Optimization of a one-way carsharing system with relocation operations

Aurélien Carlier, Alix Munier-Kordon, Witold Klaudel

Optimization of a one-way carsharing system with relocation operations

Aurélien CARLIER
Renault SAS, Technocentre,[1]
Technological Research Institute SystemX,[2]
[1] Guyancourt, [2] Palaiseau, France
Email: aurelien.carlier@renault.com
aurelien.carlier@irt-systemx.fr

Alix MUNIER KORDON
Sorbonne Universités,
UPMC Univ Paris 6,
UMR 7606, LIP6,
F-75005, France
Email: alix.munier@lip6.fr

Witold KLAUDEL
Renault SAS,
Guyancourt, France
Email: witold.klaudel@renault.com

***Abstract*—Carsharing systems have gathered increasing attention and are now recognised as innovative and ecological solutions to transportation issues. Especially, one-way carsharing systems offer a high level of service because they exempt users from the obligation to return the vehicle at the same station where it was borrowed. Unfortunately, this flexibility comes with design complexity. This paper addresses a part of those design issues and deals with fleet dimensioning including vehicle relocation operations.**

We propose mathematical programming oriented approach and we introduce a simple linear model based on integer flow variables. This model is organized around three optimization criteria: maximizing satisfied carsharing demands while minimizing the fleet of vehicles and the relocation operations. From random generated benchmarks, we study the scalability of our method and we show that computation times remain acceptable for representative problem complexities.

***Index Terms*—one-way carsharing, operational research, carsharing demand generation, transport optimization, mathematical modelling.**

I. INTRODUCTION

In the last decade, the importance of adapting the way we move in our societies has been highlighted by a lot of researches. The greater accessibility to private car has more particularly resulted in serious negative externalities, although it significantly improved people mobility in urban areas. Pollution, excessive energy consumption and wasted time evolve largely from the road network congestion. In many situations, current public transport does not offer a valuable alternative to the private car and a lot of efforts are made to find a different and acceptable solution [1]. One of them is carsharing.

A carsharing system involves a small to medium fleet of vehicles, available at several stations, to be used by a relatively large group of members [2]. This service constitutes a real alternative to private car because it releases the user from constraints related to individual property like insurance, maintenance, fuel (or electricity), taxes, vehicle depreciation, etc. Such systems became popular in the last couple of decades and nowadays it is not unusual to find one in most of the world's major cities in industrialized countries. Their numerous benefits fostered several researches on environmental impacts, social characterization of the users, understanding of urban trip patterns, demand modelling, vehicle imbalance problem, operator relocation operations policy, optimal system dimensioning, etc. Jorge and Correia [3] give a review of these topics.

Initially, in the so called *round-trip* carsharing systems, the users have to pick up and return the vehicles at the same place. From an operational point of view, this limitation simplifies greatly the dimensioning and the management of the vehicles fleet. In this paper, we will focus on *one-way* carsharing systems which allows users to pick up a vehicle from a station and return it at a different one. The greater flexibility of these systems helps capture more requested trips (referred further as *demands*) [4] but comes also with serious operational issues, especially those from unbalanced vehicle distribution. As a consequence, the dimensioning of a one-way carsharing system must include vehicle relocations to capture and meet real demand as much as possible. Relocation is often not considered in car rental services [5], [6] since the price of one-way trips is usually fully supported by the customers.

Some studies have been conducted to find optimal vehicle relocations when a system already exists (*e.g.* [7], [8], [9], [10]), while others searched for optimal dimensioning of the vehicle fleet taking into account those relocation operations. In most cases, mathematical programming is considered as a good and pertinent approach to describe and solve those problems. Fan *et al.* [11] proposed for example a multi-stage stochastic linear integer model accounting for demand variations and indicating operator profit improvement. More recently, Correia and Antunes [12] developed three mathematical programming models to balance vehicle stocks while dimensioning the whole system in terms of location, number and size of stations. The objective was again maximizing the profit for the operator considering all the revenues and costs involved.

