedure.

To limit the size of the working set W, we adopt a restriction strategy which is reminiscent of the RSD (Restricted Simplicial Decomposition) technique of Lawphongpanich and Hearn (1984). At every iteration, an optimal strategy $\bar{s}$ corresponding to some O-D pair is inserted in the working set W whenever its cost (with respect to the current strategic vector x) is *significantly* less than that of the current average cost, i.e.,

$$C^{\bar{s}} \leqslant C^{\tilde{s}} - \epsilon_1$$

for some predetermined tolerance ϵ_1. Moreover, any strategy whose flow value falls below some threshold value ϵ_2 is removed from the working set.

A drawback of the Frank-Wolfe method is the averaging aggregation process that takes place over all O-D pairs: adopting a common stepsize may actually worsen the equilibrium conditions for some O-D pairs and have an adverse effect on convergence. To remedy this situation, one may use different stepsizes for each O-D pair. In the scheme STRATEQ2, a disaggregate stepsize rule is based on the relative costs of $\bar{s}$ and the average cost C^s of strategies s for each individual O-D pair:

$$\theta_s^k = 1 - \frac{C^{\bar{s}}}{C^s}.$$

The number θ_s^k measures the average dissatisfaction of users associated with the O-D pair k. The implementation of this scheme does not incur much computational work since the strategic information is readily available in disaggregate form.

7.4. Projection Algorithms

A vector x^* is a solution of the variational inequality VI(F, X) if and only if it is a solution of the fixed point problem

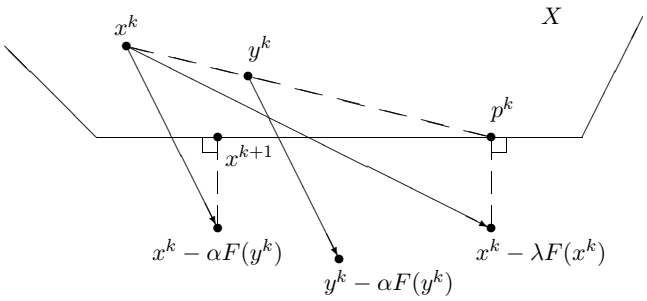
$$x^* = p_\alpha(x^*),$$

where α is a positive scalar and $p_\alpha(x) = \text{proj}_X(x - \alpha F(x))$ denotes the Euclidian projection of the vector $x - \alpha F(x)$ over the convex set X. In our context, the set X is defined as the convex combination of strategic flows in the working set W, i.e., the Euclidian product of simplices defined by nonnegativity and demand constraints. The "vanilla style" projection algorithm generates a sequence of iterates according to the formula

$$x^{k+1} = p_\alpha(x^k) = \text{proj}_X(x^k - \alpha F(x^k)),$$

where the projection operation can be performed very efficiently, due to the simplicial structure of the set X. Several variants of this basic scheme have been proposed over the years. One of them, due to Konnov (1993), converges under weaker conditions than those of the standard algorithm, and performs automatic updates of its internal parameters. (Konnov's scheme is globally convergent if the cost function C is pseudomonotone over the feasible set X. A function F is pseudomonotone on a convex set X if, for any x and y in X, there holds: $\langle F(x), x - y \rangle \leqslant 0 \Rightarrow \langle F(y), x - y \rangle \leqslant 0$. A monotone function is pseudomonotone but the converse statement does not hold in general.) For specific choices of these parameters, Konnov's algorithm reduces to the robust extragradient method of Korpelevich (1977).

Figure 7. Konnov's algorithm.



Konnov's algorithm involves positive parameters α, λ, and $\theta \in (0, 1)$. The computation of x^{k+1} requires two projections and one convex combination, according to the formulae

$$p^k = \text{proj}_X(x^k - \lambda F(x^k)),$$
$$y^k = (1-\theta)x^k + \theta p^k,$$
$$x^{k+1} = \text{proj}_X(x^k - \alpha F(y^k)),$$

which are illustrated geometrically in Figure 7. The computational cost of these operations is very small with respect to the CAPLOAD and CAPSHORT steps.

To take advantage of the ease of implementation of the FW method and the better convergence rate of the projection algorithm, we consider the following combination of both methods:

- the iterations are regrouped into cycles;
- each cycle consists of $K = K_1 + K_2$ iterations;
- the first K_1 iterations are projection iterations;
- the remaining K_2 iterations are FW iterations whose role is mainly to enrich the working set W through algorithm CAPSHORT.

8. Numerical Results

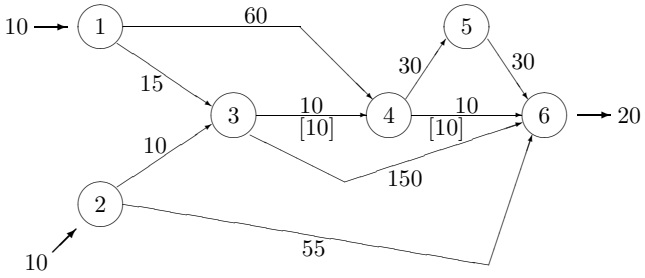
The numerical tests have been designed to illustrate the convergence behavior of the algorithms. They have been performed both on a small example for which the equilibrium was known and on larger networks where restriction strategies had to be implemented. Numerical results are provided for both the SQP and PQP rules. The quality of an approximate solution x is measured by the value of the relative gap function $g_P(x)$.

8.1. A Small Network

Our first test problem is based on the network of Figure 8, and is derived from a transit assignment situation where links $(3, 4)$ and $(4, 6)$ are served by two bus lines with an equal capacity of ten seats. The strategies of interest are listed in Table 10.

In this example, it can be argued that strategy s_3 carries positive flow at equilibrium. Otherwise, its cost (30) would be less than the expected cost of strategy s_4 (55).

Figure 8. A small test network.



