UNIVERSITÀ
DEGLI STUDI
DI BRESCIA

DIPARTIMENTO DI ECONOMIA E MANAGEMENT

WORKING PAPER

Unfairness Avoidance in System Optimal Routing of Traffic Flows

E. Angelelli
V. Morandi
M.G. Speranza

WPDEM 2/2018

Via S. Faustino 74/b, 25122 Brescia - Italy
Tel +39 030 2988742
email: paqricerca.dem@unibs.it

Unfairness avoidance in system optimal routing of traffic flows

Enrico Angelelli (1) *Valentina Morandi* (1) *M.G. Speranza* (1)

(1) *Department of Economics and Management, University of Brescia, Italy, Corresponding author: valentina.morandi1@unibs.it*

Abstract

In static traffic assignment problems, the user experienced travel time in a system optimum solution can be much higher than in a user equilibrium solution. On the other hand, the total travel time in a user equilibrium solution can be significantly higher than the total travel time in a system optimum solution. A compromise solution between the two assignments seems to be the right choice for traffic regulators aiming at improving the network performance while satisfying users needs. All works in this field consist in restricting ex-ante the set of eligible paths for each driver. However, when demands flow on the road network, these paths could turn out to be very unfair for users and some excluded paths could be much fairer then the eligible ones. In order to overcome this limitation, in this paper a MILP based approach able to minimize the total travel time spent on the road network while controlling the unfairness experienced by users is proposed. Computational results show that the obtained total travel time is very close to the system optimum one while guaranteeing a very low level of experienced unfairness. The underlying idea is to bound the experienced travel time of each user to a fixed threshold, representing the maximum level of unfairness the regulator decides to allow. The MILP formulation requires the enumeration of all feasible paths from each origin to each destination each one corresponding to a binary variable and, hence, it becomes computationally intractable even considering small road networks. To this aim an efficient and accurate heuristic algorithm is also proposed.

Keywords: Traffic assignment, unfairness, matheuristic for traffic assignment, constrained system optimum

1 Introduction

Urban areas are world-wide affected by severe road congestion problems. Traffic congestion is a result of an economic growth that causes an increase of private and commercial transportation. Network infrastructure is not dynamic and, hence, when the demand for transportation increases, congestion occurs. In order to reduce the traffic congestion effects, a centralized traffic assignment method, able to route vehicles to minimize the congestion impact, is a valuable choice. Almost all vehicles are nowadays equipped with sat-nav devices that can also display the current traffic flows and, consequently, return the fastest path for the vehicle. The route suggested by these devices does not consider the impact of individual choices on the congestion of the whole road network. For instance, all commuters entering the network at the same point and heading to the same destination have the same information and, consequently, the same path will be suggested to them. In this case congestion is simply shifted from already congested paths to the suggested ones. Thus, coordination among users is fundamental in planning an efficient traffic assignment. The issue that has to be taken into account is the adherence of drivers to the assignment.

Traditionally, traffic assignment concerns assigning routes to drivers in a transportation networks and is usually defined on a road network with an origin-destination (OD) matrix specifying the demand for transportation, i.e, the number of vehicles per time unit that is expected to travel from each origin to each destination (see Sheffi (1985), Ben-Akiva and Lerman (1985), Florian and Hearn (1999) and de Dios Ortuzar and Willumsen (2011) for demand forecasting methodologies).

In traffic assignment, experienced travel times are computed for each network arc through the so-called *latency function* $t_a(x)$, which depends on the arc traffic flow x. The concept of arc length is defined as the arc *normal length*, i.e. an *a priori* estimate of the travel time of an arc.

Traffic assignment models were first presented in the seminal work Wardrop (1952). In this work, the two most famous principles on traffic assignment (the user equilibrium and the system optimum) are stated and described through their properties. The user equilibrium represents an assignment in which the travel times along all used routes from an origin to a destination are equal and not more than the travel time that would be experienced by a single user on any other route. This is the so-called natural assignment since it is the equilibrium reached when all users decide on their own the route to be used. On the other hand, the system optimum is an assignment in which

the total travel time is minimized and the assignment is centralized.

The difference in terms of total travel time between implementing a user equilibrium and a system optimum traffic assignment is commonly known as *price of anarchy* and represents how much the system would pay in terms of total travel time if the natural assignment is implemented (see Mahmassani and Peeta (1993) for further discussions and references). However, there are drawbacks also in implementing a system optimum traffic assignment. While the user equilibrium ensures fairness for users travelling between the same origin and destination, in a system optimum traffic assignment, some users may be assigned to paths that are much longer than paths assigned to other users for the same OD pair.

In order to reduce the price of anarchy while maintaining fairness between users, the system optimal routing of traffic flows with user constraints was first presented in Jahn et al. (2000). This assignment considers as eligible paths those paths that have a normal length within a certain percentage of the path with the shortest normal length. Later, in Jahn et al. (2005), the work has been improved considering new unfairness measures. Computational results, in both cases, demonstrate that the system optimum total travel time can be nearly achieved with the restricted set of eligible paths and theoretical bounds are derived in Schulz and Stier-Moses (2006). Angelelli et al. (2016a) presents the first attempt to use a linear programming approach to solve the system optimal routing with users constraints problem. Here, the total travel time is minimized while keeping the network non-congested, if possible, or at its minimum congestion level otherwise. The set of eligible paths is restricted as in previous works. Given the fact that the number of paths is in the worst case exponential in the instance size, in Angelelli et al. (2016c) a heuristic algorithm is proposed. Subsequently, in Angelelli et al. (2016b) a linear programming model, in which a flow-dependent latency function is embedded, is presented. The proposed model adopts a piecewise approximation of the convex latency function using only continuous variables. Several variants of the system optimal routing with user constraints have been proposed in literature. In Correa et al. (2007) the maximum experienced travel time instead of the total travel time is minimized allowing all possible paths from origin to destination. In Lujak et al. (2015) a model, in which the weighted geometric mean of the experienced travel times is minimized, is proposed. The solution method is based on a multi-agent negotiation model. Recently, in Angelelli et al. (2018) a trade-off solution between minimizing the maximum and the average arc congestion has been proposed along with a

heuristic algorithm.

The system optimal routing of traffic flows with users constraints shown in Jahn et al. (2005) and, in a linearized version, in Angelelli et al. (2016b), assigns paths to OD pairs to minimize the total travel time experienced by users while bounding the set of eligible paths choosing them a priori on the basis of the path normal length. However, this path selection method does not consider the impact of the actual traffic flows on each arc of the network. Moreover, a path that was eligible a priori may turn out not to respect the desired fairness level. Moreover, when the demand for transportation is particularly high (as during the rush-hour), a priori non-eligible paths could turn out to be the best or one of the best choices considering flows deriving from the assignment. Thus, paths selection should be done considering the current path travel times and by allowing their use only if a certain threshold is not reached. In order to consider the current travel times, the path selection has to be embedded into the model and travel time for each used path is there bounded to be fair enough for users. To this aim, in this paper, we propose a new measure, called *fastest path unfairness*, that is the ratio of the experienced travel time and the minimum travel time the driver could experience on the road network with actual flows. This measure represents the unfairness experienced by using the recommended path instead of following the recommendation of any sat-nav device with real-time traffic information.

A mixed-integer linear programming model, called the *unfairness constrained system optimum model* (UC-SO), is presented that minimizes the total travel time spent on the network while bounding the experienced travel time for each used path to be less than a certain percentage of the travel time that would be experienced on the fastest path given actual network flows for the OD pair. All possible paths from origin to destination are considered as eligible. A piecewise linear approximation of the flow-dependent latency function ensures the linearity of the model. The model requires the enumeration of all possible paths from origin to destination each one associated to a binary variable and, hence, is difficult to solve even when small instances are considered. To this aim, we propose also a matheuristic able to return a high quality solution in a very short time. An extensive computational study shows that the resulting total travel time is very near to the system optimum one while maintaining the experienced unfairness low enough to encourage a complete compliance to the proposed assignment, contrary to the system optimum that usually produces very high unfairness levels for some users.

The remainder of the paper is organized as follows. In Section 2, the unfairness constrained system optimum model (UC-SO) is presented along with three alternative bounding unfairness policies for traffic assignment. In Section 3, the matheuristic to solve the unfairness constrained system optimum is provided. In Section 4, the results of an extensive study of the unfairness constrained system optimum model and the matheuristic algorithm are shown. Finally, in Section 5, some concluding remarks are provided.