Although mathematical programming is an effective and suitable approach to deal with the optimization of carsharing systems, the studies mentioned above often underline limitations due to computation time and solver capability. This frequently led to model simplifications which makes results unrealistic. Thus, our first aim is to develop a simple realistic and scalable flow based model taking into account relocation

operations. The scalability is confirmed by the acceptable computation times for the large instances we analysed.

Three optimization criteria are considered: maximizing the total number of satisfied demands and minimizing the total number of vehicles and relocation operations. We experimentally proved that the two last criteria are in opposite and that there exists a tradeoff area between them.

To the best of our knowledge, there is no available benchmark for testing our model. Most authors use real data given by a carsharing operator or extrapolated from real life observations. Hence, we developed a random data generator taking into account the demand variation over the course of a day.

This paper is organised as follow. Section II formulates on the one hand the problem in mathematical terms, and proves on the other hand that the number of vehicles can be inferred from any solution. Section III is dedicated to numerical experiments based on randomly generated data. Results specifically focus on computation times, 3-dimensional Pareto frontiers and scalability. Finally, section IV concludes the paper with directions for further research.

II. Problem Modeling

This section aims at modelling our optimization problem using an integer linear program. The inputs and the outputs of the problem are first described. Second subsection is dedicated to the building of an oriented valued graph, namely the Timed Extended Graph. This graph, previously introduced by Ahuja *et al.* [13], allows to express all the constraints of the problem following the time and the space dimensions. The third subsection introduces the decision variables of our optimization problem. More precisely, it is shown that vehicles can be equivalently aggregated into flows to express all the constraints and the criteria of our optimization problem. Last subsection presents its formulation using integer linear programming.

A. Inputs and outputs of the problem

In order to describe the mathematical problem, let us define first the required inputs. As discussed later, those data can be extracted from various sources such as simulation tools or real operating carsharing system data for example.

First of all, let $\mathcal{H} = \{1, \cdots, T\}$ be the set of time-steps considered in the study. For relevance issues, it has to cover a representative period of time, as an average weekday or an average week for instance.

The set of carsharing stations is defined as $\mathcal{N} = \{1, \cdots, N\}$ and comes with $Z(i)$ the maximum size (*i.e.* maximum capacity, in terms of number of vehicles) of station $i \in \mathcal{N}$. Then, the demand $D(i, j, t)$ contains the number of passengers wishing to borrow a car from station $i \in \mathcal{N}$ at time $t \in \mathcal{H}$ to station $j \in \mathcal{N}$. Note here that there is no condition on the station themselves, thus allowing the travel to be a "round trip" or a "one-way" travel. In this case, i is simply equal to j.

Finally, $\delta(i, j, t)$ represents the travel time it takes for a car-user from station $i \in \mathcal{N}$ to station $j \in \mathcal{N}$, when departure time from i is $t \in \mathcal{H}$. We suppose that for any triple $(i, j, t) \in \mathcal{N} \times \mathcal{N} \times \mathcal{H}$, $\delta(i, j, t) < T$. This assumption comes from the fact that the highest distance from different stations is quite low (usually less than 150 or 200 kilometres) and that the time-steps are covering at least a day.

A feasible solution of our problem consists on a set of vehicle tours, each of them modelling the situation of a car at each time step. The three criteria (the demand, the number of vehicles and the number of relocation operations) can be polynomially computed from any feasible solution.

B. Time Extended Graph

To deal with discrete-time dynamic networks, Ahuja *et al.* [13] suggested the use of *time-space network*, also known as *Time Extended Graphs* (TEG). It is a static network constructed by expanding the original network in the time dimension, considering a separate copy of every node $i \in \mathcal{N}$ at every discrete time-step $t \in \mathcal{H}$. Thus, from the data listed above, let us consider $G = (\mathcal{X}, \mathcal{A}, u)$ as a TEG defined as follows:

1) Set of nodes: Nodes are couples (i, t) with $i \in \mathcal{N}$ and $t \in \mathcal{H}$ associated with station i at time t. Formally, $\mathcal{X} = \mathcal{N} \times \mathcal{H}$.