This shows that, at equilibrium, the (expected) cost of s_3 is at most 55. Because the cost of strategy s_2 exceeds the cost of strategy s_3 by five units, we deduce that, at equilibrium, it must be equal to $55 + 5 = 60$, which is less than the cost of strategy s_1. Hence, all flow originating from node 1 will use strategy s_1. Now, if all flow from origin 2 used s_3, the cost of this strategy would be equal to

$$C_3^* = 10 + \frac{10}{20} \times (10 + 10) + \frac{10}{20} \times 150 = 95,$$

clearly a nonequilibrium situation. We conclude from these observations that both x_3 and x_4 are positive. To find the value of x_3, we simply write that its cost must be equal to 55, i.e.,

$$10 + \frac{10}{10 + x_3^*}(10 + 10) + \frac{x_3^*}{10 + x_3^*}150 = 55.$$

The solution of this linear equation, $x_3^* = 50/21$, yields the unique equilibrium strategic flow $x^* = (10, 0, 50/21, 160/21)$, both for the single queue and parallel cases.

The convergence of the STRATEQ1 method (Table 11), based on harmonic stepsizes, is quite typical of FW's sublinear behavior. The adaptive stepsize strategy STRATEQ2 (Table 12), which performs a scaling with respect to each O-D pair's contribution, significantly improved the convergence to the equilibrium solution. As expected, the projective methods, under standard choices of their respective parameters, converged yet faster. The fastest was the standard projection method (Table 13), followed closely by Konnov's method (Table 14) and the extragradient (Table 15). Konnov's algorithm might be preferred to the standard projection method on the basis of its weaker convergence requirements. Table 16, which displays the number of iterations required to achieve a predetermined

Table 10. First example.

Node	1	2	3	4	5	6
s_1	[3]	[ ]	[4, 6]	[6, 5]	[6]	[ ]
s_2	[4]	[ ]	[ ]	[6, 5]	[6]	[ ]
s_3	[ ]	[3]	[4, 6]	[6, 5]	[6]	[ ]
s_4	[ ]	[6]	[ ]	[ ]	[ ]	[ ]

Table 11. STRATEQ1 on the small network.

# Iter.	Strategic Flows				Strategic Costs				Relative Gap (%)
	s_1	s_2	s_3	s_4	s_1	s_2	s_3	s_4	
0	10.00	0.00	10.00	0.00	100.00	70.00	95.00	55.00	35.897
1	5.00	5.00	5.00	5.00	51.67	86.67	46.67	55.00	18.056
2	6.67	3.33	6.67	3.33	76.88	82.50	71.88	55.00	9.052
3	7.50	2.50	5.00	5.00	69.00	80.00	64.00	55.00	5.524
5	8.33	1.67	3.33	6.67	59.69	77.14	54.69	55.00	2.649
10	9.09	0.91	2.73	7.27	58.53	74.17	53.53	55.00	2.177
20	9.52	0.48	2.86	7.14	61.84	72.27	56.84	55.00	0.867
50	9.80	0.20	2.55	7.45	60.54	70.96	55.54	55.00	0.295
100	9.90	0.10	2.48	7.52	60.36	70.49	55.36	55.00	0.163
200	9.95	0.05	2.44	7.56	60.26	70.25	55.26	55.00	0.098
500	9.98	0.02	2.40	7.60	60.03	70.10	55.03	55.00	0.024
1,000	9.99	0.01	2.39	7.61	60.01	70.05	55.01	55.00	0.011
2,000	10.00	0.00	2.38	7.62	60.00	70.02	55.00	55.00	0.005
10,000	10.00	0.00	2.38	7.62	60.00	70.00	55.00	55.00	0.001

Table 12. STRATEQ2 on the small network.

# Iter.	Strategic Flows				Strategic Costs				Relative Gap (%)
	s_1	s_2	s_3	s_4	s_1	s_2	s_3	s_4	
0	10.00	0.00	10.00	0.00	100.00	70.00	95.00	55.00	35.897
1	7.00	3.00	5.79	4.21	72.38	81.54	67.38	55.00	7.221
2	7.34	2.66	4.73	5.27	65.95	80.51	60.95	55.00	5.242
3	7.82	2.18	4.26	5.74	64.82	78.95	59.82	55.00	4.113
5	8.52	1.48	3.65	6.35	63.48	76.44	58.48	55.00	2.619
10	9.39	0.61	2.90	7.10	61.57	72.87	56.57	55.00	0.970
20	9.88	0.12	2.48	7.52	60.33	70.60	55.33	55.00	0.179
50	10.00	0.00	2.38	7.62	60.00	70.01	55.00	55.00	0.002
100	10.00	0.00	2.38	7.62	60.00	70.00	55.00	55.00	0.000

Table 13. Projection algorithm on the small network ($\alpha = 0.2$).

# Iter.	Strategic Flows				Strategic Costs				Relative Gap (%)
	s_1	s_2	s_3	s_4	s_1	s_2	s_3	s_4	
0	10.00	0.00	10.00	0.00	100.00	70.00	95.00	55.00	35.897
1	7.00	3.00	6.00	4.00	73.88	81.54	68.88	55.00	7.616
2	7.77	2.23	4.61	5.39	67.36	79.13	62.36	55.00	4.691
3	8.94	1.06	3.88	6.12	67.33	74.78	62.33	55.00	2.880
5	10.00	0.00	2.66	7.34	62.29	70.00	57.29	55.00	0.516
10	10.00	0.00	2.38	7.62	60.00	70.00	55.00	55.00	0.000

Table 14. Konnov's algorithm on the small network ($\alpha = 0.2$, $\lambda = 0.1$, and $\theta = 0.001$).

# Iter.	Strategic Flows				Strategic Costs				Relative Gap (%)
	s_1	s_2	s_3	s_4	s_1	s_2	s_3	s_4	
0	0.00	10.00	5.00	5.00	51.67	86.67	46.67	55.00	28.485
1	3.50	6.50	5.83	4.17	53.42	88.42	48.42	55.00	20.019
2	7.00	3.00	6.49	3.51	77.19	81.54	72.19	55.00	8.616
3	7.43	2.57	4.77	5.23	66.87	80.21	61.87	55.00	5.211
5	9.50	0.50	3.27	6.73	65.10	72.36	60.10	55.00	1.662
10	10.00	0.00	2.38	7.62	60.00	70.00	55.00	55.00	0.000

Table 15. Extragradient algorithm on the small network ($\alpha = 0.1$).