2 The unfairness constrained system optimum model

In this section, the unfairness constrained system optimum model (UC-SO), aiming at searching the system optimal path assignment while upper bounding the experienced path travel time, is presented. Consider a directed network $G = (V, A)$, where V and $A \subseteq V \times V$ represent, respectively, the set of vertices and the set of arcs. Arcs $(i, j) \in A$ represent road segments while vertices represent junctions between roads and/or an origin or destination point for an OD pair. A latency function $t_{ij}(x_{ij})$, representing the arc travel time depending on the rate of vehicles x_{ij} entering the arc, is consistently associated to each arc $(i, j) \in A$. In addition, each arc is associated to a number of parameters as the free-flow travel time t_{ij}^{FF} $(= t_{ij}(0))$, the arc travel time under user equilibrium t_{ij}^{UE} and a tuning parameter u_{ij} used to shape the latency function. The most popular latency function is the U.S. Bureau of Public Road (BPR) function $t_{ij}(x) = t_{ij}^{FF}[1 + 0.15(\frac{x_{ij}}{u_{ij}})^4]$ and it is the one used in our model. Transportation demand rates are represented by the set $C \subseteq V \times V$ of origin-destination (OD) pairs. Each OD pair $c \in C$ is associated with an origin $O_c \in V$, a destination $D_c \in V$, and a demand rate d_c from O_c to D_c. All possible paths between each origin O_c and each destination D_c are allowed and the set of paths from O_c to D_c is denoted by K_c with $K = \bigcup_{c \in C} K_c$. The indicator a_{ij}^{kc} takes value 1 if path $k \in K_c$ contains arc $(i, j) \in A$ and takes value 0 otherwise. In addition, each OD pair is consistently associated with its experienced travel time under user equilibrium t_c^{UE} and its fastest path travel time under free-flow conditions t_c^{FF}. We define the *fastest path unfairness* for each path $k \in K$ as the relative difference in terms of experienced travel time between the path k and the fastest path from origin to destination considering the actual arc flows. then, the model bounds unfairness by allowing to use only those paths that have a fastest path unfairness lower than a certain percentage γ_{FP}. The experienced arc travel time is a non-

linear function of the arc flow rate x_{ij}. In order to linearize the model, an approximated version of the total arc travel time is adopted: $\sigma_{ij}(x_{ij}) \approx t_{ij}(x_{ij})x_{ij}$ (for the linearization technique, see Angelelli et al. (2016b)).

An upper bound on the flow rate x_{ij}, U_{ij}, is fixed and each non-linear term $t_{ij}(x_{ij})x_{ij}$ is linearized on the range $[0,U_{ij}]$ by a piecewise linear function.

The range $[0,U_{ij}]$ is partitioned in n flow rate intervals, where n is defined as the accuracy level, at fixed break-points $B = \{b_{ij}^0 = 0, b_{ij}^1, ..., b_{ij}^{n-1}, b_{ij}^n = U_{ij}\}$ with corresponding values $f_{ij} = \{f_{ij}^0 = 0, f_{ij}^1 = t_{ij}(b_{ij}^1)b_{ij}^1, ..., f_{ij}^n = t_{ij}(U_{ij})U_{ij}\}$. The interval width is denoted by $\Delta_{ij}^h = b_{ij}^h - b_{ij}^{h-1}$ $(h = 1, \ldots, n)$.

The UC-SO model follows:

$$\min \quad \sum_{(ij)\in A} \sigma_{ij}$$

$$x_{ij} = \sum_{c\in C}\sum_{k\in K_c} a_{ij}^{ck} y_{ck} \qquad \forall (i,j) \in A \qquad (1)$$

$$d_c = \sum_{k\in K_c} y_{ck} \qquad \forall c \in C \qquad (2)$$

$$x_{ij} = \sum_{h=1}^{n} \lambda_{ij}^h \qquad \forall (i,j) \in A \qquad (3)$$

$$\sigma_{ij} = \sum_{h=1}^{n} \frac{f_{ij}^h - f_{ij}^{h-1}}{\Delta_{ij}^h} \lambda_{ij}^h \qquad \forall (i,j) \in A \qquad (4)$$

$$\tau_{ij} = \sum_{h=1}^{n} \frac{t_{ij}(b_{ij}^h) - t_{ij}(b_{ij}^{h-1})}{\Delta_{ij}^h} \lambda_{ij}^h \qquad \forall (i,j) \in A \qquad (5)$$

$$\tau_{ck} = \sum_{(i,j)\in A} a_{ij}^{ck} \tau_{ij} \qquad \forall c \in C \quad \forall k \in K_c \qquad (6)$$

$$\tau_{ck} \leq (1+\gamma^{FP})\tau_{ck'} + M(1 - v_{ck}) \qquad \forall c \in C \quad \forall k \in K_c \quad \forall k' \in K_c \setminus \{k\} \qquad (7)$$

$$y_{ck} \leq d_c v_{cp} \qquad \forall c \in C \quad \forall k \in K_c \qquad (8)$$

$$x_{ij} < 0 \qquad \forall (i,j) \in A \qquad (9)$$

$$y_{ck} \geq 0 \qquad \forall c \in C \quad \forall k \in K_c \qquad (10)$$

$$0 \leq \lambda_{ij}^h \leq \Delta_{ij}^h \qquad \forall (i,j) \in A \quad \forall h = 1, ..., n \qquad (11)$$

$$v_{ck} \in \{0,1\} \qquad \forall c \in C \quad \forall k \in K_c. \qquad (12)$$

The decision variables y_c^k represent the flow of OD pair $c \in C$ routed on path $k \in K_c$. The variables x_{ij} represent the total flow on arc $(i,j) \in A$. Auxiliary variables λ_{ij}^h represent the amount of flow x_{ij} assigned to interval $[b_{ij}^{h-1}, b_{ij}^h]$. Variables τ_{ij} represent the travel time experienced on arc $(i,j) \in A$ while variables τ_{ck} represent the experienced travel time on the k-th path of OD pair c. Moreover, variables σ_{ij} represent the arc experienced travel time multiplied by the arc flow rate x_{ij}. Binary variables v_{ck} equal 1 if the flow on path $k \in K_c$ is greater than zero and 0 otherwise. Constraints (1) set the flow on an arc as the sum of the flow on each path passing through the arc. Constraints (2) ensure that the demand d_c of OD pair $c \in C$ is routed on paths in K_c. Constraints (3) ensure that the entire flow x_{ij} is covered by λ variables. The convexity of the objective function and constraints (4) guarantee that in an optimal solution the flow x_{ij} is covered by an assignment of λ variables such that $\lambda_{ij}^h > 0$ if and only if $\lambda_{ij}^{h-1} = \Delta_{ij}^{h-1}$ for $h = 2, \ldots, n$. Indeed, the sum of $\sigma_{ij} = \sum_{h=1}^{n} \frac{f_{ij}^h - f_{ij}^{h-1}}{\Delta_{ij}^h} \lambda_{ij}^h$ is minimized and, for each interval h, $\frac{f_{ij}^h - f_{ij}^{h-1}}{\Delta_{ij}^h} > \frac{f_{ij}^{h-1} - f_{ij}^{h-2}}{\Delta_{ij}^{h-1}}$ for convexity. Thus, the model tends to assign vehicles to a lower index interval first since the impact on the objective function is lower. Similarly, Constraints (5) identify the travel time experienced on each arc using λ variables and parameters of the piecewise function. Constraints (6) set the path travel time as the sum of the flow on each arc belonging to path. Constraints (7) ensure that if the path is used, i.e. $v_{ck} = 1$, then the path travel time does not exceed the travel time experienced on all the other OD pair paths by a fixed percentage γ^{FP}. If path is not used ($v_{ck} = 0$) then the path travel time is unbounded. If the path is used ($v_{ck} = 1$), then the path travel time is constrained to be lower than a certain threshold γ^{FP} of the experienced travel time on all paths $k' \in K_c \setminus k$, even if they are not used. Alternative bounding unfairness policies are shown in Sections 2.1. Constraint (8) ensure that the variable v_{ck} is set to 1 if flow y_{ck} on path $k \in K_c$ is greater than zero. Finally, constraints (9-12) define the domain of the variables x_{ij}, y_{ck}, v_{ck} and λ_{ij}^h.