Let η and θ the functions which reciprocally return the station and the time-step associated with every element of $\mathcal{X}$. More formally,

$$\eta : \mathcal{X} \to \mathcal{N} \text{ with } x = (i, t) \mapsto \eta(x) = i$$

$$\theta : \mathcal{X} \to \mathcal{H} \text{ with } x = (i, t) \mapsto \theta(x) = t$$

2) Set of arcs: Edges are partitioned into 3 sets $\mathcal{A}_1$, $\mathcal{A}_2$ and $\mathcal{A}_3$ defined as follows:

- $\mathcal{A}_1$ is the set of arcs representing the possibility for the vehicles to stay at a station between two consecutive time-steps. Formally, $\mathcal{A}_1 = \{(x, y) \in \mathcal{X} \times \mathcal{X} \mid \eta(x) = \eta(y) \text{ and } \theta(y) = \theta(x) + 1 \bmod T\}$.
- $\mathcal{A}_2$ are the arcs associated with a demand: each demand $D(i, j, t) > 0$ corresponds to an arc (x, y) with $x = (i, t)$ and $y = (j, t + \delta(i, j, t))$. Set $\mathcal{A}_2$ is then formally defined as $\mathcal{A}_2 = \{(x, y) \in \mathcal{X} \times \mathcal{X} \mid D(\eta(x), \eta(y), \theta(x)) \neq 0 \text{ and } \theta(x) + \delta(\eta(x), \eta(y), \theta(x)) = \theta(y) \bmod T\}$.
- $\mathcal{A}_3$ represents all the possible relocation operations over time. Any triple $(i, j, t) \in \mathcal{N} \times \mathcal{N} \times \mathcal{H}$ with $i \neq j$ is associated to an arc from $x = (i, t)$ to $y = (j, t + \delta(i, j, t) \bmod T)$. Thus, $\mathcal{A}_3 = \{(x, y) \in \mathcal{X} \times \mathcal{X} \mid \eta(x) \neq \eta(y) \text{ and } \theta(x) + \delta(\eta(x), \eta(y), \theta(x)) = \theta(y) \bmod T\}$.

The total number of arcs is given by:

$$|\mathcal{A}| = |\mathcal{A}_1| + |\mathcal{A}_2| + |\mathcal{A}_3| = N \times T + M + (N \times T) \cdot (N - 1)$$

where M is the number of requested demands. As $M \ll N^2$, $|\mathcal{A}| = \Theta(N^2 \cdot T)$.

3) Arcs Capacity: Associated with each arc $a = (x, y) \in \mathcal{A}$, is given a capacity function $u : \mathcal{A} \to \mathbb{N}$ corresponding to a maximum number of vehicles allowed on a. It is defined as follows for any arc $a \in \mathcal{A}$:

$$u(a = (x, y)) = \begin{cases} Z(\eta(x)) & \text{if } a \in \mathcal{A}_1 \\ D(\eta(x), \eta(y), \theta(x)) & \text{if } a \in \mathcal{A}_2 \\ +\infty & \text{if } a \in \mathcal{A}_3 \end{cases}$$

For any arc $a = (x, y) \in \mathcal{A}_1$, the maximum number of cars is the capacity of the station $\eta(x) = \eta(y)$. It corresponds to the demand for any arc $a \in \mathcal{A}_2$, and it is not bounded for relocation arcs $a \in \mathcal{A}_3$.

4) Additional notations: We denote by $\Gamma^-(x)$ and $\Gamma^+(x)$ respectively the set of immediate predecessors and successors of a node $x \in \mathcal{X}$, *i.e.*

$$\Gamma^-(x) = \{y \in \mathcal{X} \mid (y, x) \in \mathcal{A}\}$$
$$\Gamma^+(x) = \{y \in \mathcal{X} \mid (x, y) \in \mathcal{A}\}$$

For any couple of time instants $\forall (t, t') \in \mathcal{H}^2$, the number of time-steps between those two instants is defined through the following function

$$\begin{array}{rcl} \vartheta : \mathcal{H}^2 & \to & \mathbb{N} \\ (t, t') & \mapsto & \vartheta(t, t') \end{array}$$

$$\text{with } \vartheta(t, t') = \begin{cases} t' - t & \text{if } t \leq t' \\ T + t' - t & \text{otherwise.} \end{cases}$$