# Iter.	Strategic Flows				Strategic Costs				Relative Gap (%)
	s_1	s_2	s_3	s_4	s_1	s_2	s_3	s_4	
0	2.00	8.00	6.00	4.00	53.75	88.75	48.75	55.00	22.932
1	3.72	6.28	6.25	3.75	54.24	89.24	49.24	55.00	18.925
2	4.51	5.49	5.77	4.23	55.75	87.72	50.75	55.00	15.380
3	5.26	4.74	5.32	4.68	57.26	86.08	52.26	55.00	12.018
5	6.63	3.37	4.53	5.47	59.81	82.60	54.81	55.00	6.357
10	9.19	0.81	3.13	6.87	62.48	73.75	57.48	55.00	1.417
20	10.00	0.00	2.42	7.58	60.32	70.00	55.32	55.00	0.067
50	10.00	0.00	2.38	7.62	60.00	70.00	55.00	55.00	0.000

accuracy level, allows a quick visual comparison of all five algorithms.

In the vicinity of the equilibrium solution $x^* = (10, 0, 50/21, 160/21)$, the cost function C takes the form

$$C_1(x) = 15 + \left(\frac{10}{x_1 + x_3}\right)\left(10 + \frac{10}{10 + x_2}(10) + \frac{x_2}{10 + x_2}(60)\right) + \left(\frac{x_1 + x_3 - 10}{x_1 + x_3}\right)150,$$

$$C_2(x) = 60 + \left(\frac{10}{10 + x_2}\right)10 + \left(\frac{x_2}{10 + x_2}\right)60,$$

$$C_3(x) = C_1(x) - 5,$$

$$C_4(x) = 55.$$

The Jacobian matrix of C at the solution x^* is

$$J = C'(x^*) = \begin{pmatrix} \frac{441}{52} & \frac{105}{26} & \frac{441}{52} & 0 \\ 0 & 5 & 0 & 0 \\ \frac{441}{52} & \frac{105}{26} & \frac{441}{52} & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Because the eigenvalues of $J + J^t$ are nonnegative, the cost function C is monotone at x^*. This property might partially explain the very good behavior of the solution algorithms.

8.2. A Larger Instance: The Sioux Falls Network

The second series of tests was performed on the network illustrated in Figure 9. This network of 24 nodes, 41 arcs, and 4 O-D couples (see Table 17) is a simplified acyclic version of the Sioux Falls network used by Suwansirikul et al. (1987) in the static case. Although this network is small, the complexity of its strategic structure is very high (see Table 18). In this example the tolerance parameters ϵ_1 and ϵ_2 have been set to 10^{-4} and 0.005 for algorithm STRATEQ1 and to 10^{-4} and 10^{-4} for STRATEQ2. The projective methods were initialized by 100 iterations of the FW algorithm. Next, the method proceeded through 20 cycles, each cycle consisting of

- 100 projection steps over the set of strategies in the current working set;
- 5 FW steps aimed at enriching the working set.

The Frank-Wolfe method with harmonic stepsizes (STRATEQ1) produced a solution with a gap value less than 1% in 20 iterations (Table 19), which is reasonable, and then exhibited the typical tailing effect. However, there is no doubt that global convergence to an equilibrium solution occurs, as observed from the disequilibrium information provided for each O-D pair. While the behavior of STRATEQ2 (disaggregate stepsizes) is similar in the first iterations, the tailing effect is weaker, which allows the method to reach a gap value as small as 10^{-7} (see Tables 20 and 21). Yet better results were achieved by the combined FW-projection strategy (see Table 22), where Konnov's algorithm could achieve a gap value of 2×10^{-8} in as little as 20 cycles, i.e., 100 calls to the procedures CAPSHORT and CAPLOAD.

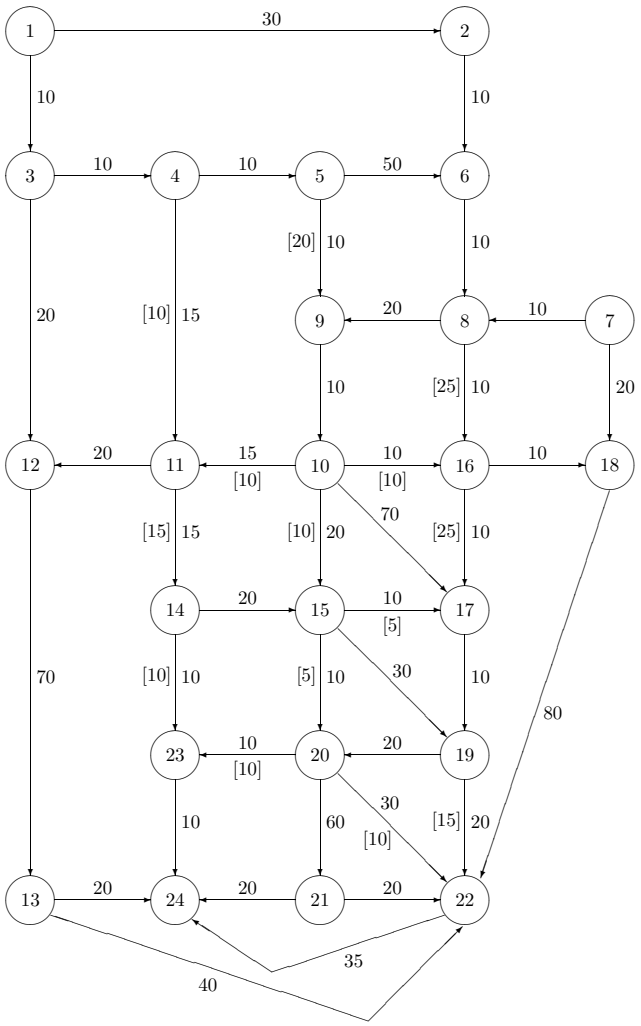
In the parallel case, numerical tests were conducted under both the exact and heuristic procedures for (i) building the best reactive strategy in CAPSHORT; (ii) evaluating the gap function.