For readers' convenience, the notation used in all models is summarized in Table 1.

2.1 On alternative bounding unfairness policies

The choice of the policy used in bounding the unfairness is crucial for a fair traffic assignment and alternative policies could be implemented, each one resulting in a different assignment. In the following we propose three alternative policies, derived from the unfairness measures proposed in

Table 1. Notation used in UC-SO model

UC-SO model notation

Sets

V	set of vertices
A	set of arcs
C	set of OD pairs
K_c	set of all possible paths for $c \in C$

Parameters

u_{ij}	tuning parameter of the latency function $F(x_{ij})$ of arc $(i,j) \in A$
U_{ij}	maximum flow allowed on arc $(i,j) \in A$
t_{ij}^{FF}	free-flow travel time of arc $(i,j) \in A$
a_{ij}^{kc}	1 if path $k \in K_c$ contains arc $(i,j) \in A$, 0 otherwise
d_c	demand of OD pair $c \in C$
D	total demand: $D = \sum_{c \in C} d_c$
γ_{FP}	maximum fastest path unfairness value allowed for path $k \in K_c$ if used, i.e. if $y_{ck} > 0$
b_{ij}^h	h-th breakpoint of the domain of variable x_{ij}
f_{ij}^h	value of the latency function $F(x_{ij})$ at breakpoint b_{ij}^h, i.e., $F(b_{ij}^h)$
Δ_{ij}^h	length of interval $[b_{ij}^{h-1}, b_{ij}^h]$
n	accuracy level

Primary decision variables

y_{ck}	flow rate of OD pair $c \in C$ routed on path $k \in K_c$
x_{ij}	total flow rate entering arc $(i,j) \in A$: $x_{ij} = \sum_{c \in C} \sum_{k \in K_c^\gamma} a_{ij}^{kc} y_{ck}$

Auxiliary decision variables

λ_{ij}^h	amount of flow assigned to interval $[b_{ij}^{h-1}, b_{ij}^h]$
σ_{ij}	approximated experienced total travel time on arc $(i,j) \in A$
τ_{ij}	approximated experienced travel time on arc $(i,j) \in A$
τ_{ck}	approximated experienced travel time on path $k \in K_c$

Jahn et al. (2005), to bound path unfairness: the free-flow unfairness policy, the user equilibrium unfairness policy and the loaded unfairness policy. The *user equilibrium unfairness* is the ratio between the experienced travel time and the experienced travel time for the same OD pair in a user equilibrium. The *free-flow unfairness* is the ratio between the experienced travel time and the fastest path travel time for the same OD pair with respect to the free-flow travel times. The *loaded unfairness* is the ratio between the experienced travel time and the experienced travel time of the fastest traveller of the same OD pair. In all cases, a new set of constraints have to replace constraints (7) in the UC-SO model.

The free-flow unfairness and the user equilibrium unfairness measures are both evaluated as the ratio between the experienced travel time and the experienced travel time on the same network in a particular status, namely in an empty network (free-flow travel time) or under user equilibrium. The choice of the path is done by considering the current flows on the network but the terms of comparison is evaluated by using an "a priori" measure.

2.1.0.1 The free-flow unfairness

In the *free-flow unfairness* model (FF-UC-SO), each used path travel time is bounded to be lower than a certain percentage of the travel time of the fastest path under free-flow conditions. It reflects an assignment in which users are experiencing a travel time that is no more than a certain percentage of the travel time they could experience on an empty network on the fastest path, namely τ_{ck}^{FF}.

The FF-UC-SO model follows:

$$\begin{aligned}
\min \quad & \sum_{(ij)\in A} \sigma_{ij} \\
& (1)-(6) \\
& \tau_{ck} \leq (1+\gamma^{FF})\tau_{ck}^{FF} + M(1-v_{ck}) \qquad \forall c \in C \quad \forall k \in K_c \qquad (13) \\
& (8)-(12).
\end{aligned}$$

Constraints (13) ensure that if the path is used, i.e. $v_{ck} = 1$, then the experienced path travel time does not exceed the travel time τ_{ck}^{FF} on the fastest path under free-flow conditions by a fixed percentage γ^{FF}. If path is not used ($v_{ck} = 0$), then the path travel time is unbounded when big enough values of M are chosen.

2.1.0.2 The user equilibrium unfairness

In the *user equilibrium unfairness* model (UE-UC-SO), each used path travel time is bounded to be lower than a certain percentage of the user equilibrium travel time experienced by its OD pair. It reflects an assignment in which users are experiencing a travel time that is no more than a certain percentage of the travel time they could experience under a user equilibrium assignment.

The UE-UC-SO model follows:

$$\begin{aligned} \min \quad & \sum_{(ij)\in A} \sigma_{ij} \\ & (1)-(6) \\ & \tau_{ck} \leq (1+\gamma^{UE})\tau_{ck}^{UE} + M(1-v_{ck}) \qquad \forall c \in C \quad \forall k \in K_c \qquad (14) \\ & (8)-(12). \end{aligned}$$

Constraints (14) ensure that if the path is used, i.e. $v_{ck} = 1$, then the path travel time does not exceed the travel time τ_{ck}^{UE} experienced in a user equilibrium traffic assignment by a fixed percentage γ^{UE}. As for the FF-UC-SO model, if path is not used ($v_{ck} = 0$), then the path travel time is unbounded.

2.1.0.3 The loaded unfairness

The loaded unfairness measure uses, as terms of comparison, the smallest travel time experienced by the OD pair given actual traffic flows. Here, used paths are compared with other used paths by the same OD pair while in UC-SO model comparison is made with all OD pair paths, used or not. As already pointed out in the introduction, this is more reliable than free-flow and user equilibrium unfairness measures since it better reflects the choices usually made by drivers which

have access to real-time data and the choice of the path is done by considering the current flows on the network. However, as for the UC-SO model, a significant number of additional constraints have to be added to the model with respect to the FF-UC-SO and the UE-UC-SO model.

In the *loaded unfairness* model (L-UC-SO), each used path travel time is bounded to be lower than a certain percentage of its OD pair used fastest path computed using the actual arc flows. It reflects an assignment in which users are experiencing a travel time that is no more than a certain percentage of the travel time experienced by other OD pair users.

The L-UC-SO model follows:

$$\begin{aligned}
\min \quad & \sum_{(ij)\in A} \sigma_{ij} \\
& (1)-(6) \\
& \tau_{ck} \leq (1+\gamma^L)\tau_{ck'} + M(2-v_{ck}-v_{ck'}) \qquad \forall c \in C \quad \forall k \in K_c \quad \forall k' \in K_c \setminus \{k\} \qquad (15) \\
& (8)-(12).
\end{aligned}$$

Constraints (15) ensure that if the path is used, i.e. $v_{ck} = 1$, then the path travel time does not exceed the travel time experienced on all the other OD pair used paths by a fixed percentage γ^L. If path is not used ($v_{ck} = 0$) then the path travel time is unbounded. If the path is used ($v_{ck} = 1$), then the path travel time is constrained to be lower than a certain threshold γ^L of the experienced travel time on all used paths, namely paths $k' \in K_c \setminus k$ with $v_{ck'} = 1$.

Solving the UC-SO model implies that an intractable number of constraints and variables have to be inserted in the formulation. In order to solve the model via an exact method we have implemented a separation procedure in order to insert only those constraints that are needed to obtain the optimal solution. However, this has turned out to be not sufficient to use the exact formulation on instances with more than 50 nodes. In order to overcome this limitation, in Section 3 an heuristic algorithm to solve the UC-SO model has been presented.

2.2 System optimal routing of traffic flows with user constraints and the UC-SO model: a comparison

As introduced in the literature review, the system optimal routing of traffic flows with user constraints concerns using a restricted set of eligible paths for each OD pair. Eligible paths are those paths that have a normal length that is lower than a certain percentage of the shortest path normal length. We denote this percentage as γ.

Here an example of the different formulations behaviour is proposed. In Figure 1 the network used in the experiment is shown where t^{FF} and u are parameters used for the latency function. In using the formulations proposed by Angelelli et al. (2016b) and by Jahn et al. (2005), the path set is computed by considering the γ percentage. Let the γ be equal to 1%. In this case, only shortest paths from each origin to each destination are considered as eligible and, hence, all demand will flow on them.