For each arc $a = (x, y) \in \mathcal{A}$, let us define the boolean value ϵ_a as:

$$\epsilon_a = \begin{cases} 0 & \text{if } \theta(x) \leq \theta(y) \\ 1 & \text{otherwise} \end{cases}$$

The time required for a movement from x to y is then given by the function

$$\begin{array}{rcl} \ell : \mathcal{A} & \to & \mathbb{N} \\ a = (x, y) & \mapsto & \ell(a) = \theta(y) - \theta(x) + \epsilon_a \cdot T \end{array}$$

By extension, if $\mu = (a_1, \cdots, a_p) \in \mathcal{A}^p$ is a path of the TEG from x to y, the value $\ell(\mu) = \sum_{i=1}^{p} \ell(a_i)$ is the total time required for a vehicle going from x to y following μ.

For any time value $t \in \mathcal{H}$, let us define the set $\mathcal{C}_t(\mu)$ as the arcs $a = (x, y)$ from μ starting at time t or earlier but ending after t. Formally, $\mathcal{C}_t(\mu) = \{a = (x, y) \in \mu \mid \vartheta(\theta(x), t) < \ell(a)\}$.

C. Decision variables

The aim of our study is to compute the planning of each vehicles during the period. At any time, each vehicle is either parked in a station or in transit between two stations. Its position over the period can be modelled as a vehicle tour *i.e.* a circuit $c = (a_1, \cdots, a_p)$ in the TEG.

The size of any feasible solution may be highly reduced if we only consider the number of vehicles passing through each arc. For each arc $a = (x, y) \in \mathcal{A}$, we call $\varphi(a)$ the flow of vehicles transiting through the arc a. It can be interpreted as:

- the number of vehicle staying in station $\eta(x)$ between two consecutive time-steps $\theta(x)$ and $\theta(y)$, if $a \in \mathcal{A}_1$;
- the number of vehicle picked by users from station $\eta(x)$ at time $\theta(x)$ to station $\eta(y)$, if $a \in \mathcal{A}_2$;
- the number of vehicle relocated between stations $\eta(x)$ and $\eta(y)$ at time $\theta(x)$, if $a \in \mathcal{A}_3$.

The total number of vehicles transiting to any node $x \in \mathcal{X}$ is clearly constant, thus

$$\sum_{y \in \Gamma^-(x)} \varphi((y, x)) = \sum_{y \in \Gamma^+(x)} \varphi((x, y)).$$

It is immediate that a feasible flow may be obtained from any feasible set of vehicle tours. We prove in the following that vehicle tours may be easily computed from a feasible flow.

Next lemmas characterize the total time of any circuit c and the exact number of vehicles required for a unitary flow on c.

Lemma 1: The total time of any circuit $c = (a_1, \cdots, a_p)$ is $\ell(c) = T \times \sum_{i=1}^{p} \epsilon_{a_i}$.

Proof. Let x_i, $i \in \{1, \cdots, p+1\}$ be the sequence of elements from $\mathcal{X}$ such that, $x_{p+1} = x_1$ and $\forall i \in \{1, \cdots, p\}$, $a_i = (x_i, x_{i+1})$. The total time of c is then

$$\begin{array}{rl} \ell(c) & = \sum_{i=1}^{p} \ell(a_i) \\ & = \sum_{i=1}^{p} (\theta(x_{i+1}) - \theta(x_i) + \epsilon_{a_i} \cdot T) \\ & = T \times \sum_{i=1}^{p} \epsilon_{a_i}. \end{array}$$

the result. ■

Lemma 2: Let c be a circuit and φ_c a feasible flow such that

$$\varphi_c(a) = \begin{cases} 1 & \text{if } a \text{ belongs to } c \\ 0 & \text{otherwise} \end{cases}$$

The minimum number of vehicles to insure φ_c is $\frac{\ell(c)}{T}$.