Note that the exact method is only practical when the network under consideration has low density, which is (barely) the case of the Sioux Falls network. If the heuristic

Table 16. Number of iterations required to achieve a 10^{-k} gap value (small network).

Method	Gap Function (%)									
	10^1	10^0	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
STRATEQ1	2	13	106	883	8,716	—	—	—	—	—
STRATEQ2	1	10	24	39	54	69	84	99	104	109
Projection	1	5	6	8	9	10	11	13	14	15
Konnov	2	6	7	9	10	11	12	14	15	16
Extragradient	4	11	19	27	35	44	52	60	68	77

Figure 9. The Sioux Falls network.



is adopted, one must realize that the estimated gap is not anymore a reliable measure of equilibrium and can even turn out to be negative!

Under PQP, convergence occurred, albeit at a slower rate (see Tables 23, 24, and 25). When applying the heuristic rule, we could not obtain a true gap value significantly less than 0.5% (see Tables 27 and 28). This is satisfying for all practical purposes, but unsatisfactory from a numerical analysis point of view.

Table 17. O-D information for the Sioux Falls network.

O-D Pair	Notation	Demand
(1, 24)	**OD1**	35
(1, 22)	**OD2**	25
(7, 24)	**OD3**	20
(7, 22)	**OD4**	20

The numerical tests confirmed that the single queue and parallel processing rules yield distinct solutions, both in terms of equilibrium strategies and costs. In general, we observed that the number of strategies involved in the parallel case was higher.

Although monotonicity of the cost function is not required for the convergence of our algorithm, we found it interesting to check whether this condition held for our test problems. Table 29 lists the number of times we uncovered counterexamples for both the single queue and parallel cases. Better (or worse) yet, we could find a counterexample to the pseudomonotonicity of C in the single queue case. This is an important theoretical counter-result because the proof of global convergence for the extragradient and Konnov methods are based on the pseudomonotonicity of the cost function.

8.3. Random Grid Networks

The last series of numerical tests was performed on rectangular grid networks with randomly generated costs. A grid is composed of mn nodes and $2mn - m - n$ arcs; the horizontal arcs are oriented from left to right, and the vertical arcs are oriented either upward or downward, depending on whether they lie on an odd-numbered or an even-numbered column of the grid (see Figure 10). Such a network can trivially be topologically ordered.

A number l of O-D pairs is randomly selected on the grid. To ensure that there exists at least one feasible path from an origin to its corresponding destination, we assume that

- the destination column lies to the right of the origin column;
- the origin and the corresponding destination lie on distinct columns.

Demand is uniformly distributed over all O-D pairs, and arc costs follow a uniform distribution $\mathcal{U}[10, 100]$. The capacity of the arcs on the Hamiltonian path starting at the

Table 18. A typical strategy for the Sioux Falls network (O-D pair 1–24).

Node	1	2	3	4	5	6	7	8
Preference Set	[3]	[ ]	[4]	[11, 5]	[9, 6]	[8]	[ ]	[16, 9]
Node	9	10	11	12	13	14	15	16
Preference Set	[10]	[11, 15, 16, 17]	[14, 12]	[13]	[24]	[23, 15]	[20, 17, 19]	[17, 18]
Node	17	18	19	20	21	22	23	24
Preference Set	[19]	[22]	[20]	[23, 22, 21]	[24]	[24]	[24]	[ ]

Table 19. STRATEQ1 on Sioux Falls network (SQP).

	Number of Strategies				Gap Contribution (%)				
# Iter.	**OD1**	**OD2**	**OD3**	**OD4**	**OD1**	**OD2**	**OD3**	**OD4**	Total
0	1	1	1	1	10.984	5.749	2.833	4.599	24.165
1	2	2	2	2	4.378	4.410	3.915	1.855	14.559
2	3	3	3	3	2.460	0.905	2.673	0.647	6.685
3	3	3	4	3	0.137	1.537	1.416	0.099	3.189
5	4	4	4	4	0.475	1.538	1.014	0.238	3.264
10	4	5	4	5	1.120	0.134	0.606	0.121	1.980
20	4	5	5	5	0.080	0.316	0.306	0.161	0.863
50	4	6	5	5	0.184	0.119	0.124	0.073	0.500
100	4	7	5	5	0.087	0.102	0.066	0.049	0.304
200	4	7	5	5	0.020	0.018	0.035	0.011	0.085
500	4	6	2	4	0.016	0.011	0.002	0.007	0.036
1,000	4	5	2	2	0.017	0.008	0.005	0.006	0.036
2,000	2	5	2	2	0.005	0.001	0.003	0.001	0.011
10,000	2	4	2	2	0.000	0.000	0.001	0.000	0.002

Table 20. STRATEQ2 on Sioux Falls network (SQP).

	Number of Strategies				Gap Contribution (%)				
# Iter.	**OD1**	**OD2**	**OD3**	**OD4**	**OD1**	**OD2**	**OD3**	**OD4**	Total
0	1	1	1	1	10.984	5.749	2.833	4.599	24.165
1	2	2	2	2	3.680	1.791	0.886	1.321	7.677
2	2	2	3	2	1.632	2.069	0.821	0.409	4.932
3	2	3	3	2	0.905	2.090	0.711	0.233	3.939
5	2	3	3	2	0.393	1.735	0.560	0.142	2.830
10	2	3	3	2	0.116	0.950	0.347	0.105	1.518
20	3	3	3	2	0.068	0.250	0.133	0.068	0.520
50	3	4	3	3	0.016	0.051	0.014	0.030	0.111
100	3	5	2	3	0.007	0.030	0.014	0.028	0.080
200	3	5	2	3	0.005	0.018	0.013	0.018	0.054
500	3	5	2	3	0.003	0.011	0.007	0.007	0.028
1,000	3	5	2	3	0.002	0.007	0.003	0.003	0.014
2,000	3	5	2	3	0.000	0.003	0.001	0.001	0.004
10,000	2	5	1	2	0.000	0.000	0.000	0.000	0.000

Table 21. Solutions after 10,000 iterations on Sioux Falls network (SQP).