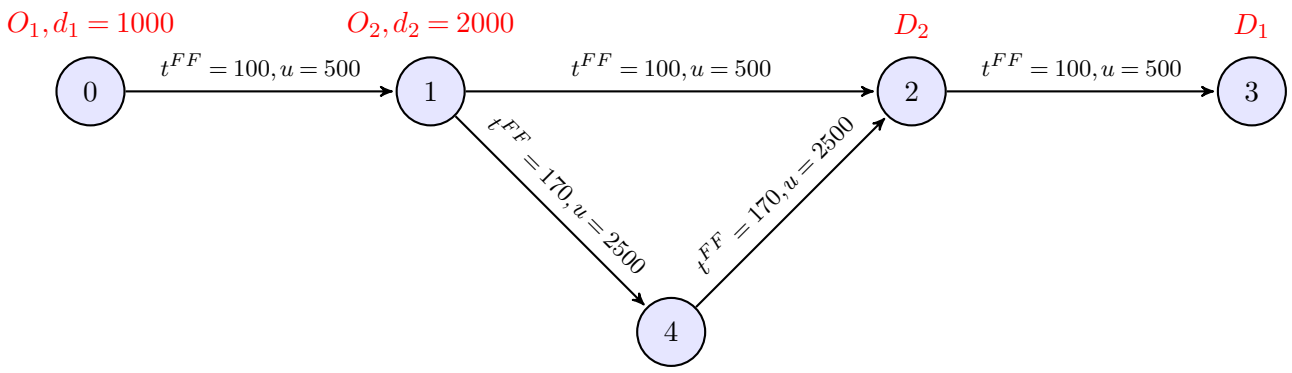


Figure 1. The network

In Figure 2 the solution obtained using the system optimal routing of traffic flows with user constraints, as proposed in Angelelli et al. (2016b), is shown. Here OD pair 1 entirely flows on path 0-1-2-3 while OD pair 2 entirely flows on path 1-2 since they are the only allowed paths. Travel times are the following:

- OD pair 1:
  - 0-1-2-3: 20220 sec

- OD pair 2:
  - 1-2: 19540 sec

The total travel time in this assignment is 59.3 millions of seconds.

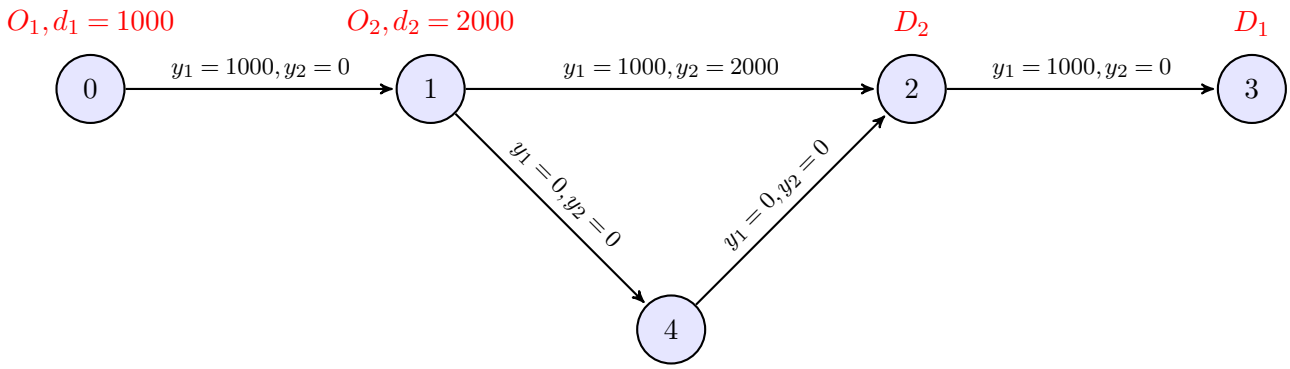


Figure 2. System optimal routing of traffic flows with users constraints using $\gamma = 1\%$

We recall that this solution is not feasible for the UC-SO model since there are other paths that are much faster than the used ones. In order to compare the impact of limiting the available paths, the objective function under a pure system optimal traffic assignment is 1.658 millions of seconds. Noticed that in this pathological case the total travel time, using the model proposed in Angelelli et al. (2016b), is more than one order of magnitude larger. In order to overcome this limitation, the traffic regulator can choose to consider a larger γ value. In fact, allowing $\gamma = 80\%$, the path 0-1-4-2-3 become eligible for OD pair 1.

In Figure 3 the solution using $\gamma = 80\%$ is shown. Here OD pair 1 entirely flows on path 0-1-4-2-3 while OD pair 2 flows on path 1-2. Travel times are the following:

- OD pair 1:
  - 0-1-4-2-3: 1021.31 sec
- OD pair 2:
  - 1-2: 3940 sec

The total travel time here is 8.9 millions of seconds and, even in this case, the solution is infeasible for the UC-SO model since OD pair 2 would experience much less time in flowing on path 1-4-2.

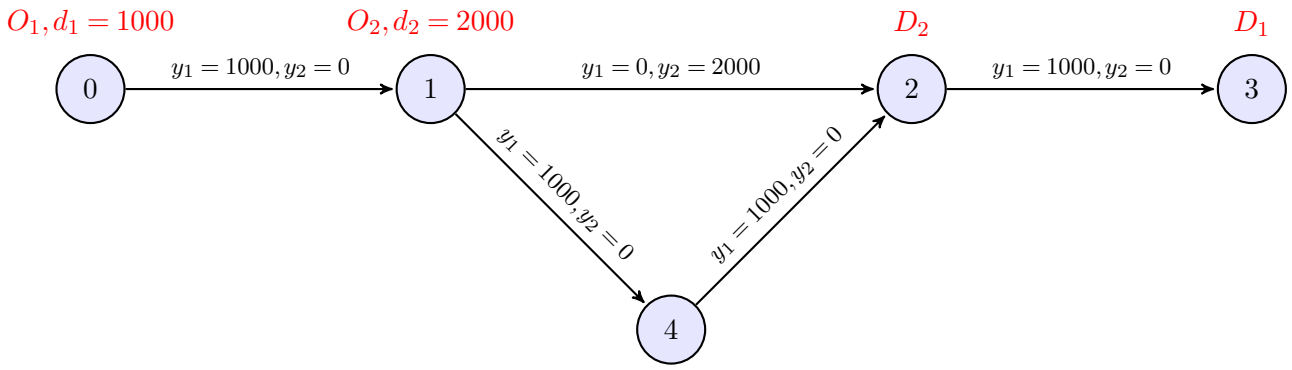


Figure 3. System optimal routing of traffic flows with users constraints using $\gamma = 80\%$

In the following, the same situation is analyzed when using the L-UC-SO model and the UC-SO model and allowing $\gamma^L = \gamma^{FP} = 1\%$. Since we allow all possible paths from origin to destination both OD pairs have two possible paths: 0-1-2-3 and 0-1-2-4-3 for OD pair 1, 1-2 and 1-4-2 for OD pair 2.

In Figure 4 the L-UC-SO model solution is shown. Here OD pair 1 entirely flows on path 0-1-2-3 while OD pair 2 entirely flows on path 1-4-2. Travel times are the following:

- OD pair 1:
  - 0-1-2-3: 1025.056 sec (fastest path)
  - 0-1-4-2-3: 1044.629 sec (the relative difference with the fastest path is 1.9%, not used)
- OD pair 2:
  - 1-2: 341.685 sec (fastest path, not used)
  - 1-4-2: 361.258 sec (the relative difference with the fastest path is 5.7%)

Here no flow is sent on paths 0-1-4-2-3 and 1-2 since even sending a very small ϵ flow on these paths will make this solution not feasible for the UC-SO model because Constraints (7) would be violated.