Proof. The total number of vehicles needed at time $t \in \mathcal{H}$ for c is clearly $|\mathcal{C}_t(c)|$.

We first show that $|\mathcal{C}_T(c)| = \sum_{a \in c} \epsilon_a$. For that purpose, setting $B(c) = \{a = (x, y) \in c \mid \epsilon_a = 1\}$, we prove that $B(c) = \mathcal{C}_T(c)$.

- $B(c) \subseteq \mathcal{C}_T(c)$: If $a = (x, y) \in B(c)$, then as $\theta(x) \leq T$, $\vartheta(\theta(x), T) = T - \theta(x)$. Now, since $\epsilon_a = 1$, $\ell(a) = \theta(y) - \theta(x) + T \geq \vartheta(\theta(x), T)$ and $a \in \mathcal{C}_T(c)$.
- $\mathcal{C}_T(c) \subseteq B(c)$: Let consider now an arc $a = (x, y) \in \mathcal{C}_T(c)$. Since $\vartheta(\theta(x), T) = T - \theta(x) < \ell(a)$, we get $\theta(y) + \epsilon_a \cdot T > T$. As $\theta(y) \leq T$, we necessarily have $\epsilon_a = 1$ and thus $a \in B(c)$.

Now, by Lemma 1, $|\mathcal{C}_T(c)| = \sum_{a \in c} \epsilon_a = \frac{\ell(c)}{T}$. Lastly, the minimum number of vehicles to insure φ_c is constant over $\mathcal{H}$ and thus $\forall t \in \mathcal{H}$, the lemma. ■

Theorem 1: Any feasible solution φ can be decomposed into a set of circuits $\mathcal{S}$ such that, for any arc $a \in G$, $\varphi(a) = \sum_{c \in \mathcal{S}} \varphi_c(a)$.

Proof. The proof is by recurrence on $n(\varphi) = \sum_{a \in \mathcal{A}} \varphi(a)$. The theorem is trivially true if $n(\varphi) = 0$.

Let suppose now that $n(\varphi) > 0$, thus there exists at least one arc $a = (x, y)$ with $\varphi(a) > 0$. Set $\mu_0 = (x, y)$ and let consider the sequence of paths μ_i built as follows:

1) Stop the sequence as soon as μ_i contains a circuit c;
2) Otherwise, let $\tilde{a} = (\tilde{x}, \tilde{y})$ the last arc of μ_i. Since $\varphi(\tilde{a}) > 0$, following the conservation low of φ over nodes of G, there exists an arc a starting at $\tilde{y}$ with $\varphi(a) > 0$. We then set $\mu_{i+1} = \mu_i \cdot a$.

As G has a finite number of nodes, the algorithm stops and a non empty circuit is then returned. Let now define the flow $\hat{\varphi}$ as follows

$$\hat{\varphi}(a) = \begin{cases} \varphi(a) - 1 & \text{if } a \in c \\ \varphi(a) & \text{otherwise} \end{cases}$$

$\hat{\varphi}$ is feasible with $n(\hat{\varphi}) < n(\varphi)$, thus the theorem. ■

Note that the number of flow variables is a polynomial function on the size of the problem. It is not the case for the vehicle tours, which number can be of exponential size. The consequence is that the determination of a flow is in $\mathcal{NP}$, which is not the case for the determination of vehicle tours.

D. Modelling of the optimization problem

Three objectives are to be considered for our optimization problem: the main objective is to maximize the number of demands. The two other ones are to minimize both the number of relocations and the total number of vehicles.

As we shall see later, the number of demands and relocations are easily linearly expressed using the flows. Next theorem shows that it is also the case for the total number of vehicles:

Theorem 2: The minimum total number of cars required for a feasible flow φ equals $\sum_{a\in\mathcal{A}} \varphi(a) \cdot \epsilon_a$.