	Minimum Strategic Cost				
Heuristic	**OD1**	**OD2**	**OD3**	**OD4**	Gap (%)
STRATEQ1	120.00	139.99	111.96	100.00	1.6×10^{-3}
STRATEQ2	120.00	140.00	111.81	100.00	1.8×10^{-7}

Table 22. Combined FW-projection under STRATEQ2 on Sioux Falls network (SQP).

	Gap (%)		
# Cycles	Projection ($\alpha = 0.3$)	Konnov ($\alpha = 0.4$, $\lambda = 0.1$, $\theta = 0.1$)	Extragradient ($\alpha = 0.2$)
1	1.4×10^{-2}	1.5×10^{-2}	1.4×10^{-2}
5	1.4×10^{-2}	1.3×10^{-3}	1.5×10^{-2}
10	6.9×10^{-3}	9.9×10^{-4}	1.3×10^{-2}
15	4.1×10^{-4}	4.2×10^{-5}	6.3×10^{-3}
20	4.0×10^{-5}	2.0×10^{-6}	8.8×10^{-4}

southwestern node is set to $+\infty$, thus allowing uncapacitated paths for every O-D pair. Other arc capacities are randomly distributed according to a uniform distribution $(d/10mn)\mathcal{U}[10, 50]$. The parameters used for each of five grid networks are shown in Table 30. Because the outdegree of each arc is equal to 2, there is no need to distinguish between the single queue and parallel cases.

The numerical tests were based on the FW algorithm coupled with the STRATEQ2's adaptive stepsize strategy. Control parameters ϵ_1 and ϵ_2 were set to 0.1. On all grid test problems, a relative gap measure of 1% was reached after roughly 20 iterations (see Tables 31 to 35), and one can visualize the rate of decrease of the gap in Figure 11. As expected, the running time of a single iteration, illustrated in Figure 12, grows linearly with the number of strategies, which is, on the average, equal to two for each O-D pair. (The software was written in C and run on a Sun Ultra 10 computer operating under Solaris.)

A statistic of interest is the running time of one main iteration of STRATEQ2 as a function of the number of active

Table 23. STRATEQ1 applied to Sioux Falls network (PQP).

	Number of Strategies				Gap Contribution (%)				
# Iter.	**OD1**	**OD2**	**OD3**	**OD4**	**OD1**	**OD2**	**OD3**	**OD4**	Total
0	1	1	1	1	10.840	5.463	5.125	4.370	25.798
1	2	2	2	2	0.879	3.357	2.048	1.518	7.802
2	2	3	3	2	3.123	3.823	0.599	0.643	8.188
3	2	4	3	3	2.048	1.447	1.231	0.238	4.963
5	3	5	3	4	0.893	0.513	0.569	0.163	2.139
10	3	5	3	4	1.074	0.450	0.314	0.071	1.909
20	4	7	5	5	0.225	0.140	0.108	0.068	0.540
50	4	8	5	5	0.179	0.119	0.068	0.036	0.402
100	4	11	5	5	0.058	0.049	0.033	0.009	0.149
200	4	12	5	5	0.020	0.022	0.019	0.007	0.068
500	4	13	4	5	0.005	0.025	0.004	0.012	0.046
1,000	4	11	4	5	0.021	0.029	0.007	0.007	0.064
2,000	4	11	4	5	0.001	0.005	0.001	0.002	0.008
10,000	3	9	4	4	0.001	0.001	0.000	0.000	0.003

Table 24. STRATEQ2 applied to Sioux Falls network (PQP).

	Number of Strategies				Gap Contribution (%)				
# Iter.	**OD1**	**OD2**	**OD3**	**OD4**	**OD1**	**OD2**	**OD3**	**OD4**	Total
0	1	1	1	1	10.840	5.463	5.125	4.370	25.798
1	2	2	2	2	5.237	1.348	2.955	0.998	10.537
2	2	2	3	2	2.473	1.586	1.938	0.399	6.397
3	3	3	3	2	2.503	1.379	1.685	0.332	5.900
5	3	5	3	2	1.460	1.349	1.109	0.213	4.131
10	3	7	3	2	0.509	0.982	0.490	0.221	2.201
20	3	7	3	2	0.239	0.302	0.196	0.111	0.849
50	4	8	4	3	0.046	0.058	0.073	0.005	0.181
100	4	8	4	5	0.019	0.008	0.038	0.005	0.070
200	4	8	3	5	0.008	0.003	0.014	0.002	0.027
500	4	10	3	5	0.002	0.003	0.004	0.001	0.009
1,000	4	10	3	5	0.001	0.002	0.002	0.001	0.006
2,000	4	10	3	5	0.001	0.001	0.001	0.000	0.003
10,000	4	10	3	5	0.000	0.000	0.000	0.000	0.000

Table 25. Solutions after 10,000 iterations on Sioux Falls network (PQP).

	Minimum Strategic Cost				
Heuristic	**OD1**	**OD2**	**OD3**	**OD4**	Gap (%)
STRATEQ1	120.00	140.00	114.30	100.00	2.9×10^{-3}
STRATEQ2	120.00	140.00	114.31	100.00	3.2×10^{-4}

Table 26. Combined FW-projection under STRATEQ2 (Sioux Falls) (PQP).