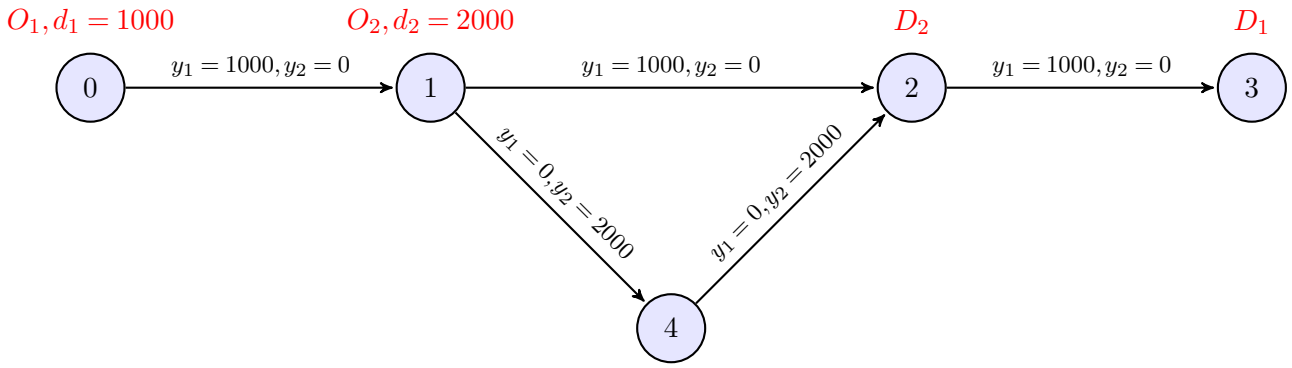


Figure 4. L-UC-SO model solution

In Figure 5 the solution using the UC-SO model is shown. Here OD pair 1 entirely flows on path 0-1-4-2-3 while OD pair 2 flow is sent on its two paths. Travel times are the following:

- OD pair 1:
  - 0-1-2-3: 1040.408 sec (fastest path)
  - 0-1-4-2-3: 1043.978 sec (the relative difference with the fastest path is 0.34%)
- OD pair 2:
  - 1-2: 357.037 sec (fastest path)
  - 1-4-2: 360.607 sec (the relative difference with the fastest path is 0.99%)

Here flow on 1-2 is higher than in the previous solution. This is because the model divides the flow in such a way the fastest path on the network is very similar to the experienced travel times.

Differences in the objective function are very small. In fact, the L-UC-SO model produces a total travel time that is 1.749 millions of seconds while the UC-SO model produces a total travel time that is 1.763 millions of seconds.

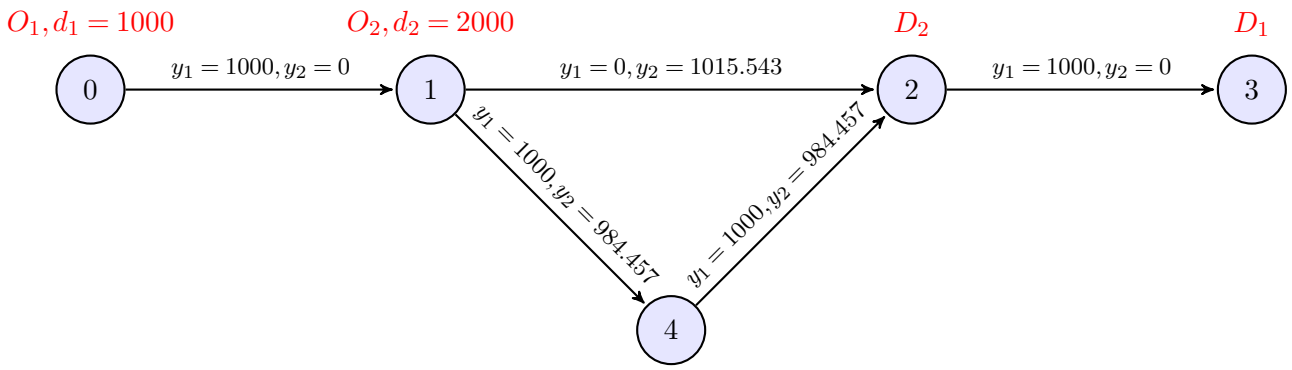


Figure 5. UC-SO model solution

3 A matheuristic for the unfairness constrained system optimum model

Solving the UC-SO model implies the generation of all the feasible paths from origin to destination for each OD pair. The cardinality of the path set obviously depends on the instance size and on the number of OD pairs involved. The number of paths has been proved to be exponential in the instance size (see Angelelli et al. (2016a) for details). Moreover, each path corresponds to a number of continuous variables and to a binary variable in the UC-SO model. Thus, the UC-SO model could turn out to be a MILP formulation containing millions of binary variables and, hence, too big to be tractable in practice. In order to overcome this limitation, a matheuristic algorithm, called *Path Construction Matheuristic algorithm* (PC-M), has been developed. It generates only a small subset of all possible paths from each origin to each destination on which the MILP formulation is small enough to be tractable in practice. The search for these promising paths is iteratively performed by using information provided by the following linear Rel-UC-SO model:

$$\min \quad \sum_{(ij)\in A} \sigma_{ij} + \sum_{c\in C}\sum_{k\in K_c} z_{ck}$$

$$(1)-(11)$$

$$z_{ck} \leq 2 - 2v_{ck} \qquad \forall c \in C \quad \forall k \in K_c \tag{16}$$

$$z_{ck} \leq 1 - \frac{1}{2}v_{ck} \qquad \forall c \in C \quad \forall k \in K_c \tag{17}$$

$$z_{ck} \in [0,1] \qquad \forall c \in C \quad \forall k \in K_c \tag{18}$$

$$v_{ck} \in [0,1] \qquad \forall c \in C \quad \forall k \in K_c. \tag{19}$$

The Rel-UC-SO model consists in the linear relaxation of the UC-SO model (variables v_{ck} are relaxed in Constraints (16)) with a modified objective function and additional technical constraints (16)-(18) that aid the formulation in finding results that are more similar to the MILP formulation ones. In Rel-UC-SO model variables v_{ck} are bounded to be in the interval $[\frac{y_{ck}}{d_c}, 1 + \frac{(1+\gamma^{FP}) \min\limits_{k'\in K_c}\{\tau_{ck'}\} - \tau_{ck}}{M}]$ because of Constraints (13) and (8). Furthermore, z_{ck} is bounded to be greater than $2-2v_{ck}$ and $1-\frac{1}{2}v_{ck}$ while minimizing $\sum\limits_{c\in C}\sum\limits_{k\in K_c} z_{ck}$. If the path is not used ($y_{ck} = 0$), then v_{ck} is as the biggest value for which Constraints (13) are satisfied, i.e. $\min\{1, 1 + \frac{(1+\gamma^{FP}) \min\limits_{k'\in K_c}\{\tau_{ck'}\} - \tau_{ck}}{M}\}$. If the path is used ($y_{ck} > 0$), then value v_{ck} has to be fixed as close as possible to 1. This work is done by Constraints (16)-(18) together with the modified objective function.

The PC-M algorithm starts considering only the fastest path under free-flow conditions for each OD pair, solves the Rel-UC-SO model on these paths, uses the outcome variables to weight an auxiliary network on which a path search is performed, and repeats. The heuristic terminates when no new paths can be found.

PC-M algorithm workflow is shown in Algorithm 1. The set P is initially set as an empty set and the set $\bar{P}$ is initially filled with the shortest paths from O_c to D_c using free-flow travel times. As long as the PC-M algorithm is able to generate new paths, namely while $\bar{P} \neq \emptyset$, the new paths in $\bar{P}$ are added to P, the Rel-UC-SO model is solved using P as path set and, finally, the routine $findFasterPaths(x)$ is called in order to find new promising paths $\bar{P}$. When no new paths

are found, the UC-SO model is solved considering P as path set. The PC-M algorithm solves the UC-SO model allowing a relative MILP gap tolerance equal to 1%. This allow the PC-M algorithm to be quick in finding the heuristic solution. However, a smaller tolerance gap can be chosen in order to obtain an even better heuristic solution at the cost of a significant increase in terms of computational time needed to return a solution.

At each iteration of PC-M algorithm, the linear programming model Rel-UC-SO returns an assignment x. Given the solution x, $findFasterPaths$ routine seeks to find the fastest path for each OD pair $c \in C$ using the current traffic assignment x in determining the traversing times through the latency function. If no new paths can be found, the $findFasterPaths$ routine returns an empty set as shown in Algorithm 2.