Proof. Let $\mathcal{S}$ be a set of circuits obtained from the decomposition of φ following Theorem 1 and let V be the minimum number of cars associated with φ. By Lemmas 1 and 2, the total number of car of any circuit $c \in \mathcal{S}$ is

$$\sum_{a\in c} \epsilon_a = \sum_{a\in\mathcal{A}} \epsilon_a \cdot \varphi_c(a)$$

and thus

$$V \leq \sum_{c\in\mathcal{S}} \sum_{a\in\mathcal{A}} \epsilon_a \cdot \varphi_c(a).$$

Now, from Theorem 1, $\varphi(a) = \sum_{c\in\mathcal{S}} \varphi_c(a)$. Thus,

$$\sum_{c\in\mathcal{S}} \sum_{a\in\mathcal{A}} \epsilon_a \cdot \varphi_c(a) = \sum_{a\in\mathcal{A}} \epsilon_a \cdot \sum_{c\in\mathcal{S}} \varphi_c(a) = \sum_{a\in\mathcal{A}} \varphi(a) \cdot \epsilon_a.$$

Lastly, the total number of vehicles required at time T to reach φ is exactly $\sum_{a\in\mathcal{A}} \varphi(a) \cdot \epsilon_a$, the theorem. ■

The modelling of our optimization problem follows. R and C are fixed bounds for respectively the total number of relocation operations and vehicles. Equation (1) is the maximization of the total demand. Equation (2) expresses the bound on the total number of relocation. Equation (3) expresses these on the total number of vehicles. Equations (4), (5) and (6) are lastly flow constraints.

$$\max \quad \sum_{a\in\mathcal{A}_2} \varphi(a) \tag{1}$$

$$s.t. \begin{cases} \displaystyle\sum_{a\in\mathcal{A}_3} \varphi(a) \leq R & (2) \\ \displaystyle\sum_{a\in\mathcal{A}} \varphi(a) \cdot \epsilon_a \leq C & (3) \\ \varphi(a) \leq u(a) & \forall a \in \mathcal{A} \quad (4) \\ \displaystyle\sum_{y\in\Gamma^-(x)} \varphi((y,x)) = \sum_{y\in\Gamma^+(x)} \varphi((x,y)) & \forall x \in \mathcal{X} \quad (5) \\ \varphi(a) \in \mathbb{N} & \forall a \in \mathcal{A} \quad (6) \end{cases}$$

The total number of equations is around $2|\mathcal{A}| + N \times T = \Theta(N^2 \cdot T)$.

III. Experimentations / results

In order to test the linear mathematical model, we first need data. Unfortunately, we do not yet have enough relevant and actual data that could be used for study purposes. To cope with that, we propose to generate data using a random generator described in the first subsection. Then, the second subsection gives some discussion about computation times and solutions analysis of small instances, based on generated data. Three-dimensional Pareto frontier is especially presented. Finally, third subsection is dedicated to scalability experimentations, particularly the solver's performance when the problem increases in size.

A. Random data generator

A generator has been then implemented to emulate real demand data over time. We decided to focus our study on representing an average weekday demand. As a consequence, all the data depending on time are generated over a 24 hours period, segmented into T time-steps. The total number of time-steps is user-settable and can vary from 24 to 1440 (representing respectively a 60 and 1 min time-step).

Basically, the generator is based on two phases: station and demand generation. The first phase positions N carsharing stations within a given territory. Maximum size for each station is randomly generated using a discrete uniform distribution over an integer interval $[Z_{\min}, Z_{\max}]$ given by the user as a parameter. The station positioning is made over two distinct zones: a central area (in general representing the center of the city) included in a larger one (representing the suburbs area), both defined as a square. The generation algorithm takes two additional parameters: the percentage of total area the center must represent and the probability that a station is contained in the center. Once the geographic division is made, every station is then positioned randomly in the area where it belongs.