	Gap (%)		
# Cycles	Projection ($\alpha = 0.2$)	Konnov ($\alpha = 0.2$, $\lambda = 0.1$, $\theta = 0.1$)	Extragradient ($\alpha = 0.1$)
1	1.2×10^{-2}	1.2×10^{-2}	1.3×10^{-2}
5	4.8×10^{-3}	4.8×10^{-3}	7.4×10^{-3}
10	2.3×10^{-3}	2.3×10^{-3}	4.6×10^{-3}
15	1.3×10^{-3}	1.3×10^{-3}	3.1×10^{-3}
20	7.1×10^{-4}	7.1×10^{-4}	2.1×10^{-3}

strategies (see Table 36), which was obtained by running FW without an upper bound on the size of the working set W. To obtain a reliable estimate, the running time was averaged over three iterations for each fixed size of the working set.

Finally, we implemented the standard projection algorithm within an RSD (restricted simplicial decomposition) framework. Each main iteration consists of five steps of the projection algorithm followed by one STRATEQ2 step. The initial strategic flow vector was obtained by performing five STRATEQ2 steps. The results (see Table 37) show a clear improvement over strategy STRATEQ2 and lead us to believe that truly large-scale instances could be tackled using this algorithmic approach.

9. Conclusion

In this paper, we have shown that the concept of "strategy," previously proposed as a paradigm for transit assignment models, can be adapted in a nontrivial fashion to the realm of assignment on capacitated, acyclic networks. Although

Table 27. STRATEQ1: Heuristic approach on Sioux Falls network (PQP).

# Iter.	$\|W\|$	Heuristic Gap (%)	True Gap (%)
0	4	25.798	25.798
1	8	7.802	7.802
2	10	6.310	6.310
3	14	3.229	3.229
5	15	4.208	4.208
10	15	1.616	1.622
20	16	1.022	1.061
50	16	0.474	0.536
100	16	0.133	0.430
200	16	0.104	0.316
500	16	0.019	0.359
1,000	16	0.009	0.333
2,000	15	0.004	0.345

Table 28. STRATEQ2: Heuristic approach on Sioux Falls network (PQP).

# Iter.	$\|W\|$	Heuristic Gap (%)	True Gap (%)
0	4	25.798	25.798
1	8	10.537	10.537
2	9	6.397	6.397
3	11	5.725	5.900
5	13	4.327	4.327
10	14	1.988	2.458
20	14	0.933	0.933
50	16	0.214	0.342
100	16	0.074	0.372
200	15	−0.089	0.626
500	15	−0.044	0.663
1,000	15	−0.016	0.732
2,000	11	−0.001	0.725

Table 29. Influence of queue discipline (Sioux Falls).

		Counterexample to		
Queue Discipline	# Tests	Monotonicity	Pseudo-monotonicity	Suboptimality of CAPSHORT
SQP	1,000	24	1	0
PQP	1,000	1	0	213

Figure 10. A grid network ($m = 4$ and $n = 3$).

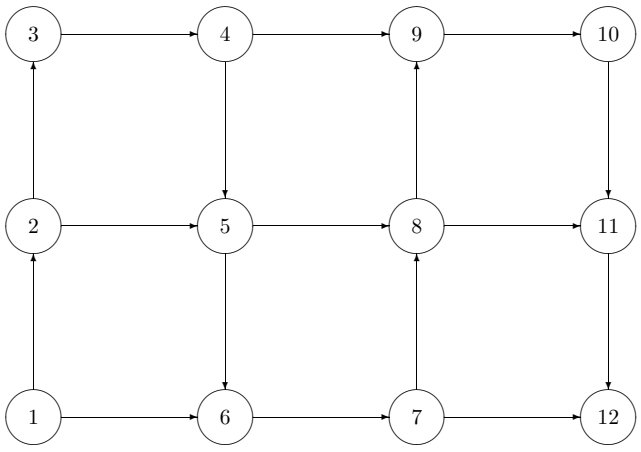


Table 30. Parameters of the grid networks.

	A	**B**	**C**	**D**	**E**
m	10	20	20	20	40
n	8	8	16	32	32
l	10	10	20	20	40
d	1,000	1,000	1,000	1,000	1,000

Table 31. Numerical results: Grid network **A**.

# Iter.	$\|W\|$	Gap (%)	CPU (sec)
0	10	30.70	0.04
1	16	9.23	0.10
2	20	6.91	0.18
3	21	5.61	0.26
5	22	4.10	0.44
10	24	2.35	0.95
20	24	1.07	2.01
40	23	0.35	4.33
60	22	0.15	6.24
80	21	0.08	8.00
100	20	0.05	9.63

Table 32. Numerical results: Grid network **B**.

# Iter.	$\|W\|$	Gap (%)	CPU (sec)
0	10	65.11	0.08
1	19	20.04	0.22
2	28	8.49	0.44
3	33	6.56	0.71
5	35	4.61	1.28
10	39	2.56	2.87
20	40	1.15	6.28
40	40	0.39	13.21
60	35	0.19	19.72
80	29	0.12	25.02
100	27	0.09	29.75

Table 33. Numerical results: Grid network **C**.

# Iter.	$\|W\|$	Gap (%)	CPU (sec)
0	20	47.45	0.55
1	40	15.16	1.66
2	58	9.55	3.39
3	64	7.80	5.38
5	68	5.67	9.83
10	70	3.09	21.38
20	72	1.26	45.92
40	63	0.37	92.57
60	55	0.17	129.69
80	50	0.10	161.56
100	45	0.06	190.78

the model lacks the theoretical properties that would guarantee the uniqueness of the equilibrium solution, our algorithms converged to identical solutions on randomly generated networks, which is reassuring.