Algorithm 1: PC-M algorithm

input : G : graph of the road network,
C : set of OD pairs,
γ^{FP} : maximum fastest path unfairness allowed
output: x : heuristic solution of UC-SO model
global : G, C, γ, P

$P := \emptyset$;
$\bar{P}$:= shortest paths from O_c to D_c using arc lengths $t_{ij} = t_{ij}(0)$ for $c \in C$;

while $\bar{P} \neq \emptyset$ **do**
  $P := P \bigcup \bar{P}$;
  x := optimal solution of Rel-UC-SO with path set P;
  $\bar{P} := findFasterPaths(x)$;
x := optimal solution of UC-SO with path set P;
return x

Algorithm 2: findFasterPaths

input : x traffic flow
output: $\bar{P}$ path set
global : G, C, P

$\bar{P} := \emptyset$;

for $c \in C$ **do**
  p := shortest path from O_c to D_c using arc lengths $t_{ij} = t_{ij}(x_{ij})$;
  if $p \notin P$ **then**
    $\bar{P} := \bar{P} \bigcup \{p\}$;
return $\bar{P}$

4 Computational results

A set of 32 instances with 45 nodes (group A) different in terms of geography and demand patterns, 4 increasing size instances with 270, 300, 330 and 360 nodes (group B) and 2 real-world instances has been used in a computational study to assess the performance of the UC-SO model and of the PC-M algorithm.

Group A and B instances are available at `http://or-brescia.unibs.it/instances`. Details on how the instances are generated are shown in Angelelli et al. (2016a) while real world instances come from the Transportation Networks for Research GitHub repository Stabler (2018). All experiments are conducted by using the UC-SO model and using the PC-M algorithm.

Experiments are organized in two parts. In the first part, the performance of the UC-SO model on instances in group A is discussed. The analysis is conducted with γ^{FP} values ranging from 1% to 10% with increments of 1% (i.e., 10 traffic assignments). The second part is focused on testing the performance of the PC-M algorithm. The first comparison is run on instances of group A, and the second on group B and real world instances. The value of parameter γ^{FP} range from 1% to 10% with step 1%. The accuracy level n used is 1000 for group A and B instances and $n = 100$ for real-world instances.

The models were solved using CPLEX 12.6.0 on a Windows 64-bit computer with Intel Xeon processor E5-1650, 3.50 GHz, and 64 GB Ram. For all experiments, the BPR latency function has been used with $U_{ij} = 4u_{ij}$ (value of u_{ij} is provided in each instance file).

The statistics collected for each instance are described in Section 4.1. Performance of the UC-SO model is presented and discussed in Section 4.2 while the performance of the PC-M algorithm is shown in Section 4.3. Results presentation relies on graphical and tabular representation.

4.1 Statistics

In the following all the computed and collected statistics are defined. Unfairness measures used in our computational study are the the loaded unfairness and the fastest path unfairness.

- **Total travel time**

Θ^*: Total experienced travel time for the traffic assignment produced by the UC-SO model.

Θ_{SO}: Total experienced travel time for the traffic assignment produced by UC-SO model using $\gamma^{FP} = \infty$.

Θ_{UE}: Total experienced travel time for the user equilibrium traffic assignment.

Θ^H: Total experienced travel time for the traffic assignment produced by the PC-M algorithm.

Θ: Optimality gap $\Theta = \frac{\Theta^H - \Theta^*}{\Theta^*}$.

- **User experience for each OD pair** $c \in C$

τ_{ck}: Experienced travel time on path $k \in K_c$.

$\tau_c^L := \min_{k \in K_c} \{\tau_{ck} | y_{ck} > 0\}$ (minimum experienced travel time for OD pair $c \in C$).

$\tau_c^{BC} := \min_{k \in K_c} \{\tau_{ck}\}$ (minimum travel time possible for OD pair $c \in C$).

$I_L := \frac{1}{D} \sum_{c \in C} d_c \sum_{k \in K_c^\gamma} y_{ck} \frac{\tau_{ck} - \tau_c^L}{\tau_c^L}$ loaded unfairness.

$I^{BC} := \frac{1}{D} \sum_{c \in C} d_c \sum_{k \in K_c^\gamma} y_{ck} \frac{\tau_{ck} - \tau_c^{BC}}{\tau_c^{BC}}$ fastest pathunfairness.

- **Computational time**

The computational time is computed considering the time needed to generate paths and solve the model.

- **Memory usage**

The number of generated paths.

4.2 Results for UC-SO model

The traffic assignments produced by the UC-SO model have been analyzed for γ^{FP} values ranging from 1% to 10% with increments of 1%. Reported values are averaged over the 32 instances of group A.

Figure 6 shows the total travel time Θ^* measured in seconds as a function of the parameter γ^{FP}. In order to compare the total travel time obtained using the UC-SO model, in Figure 6 the total travel time under user equilibrium and system optimum is also displayed as horizontal lines. When γ^{FP} is greater than 6%, Θ^* is almost equal to Θ_{SO}. This means that, on average, allowing a fastest path unfairness value of 6% produces almost the same benefits for the system we would have without bounding the experienced fastest path unfairness. Furthermore, the total travel time is always better than the user equilibrium travel time. In fact, the user equilibrium traffic assignment is a feasible solution for the UC-SO model even when $\gamma^{FP} = 0\%$.