Then, the second phase generates randomly M demands over time between stations. First of all, the generator has to schedule and position randomly each request over time, which means defining a probability distribution. In order to do so, the generator allows to specify the distribution profile the demand will follow. In other words, it consists on defining, for every couple of hours (thus 12 values), the relative level of demand

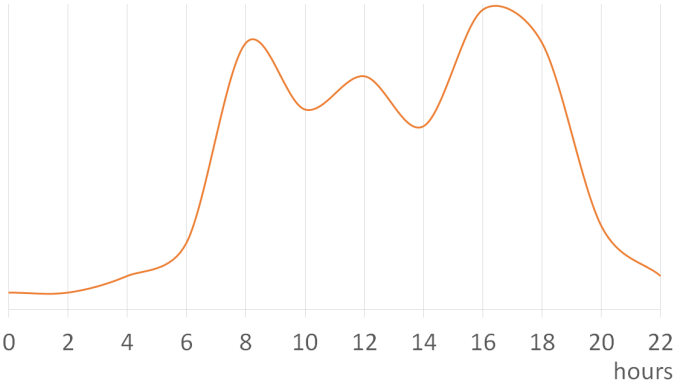

Fig. 1. Classical average demand profile in dense areas over the course of a weekday

TABLE I
COMPUTATION TIMES IN SECONDS

	μ	σ	min	max
LP	$0,56$	$0,25$	$< 0,01$	$1,17$
ILP	$1,93$	$1,86$	$< 0,01$	$16,77$

$Dem(t)$ at this time t. Then, the probability distribution $\mathcal{P}$ is obtained normalizing all the values *i.e.* $\mathcal{P}(t) = Dem(t)/\sum_t Dem(t)$. Typically, most profile distributions are very similar to the one presented in Figure 1. Now, the next step is to identify the two stations concerned by this demand at that time: where it starts and where it goes. Usually, in dense urban area, there are two rush hour slots, also called traffic peaks (generally the morning between 7 and 10 o'clock; the evening between 16 and 19 o'clock) for which the demand goes globally in the same direction: from the suburbs to the center during the morning and from the center to the suburbs the evening. The generator allows to define such rush time slots as well as the proportion of total demand during those rushes. Finally, it's also possible to specify an average car speed and a penalty coefficient during rush time for the calculation of travel-time between stations.

B. Experimentations

1) Experimental context: All experimental results addressed bellow were made using an Intel(R) Core(TM) i3-3227U CPU running at 1.9GHz. The linear programs were expressed using the AMPL format [14] and numerical results were obtained using GLPK v4.52 [15].

2) Experimentation of small instances: First of all, we start our experimentations considering $N = 10$ stations, $T = 144$ time-steps and $M = 500$ demands. The upper bound of relocation operations and the number of vehicles belong to the set $\{0, 10, \cdots, 80\}$. For those fixed values, 30 instances were generated using the random generator previously described for studding the computation time of the optimal solutions and the impact of model's building time.

Table I presents the average computation time μ and the standard deviation σ for both integer linear program ILP and its relaxation to linear program LP. The minimal and maximal values are also specified. The first observation is that computation times remains quite low, in the order of a half-second for LPs and two seconds for ILPs. The major part is taken by the building of the mathematical program which takes 34s (mostly by the conservation law) regardless of which model is built. Secondly, when the solver runs the LP model, almost 8 problems (exactly $7,66$ on average) under 81 admit a non-integer value of the objective function, over the 30 generated instances. This result represents almost 10% of all instances. However, every time we get a non-integer optimal value, the integer one has always the same integer part. Hence it comes that we conjecture the existence of an integer optimal solution for LP and that the difference between the two criteria values comes from rounding errors.

Figure 2 shows a 3-dimensional Pareto frontier for a particular instance. When all the demands are satisfied, the minimal number of vehicles equals 50. In this case, at least 70 relocation operations are necessary. If the number of vehicle increases to 70, 50 relocations are needed. This confirms that the two criteria are in opposition and that there exists a tradeoff area between them.

3) Scalability experimentations: The aim of this second experimentation is to measure the overall computation times following the size of the problem. We observed previously that its most important part comes from the linear program generation.