The next step of our study will be the analysis of a modified model involving access priorities. Such a model would

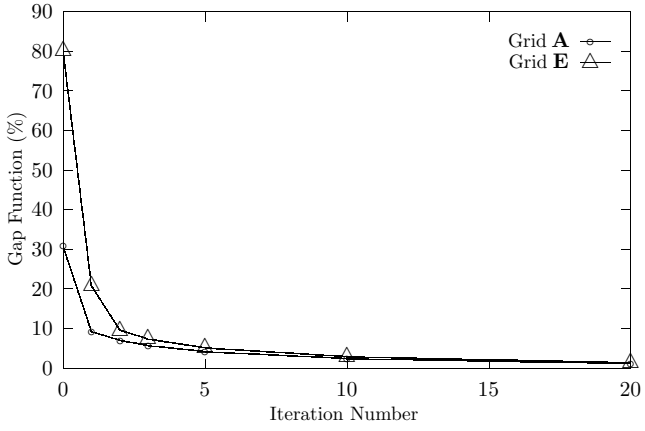
Table 34. Numerical results: Grid network **D**.

# Iter.	$\|W\|$	Gap (%)	CPU (sec)
0	20	68.87	1.26
1	36	27.75	3.71
2	52	11.25	7.59
3	61	8.53	12.49
5	65	5.70	22.87
10	65	2.73	49.43
20	64	0.95	101.60
40	52	0.25	186.62
60	43	0.12	252.32
80	40	0.09	308.99
100	38	0.07	361.00

Table 35. Numerical results: Grid network **E**.

# Iter.	$\|W\|$	Gap (%)	CPU (sec)
0	40	80.06	8.17
1	76	20.69	26.07
2	111	9.56	57.48
3	132	7.28	99.32
5	134	5.06	185.78
10	144	2.81	470.69
20	151	1.24	1,044.05
40	146	0.36	2,074.35
60	124	0.13	2,955.51
80	107	0.06	3,637.78
100	98	0.03	4,229.89

Figure 11. Decrease of the gap function for grids **A** and **E** (STRATEQ2).



subsume a time-discretized formulation of a dynamic traffic assignment model where the FIFO rule would be automatically satisfied and where arc capacities would be an integral part of the model. The time-dependent network on which the model is based will naturally be topologically ordered with respect to the time increments, thus fulfilling the acyclicity condition ensuring the validity of the loading procedure CAPLOAD.

Figure 12. Running time per iteration for grids **C** and **D** (STRATEQ2).

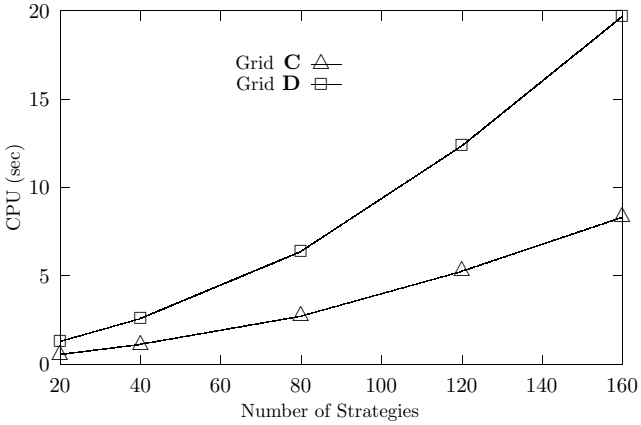


Table 36. Running times per iteration (STRATEQ2).

	CPU per Iteration (sec)				
$\|W\|$	**A**	**B**	**C**	**D**	**E**
10	0.04	0.08	—	—	—
20	0.07	0.14	0.55	1.27	—
40	0.14	0.30	1.12	2.58	8.83
80	0.35	0.87	2.73	6.40	18.01
120	0.75	1.84	5.27	12.35	32.30
160	1.13	2.91	8.33	19.69	47.83

Table 37. Projection algorithm applied to grid network **E**.

# Cycles	$\|W\|$	Gap (%)	CPU (sec)
0	134	5.1×10^{0}	120
1	53	6.3×10^{-2}	241
2	50	5.0×10^{-2}	273
3	51	4.0×10^{-2}	306
4	51	5.0×10^{-2}	340
5	51	4.0×10^{-2}	376

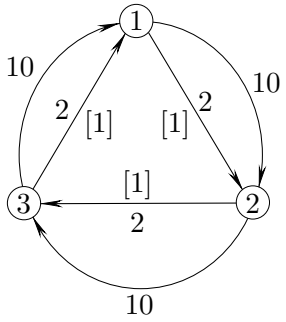
Appendix. Cyclic Networks

Subproblems associated with a given origin being cycle-free, one can address cyclic networks by iteratively solving for equilibria with respect to the origin nodes of the network, à la Gauss-Seidel.

Another approach consists of performing the loading process through an iterative procedure operating on acyclic subnetworks. To illustrate this concept, we consider the "extreme" situation corresponding to the symmetric triangular network of Figure 13, where demand is set to one on the O-D pairs $(1, 3)$, $(2, 1)$, and $(3, 2)$, cost (respectively, capacity) is equal to 2 (respectively, 1) on the capacitated arcs $(1, 2)$, $(2, 3)$, $(3, 1)$, and cost is 10 on the uncapacitated parallel arcs.