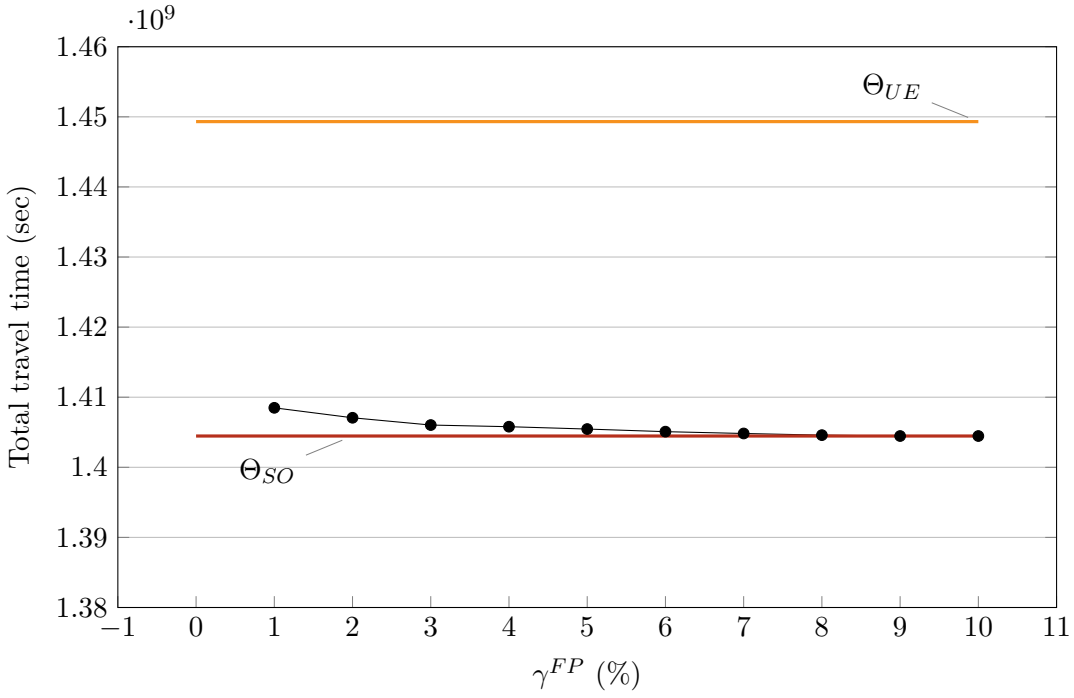


Figure 6. Total travel time, Θ^*, as a function of parameter γ^{FP}

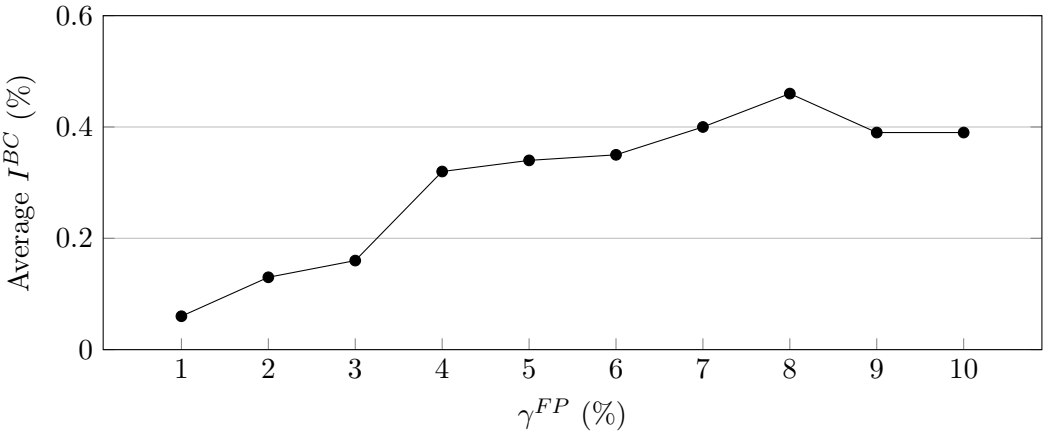
Figure 7(a) shows the average fastest path unfairness, I^{BC}, as a function of γ^{FP}. Allowing a γ^{FP} value equal to 1% the average experienced fastest path unfairness is far from being near the imposed threshold. In fact, it is lower than the 0.05%. Furthermore, even allowing greater γ^{FP} values, the average loaded unfairness is under the 0.5%. This means that, in any assignment, drivers are experiencing a travel time that is very similar to the fastest one on the actual road network and it is particularly important from the users point of view since the assignment is almost "envy-free", i.e. no user will complain because they could be routed on faster route.

We added complementary information in Figure 7(b) where the maximum experienced fastest path unfairness over all group A instances is shown as a function of γ^{FP}. This measure gives us an idea of the worst-case situation for an user. The maximum value is sensibly lower than the maximum value allowed. For any reported γ^{FP}, the maximum experienced fastest path unfairness is always under 5%. In the set of tested instances, the maximum experienced fastest path unfairness decreases when γ^{FP} increase to 8%. This due to the fact that the objective function does not involve unfairness and, hence, unfairness could decrease even if a greater value of γ^{FP} is considered. In fact, this phenomenon vanishes when results are averaged over 32 instances but it happens frequently with different γ^{FP} values. In particular, two of the 32 instances are responsible for the peak observed when $\gamma^{FP} = 8\%$. For the sake of completeness, in Figure 7(c), we report also the standard deviation of the experienced fastest path unfairness. The standard deviation shows that, on average, fastest path unfairness values are almost allocated near the average value in the distribution. We do not report the value of loaded unfairness since the behaviour is very similar to the fastest path unfairness. This is due to the fact that fastest path on the current network is almost always used. There a very few cases in which the fastest path is not used but still it happens for some OD pairs.

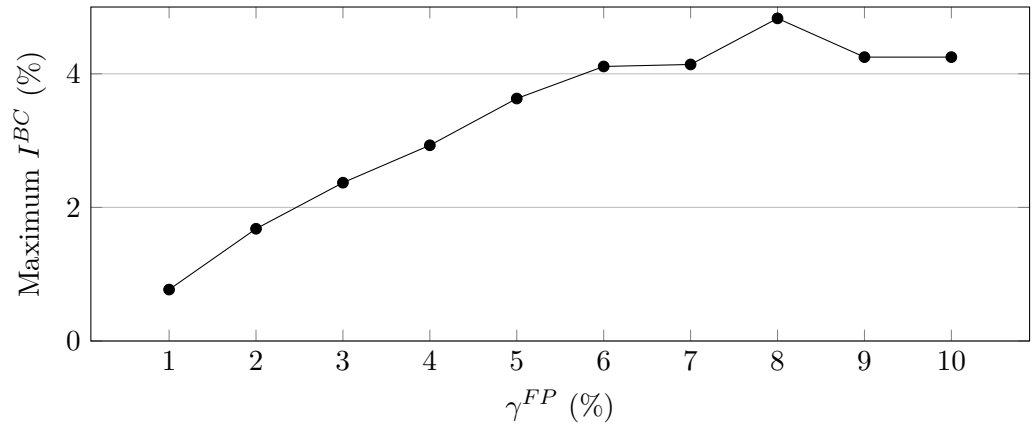
4.3 PC-M *algorithm*

To assess the computational benefits of the PC-M algorithm, the computational time of the UC-SO model, the run time of PC-M, and the optimality gap Θ (average and maximum value), for different γ^{FP} values, are shown in Table 2, where the values are averaged over the 32 instances of Group A.

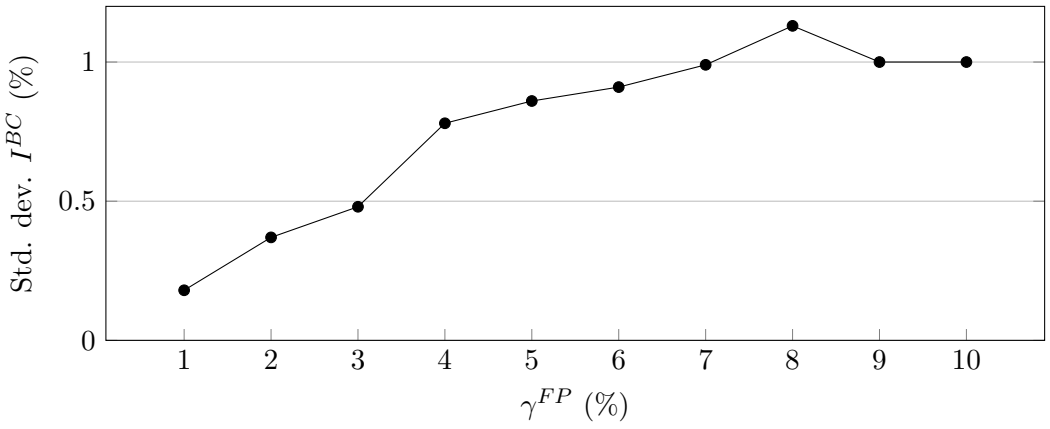
Since the number of paths generated in the UC-SO model is the same considering different γ^{FP} values (we are generating all possible paths from origin to destination), the additional run time needed to solve the UC-SO model is due to the fact that more restrictive constraints are used considering lower γ^{FP} values. In fact, the separation procedure, implemented to dynamically add constraints, adds more constraints when γ^{FP} decreases because it will detect more violations because of the lower value of γFP. On the other hand, the computational times of PC-M are almost steady. Moreover, in all cases PC-M computational time is two orders of magnitude less than the computational time needed to solve the UC-SO model. In addition, the PC-M algorithm is also very accurate. In order to show the accuracy, in Table 2, the average optimality gap along



(a) Average I^{BC}



(b) Maximum I^{BC}



(c) Standard deviation I^{BC}

Figure 7. fastest path unfairness, I^{BC}

Table 2. UC-SO run time (sec.), run time (sec.), average optimality gap (%) and maximum optimality gap (%) for PC-M on Group A instances.

γ^{FP}	UC-SO run time (sec)	PC-M run time	Θ Avg. gap (%)	Θ Max. gap (%)
1	563	1.61	0.22	0.63
2	321	1.39	0.24	1.55
3	176	1.30	0.30	1.30
4	153	1.27	0.25	1.00
5	151	1.30	0.22	0.96
6	132	1.28	0.21	0.88
7	115	1.30	0.20	0.90
8	106	1.28	0.19	0.90
9	99	1.28	0.19	0.90
10	101	1.30	0.19	0.90

with its maximum value is shown for each γ^{FP} value. The average optimality gap is very small as it is always lower than 0.30%. Furthermore, the maximum value is, in the worst case, lower than 1.55%. Along with optimality gaps, we have also computed the number of paths generated by the PC-M algorithm and the number of paths needed to solve the UC-SO model. On average, the number of paths generated for the UC-SO model is 32000 while the number of paths generated by the PC-M algorithm is, on average, 35 which means that the PC-M algorithm performs better also in terms of memory consumption (three orders of magnitude).

In Table 3, results for the 4 Group B increasing size instances are shown. Here results for the UC-SO model are missing since the solver did not return any result, either because the path generation procedure ran out of memory or the computational time exceeded the limit of 14400 seconds. In Table 3, the computational time needed to run the PC-M algorithm is shown. As for group A instances, computational time is usually higher when small values of parameter γ^{FP} are considered. Furthermore, we observe that computational time is on average around 300 seconds (5 minutes) even considering the largest instance that makes the PC-M algorithm computationally valuable in solving real-world instances. Moreover, the average fastest path unfairness is very low and far from being equal to the maximum percentage allowed. Results in Table 3 are obtained by using $n = 1000$ as accuracy parameter.

Table 3. PC-M results for group B instances.

	Stats	γ^{FP} 1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
270 nodes	Time PC-M (sec)	394	361	681	246	194	194	146	146	96	96
	$\Theta^H(10^8 sec)$	6.1548	6.1507	6.1493	6.1492	6.1492	6.1492	6.1492	6.1492	6.1492	6.1492
	Average I^{BC} (%)	0.01	0.05	0.09	0.11	0.11	0.11	0.11	0.12	0.12	0.12
	Maximum I^{BC} (%)	1.0	1.97	2.98	3.78	4.97	5.77	5.76	8.97	8.06	8.06
300 nodes	Time PC-M (sec)	576	602	456	264	324	200	143	140	138	141
	$\Theta^H(10^8 sec)$	6.8565	6.8468	6.8419	6.8419	6.8418	6.8418	6.8418	6.8418	6.8418	6.8418
	Average I^{BC} (%)	0.01	0.09	0.15	0.17	0.18	0.19	0.19	0.19	0.19	0.19
	Maximum I^{BC} (%)	1.0	2.0	3.0	3.92	4.63	5.58	6.06	6.06	6.06	6.06
330 nodes	Time PC-M (sec)	1281	1065	357	352	354	349	270	193	203	205
	$\Theta^H(10^8 sec)$	6.64	6.62	6.62	6.62	6.62	6.62	6.62	6.62	6.62	6.62
	Average I^{BC} (%)	0.05	0.14	0.2	0.23	0.25	0.25	0.25	0.27	0.27	0.27
	Maximum I^{BC} (%)	1.0	2.0	3.0	3.99	4.74	5.86	6.43	7.7	7.7	7.7
360 nodes	Time PC-M (sec)	542	261	282	136	540	97	70	86	85	84
	$\Theta^H(10^8 sec)$	6.3305	6.3137	6.3033	6.2958	6.3047	6.2908	6.2866	6.2862	6.2853	6.2846
	Average I^{BC} (%)	0.05	0.1	0.2	0.27	0.31	0.4	0.52	0.55	0.6	0.68
	Maximum I^{BC} (%)	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0

In Table 4 results using the PC-M algorithm on two real world instances (with number of nodes 224 and 416) are shown for γ^{FP} values ranging from 1% to 10%. These results were obtained using accuracy level $n = 100$ for PC-M in order to speed up the algorithm. On all experiments made, we have observed that using either $n = 1000$ or $n = 100$ is statically indifferent and, hence, in solving big instances we decided to reduce computational times using the smallest value. All collected statistics follow the same behaviour we have observed in group A and B instances. In particular, the lower parameter γ^{FP} is, the higher is the computational time needed to obtain a solution. Here the highest computational time observed is around 2 minutes.

Table 4. PC-M results on real-life instances.

	Berlin-Friedrichshain Nodes = 224, Arcs= 523, OD pairs= 506			
γ^{FP} (%)	Time PC-M (sec)	$\Theta^H(10^5 sec)$	Average I^{BC} (%)	Maximum I^{BC}
1	3.95	8.2941	0.04	1.0
2	4.3	8.2941	0.04	1.4
3	2.3	8.2855	0.06	1.4
4	2.4	8.2855	0.06	3.1
5	2.4	8.2855	0.07	3.1
6	2.4	8.2855	0.07	5.0
7	2.4	8.2855	0.07	5.0
8	2.4	8.2855	0.07	8.0
9	2.4	8.2855	0.07	9.0
10	2.3	8.2855	0.07	9.0
	Anaheim Nodes = 416, Arcs= 914, OD pairs= 1374			
γ^{FP} (%)	Time PC-M (sec)	$\Theta^H(10^7 sec)$	Average I^{BC} (%)	Maximum I^{BC}
1	115.2	8.0859	0.04	1.0
2	51.8	8.0818	0.09	2.0
3	30.1	8.0687	0.14	3.0
4	35.7	8.0563	0.21	4.0
5	21.4	8.0351	0.24	5.0
6	20.1	8.0626	0.21	6.0
7	30.1	8.0525	0.29	7.0
8	16.7	8.0314	0.28	8.0
9	16.9	8.022	0.42	9.0
10	17.2	8.013	0.43	10.0

5 Conclusions and future research

The user time constrained system optimum model proposed in this paper provides a powerful tool in controlling the user unfairness while implementing a system optimum traffic assignment. The UC-SO model produces a traffic assignment with a total travel time that is very close to the total travel time of the system optimum assignment. Moreover, user unfairness is kept low enough to guarantee a complete compliance to the system guidance. This is because users do not feel to be disadvantage with respect to the others and to the current network status. As pointed out in the introduction, the steady-state assumption, which allows us to work with static traffic assignments, may be reasonable during rush hour periods, but will not be appropriate for a period in which the demand varies over time. To accommodate varying demand over time, a time-dependent variant of the UC-SO model has to be investigated. One big issue in dealing with vehicular traffic is congestion avoidance when unexpected events occurs such as bad weather conditions, car crashes and diversions due to work in progress. In order to study this case a dynamic framework has to be studied in which the decision for drivers has to be taken at each node and the assignment can change over time.

References

E. Angelelli, I. Arsik, V. Morandi, M. Savelsbergh, and M. Speranza. Proactive route guidance to avoid congestion. *Transportation Research Part B: Methodological*, 94:1 – 21, 2016a. doi: http://dx.doi.org/10.1016/j.trb.2016.08.015.

E. Angelelli, V. Morandi, M. Savelsbergh, and M. G. Speranza. System optimal routing of traffic flows with user constraints using linear programming. Technical Report 5, University of Brescia, Department of Economics and Management, 2016b. URL http://www.unibs.it/dipartimenti/economia-e-management/ricerca/working-paper.

E. Angelelli, V. Morandi, and M. G. Speranza. Heuristic path generation for the proactive route guidance problem. Technical Report 3, University of Brescia, Department of Economics and Management, 2016c. URL http://www.unibs.it/dipartimenti/economia-e-management/ricerca/working-paper.

E. Angelelli, V. Morandi, and M. G. Speranza. A trade-off between average and maximum arc congestion minimization in traffic assignment with user constraints. Technical Report 3, University of Brescia, Department of Economics and Management, 2018. URL http://www.unibs.it/dipartimenti/economia-e-management/ricerca/working-paper.

M. E. Ben-Akiva and S. R. Lerman. *Discrete choice analysis: theory and application to travel demand*. MIT press, 1985.

J. R. Correa, A. S. Schulz, and N. E. Stier-Moses. Fast, fair, and efficient flows in networks. *Operations Research*, 55:215–225, 2007.

J. de Dios Ortuzar and L. G. Willumsen. *Modelling transport*. John Wiley & Sons, 2011.

M. Florian and D. Hearn. Network equilibrium and pricing. In *Handbook of Transportation Science*, pages 361–393. Springer, 1999.

O. Jahn, R. H. Möhring, and A. S. Schulz. Optimal routing of traffic flows with length restrictions in networks with congestion. In *Operations Research Proceedings 1999*, pages 437–442. Springer, 2000.

O. Jahn, R. H. Möhring, A. S. Schulz, and N. E. Stier-Moses. System-optimal routing of traffic flows with user constraints in networks with congestion. *Operations Research*, 53:600–616, 2005.

M. Lujak, S. Giordani, and S. Ossowski. Route guidance: Bridging system and user optimization in traffic assignment. *Neurocomputing*, 151:449–460, 2015.

H. S. Mahmassani and S. Peeta. Network performance under system optimal and user equilibrium dynamic assignments: implications for advanced traveler information systems. *Transportation Research Record*, 1993.

A. S. Schulz and N. E. Stier-Moses. Efficiency and fairness of system-optimal routing with user constraints. *Networks*, 48:223–234, 2006.

Y. Sheffi. Urban transportation network. *Equilibrium analysis with mathematical programming methods, Prentice Hall*, 1985.

B. Stabler. Transportation networks for research github repository. https://github.com/bstabler/TransportationNetworks.git, 2018.

J. G. Wardrop. Road paper. some theoretical aspects of road traffic research. *Proceedings of the institution of civil engineers*, 1:325–362, 1952.