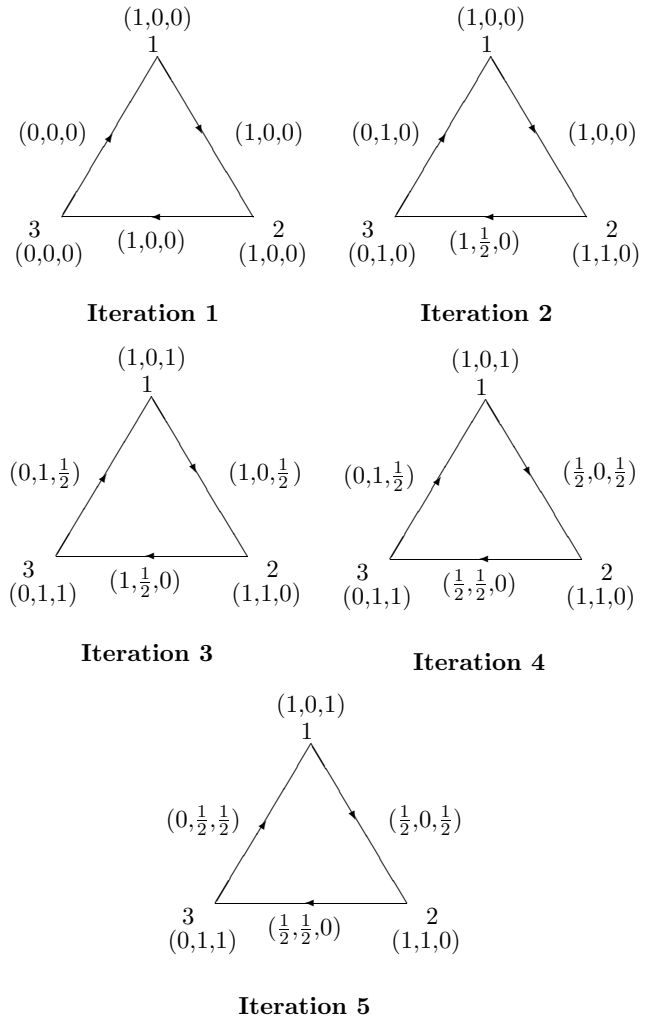
At each iteration, one loads the strategic flow associated with a given O-D pair, assuming that strategic flows

Figure 13. A cyclic network.



for the remaining O-D pairs have been fixed at their preceding values, at *every node* of the transportation network. In this example, optimal strategies are clearly achieved by trying to access a short (capacitated) arc whenever possible. A description of the iterative process is as follows. See also Figure 14, where link flows on capacitated arcs, together with nodal strategic flows, are displayed. The loading process converges in two iteration cycles, when the expected travel cost of each user is equal to 12.

Figure 14. Loading on an acyclic network.



Iteration 1

Flow for O-D pair $(1, 3)$ is simply assigned to the shortest path 1–2–3.

Iteration 2

The entire demand on O-D pair $(2, 1)$ competes with one unit of nodal strategic flow for access to the capacitated arc $(2, 3)$. According to the proportionality rule, half a unit of flow is able to access arc $(2, 3)$. The flows then regroup at node 3 to gain access to arc $(3, 1)$. At this point, note that the total flow on arc $(2, 3)$ exceeds its capacity.

Iteration 3

The same process yields the symmetric situation described in the third network of Figure 14.

Iteration 4

A second assignment of the $(1, 3)$ flow, which now takes into account the competition at the entrance of arcs $(1, 2)$ and $(2, 3)$, yields the situation of the fourth network.

Iteration 5

A second iteration for O-D pair $(2, 1)$ achieves a coherent solution, i.e., terminates the loading process.

REMARK. Although we could not prove it, we conjecture that the above procedure converges in a *finite* number of steps.

References

Astarita, Vittorio. 1996. A continuous time link model for dynamic network loading based on travel time function. J. B. Lesort, ed. *Proc. 13th Internat. Sympos. Transportation Traffic Theory*. Pergamon-Elsevier, Tarrytown, New York, 79–102.

Braess, D. 1968. Über ein Paradoxon aus der Verkehrsplanung. *Unternehmensforschung* **12** 258–268.

Daganzo, Carlos F. 1998. Queue spillovers in transportation networks with a route choice. *Transportation Sci.* **32** 3–11.

Dial, R. B. 1971. A probabilistic multipath traffic assignment model which obviates path enumeration. *Transportation Res.* **5** 83–111.

Frank, M., P. Wolfe. 1956. An algorithm for quadratic programming. *Naval Res. Logist. Quart.* **3** 95–110.

Hearn, D. W. 1980. Bounding flows in traffic assignment models. Research report 80-4, Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL.

Konnov, I. 1993. Combined relaxation methods for finding equilibrium points and solving related problems. *Mathematika* **37** 44–51.

Korpelevich, G. M. 1977. The extragradient method for finding saddle points and other problems. *Matekon* **13** 35–49.

Larsson, T., M. Patriksson. 1998. Side constrained traffic equilibrium models—Traffic management through link tolls. P. Marcotte, S. Nguyen, eds. *Equilibrium and Advanced Transportation Modelling*. Kluwer Academic Publishers, Boston, MA, 125–151.

Lawphongpanich, S., D. W. Hearn. 1984. Simplicial decomposition of the asymmetric traffic assignment problem. *Transportation Res.* **18B** 123–133.

Marcotte, P., S. Nguyen. 1998. Hyperpath formulations of traffic assignment problems. P. Marcotte, S. Nguyen, eds. *Equilibrium and*

Advanced Transportation Modelling. Kluwer Academic Publishers, Boston, MA, 175–199.

Miller-Hooks, E. 2001. Adaptive least-expected time paths in stochastic, time-varying transportation and data networks. *Networks* **37** 35–52.

Nguyen, S., S. Pallottino. 1988. Equilibrium traffic assignment for large scale transit network. *Eur. J. Oper. Res.* **37** 176–186.

Nguyen, S., S. Pallottino. 1989. Hyperpaths and shortest hyperpaths. (Berlin) B. Simeone, ed. *Combinatorial Optimization*. *Lecture Notes in Mathematics*, Vol. 1403. Springer-Verlag, 258–271.

Patriksson, M. 1994. *The Traffic Assignment Problem—Models and Methods*. Utrecht, VSP, BV, The Netherlands.

Schoeb, A. 1999. Une étude théorique et algorithmique d'un modèle d'affectation de trafic avec capacités rigides. Master's thesis, Département d'informatique et de recherche opérationnelle, Université de Montréal, Montreal, Quebec, Canada.

Spiess, H., M. Florian. 1989. Optimal strategies: A new assignment model for transit networks. *Transportation Res. B* **23** 83–102.

Suwansirikul, C., T. L. Friesz, R. L. Tobin. 1987. Equilibrium decomposition optimization: A heuristic for the continuous equilibrium network design problem. *Transportation Sci.* **21** 254–263.

Wardrop, J. G. 1952. Some theoretical aspects of road traffic research. *Proc. Inst. Civil Engr., Part 2*. 325–378.