Table II presents the generation times depending on the number of stations and time-steps. Each measure was obtained from one instance since the time needed for generating the model only depends on the size of the time extended graph. Note that when $N = 50$, $T = 288$ and $R = C = 80$, GLPK founds an optimal solution within $1'20$ minute and an

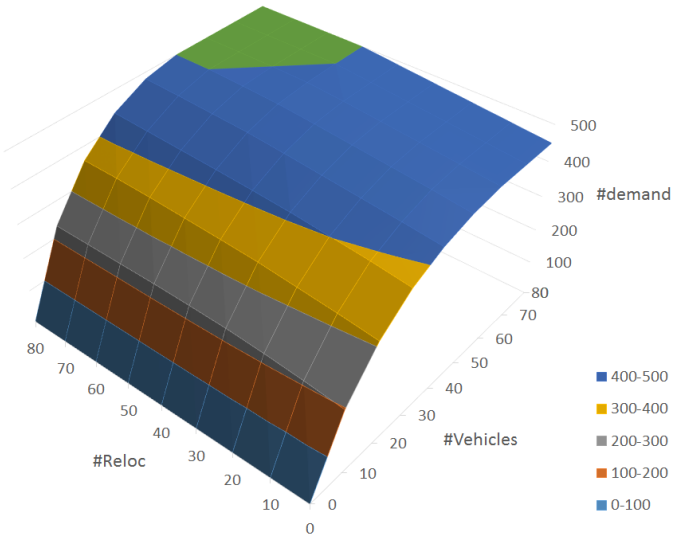

Fig. 2. 3-dimensional Pareto frontier

TABLE II
GENERATION TIME DEPENDING ON THE SIZE OF THE GRAPH (IN SECONDS)

T \ N	10	20	30	40	50
72	14	59	138 $\approx 2'30$	147 $\approx 3'$	310 $\approx 5'$
144	53	234 $\approx 4'$	343 $\approx 5'30$	632 $\approx 10'30$	1437 $\approx 24'$
288	211 $\approx 2'30$	997 $\approx 16'30$	1417 $\approx 23'30$	2481 $\approx 41'30$	4735 $\approx 1\text{h}20'$

integer optimal solution within $1'30$ minute. We observe that the generation time grows linearly following the number of arcs of the time extended graph, which is around $700\ 000$ for this instance. Moreover, this is confirmed by the correlation coefficient between generation time and the number of arcs in the graph standing at $95,9\%$. This result is coherent since the size of the linear model and the number of variables are both in $\Theta(|\mathcal{A}|)$.

For the biggest instance ($N = 50$, $T = 288$), the relocation arcs represent 98% of the total number of arcs. Therefore, we suggest to increase the scalability of the method by considering for example only short distance relocation arcs or defining them at fixed time-steps.

IV. Conclusion

In this paper, we investigated the optimization of a car-sharing *one-way* system which can be integrated in a multimodal transportation system. Specifically, in dense urban context, the demand of those systems may originate from any other modes of transportation such as trains, subways, bikes, etc. The optimization focus on fleet dimensioning when station locations and accurate demand are given. We proposed an integer linear program based on flows and we suggested three optimization criteria: maximizing the overall demand the system can absorbed, and minimizing the number of vehicles and relocation operations needed. We also described how to extract the total number of vehicles from any optimal flow.

Two numerical experiments were made using an open-source solver (GLPK). The fist one investigated computation times over small instances generated by a random generator. It turns out that the latter is negligible compared to the model building time. Furthermore, we observed that for each non-integer optimal value we get solving the LP model, its integer part is always equal to the optimal value of the corresponding ILP model. This let us think that the LP solution must contain rounding errors and might be integer. We intend to clarify this point in further research. A graphical representation of a 3-dimensional Pareto frontier served to underline and to confirm that the optimizations criteria are in opposition.

The second experiment aims at specifying the time needed for building the mathematical model when the problem grows. As it is shown in the paper, the number of constraints of the mathematical model as well as the number of decision variables are both in the range of the number of arcs present in the time extended graph. We suggested to reduce this number in order to improve model generation time selecting only those at fixed time-step representing short distance relocation operations.

The choice of GLPK instead of a commercial solver is mainly motivated by our industrial context. The good quality and the scalability of the solutions we obtained using this open-source solver allow us to transfer easily our methodology in an industrial environment.

Acknowledgment

This research work has been carried out in the framework of the Technological Research Institute SystemX, and therefore granted with public funds within the scope of the French Program "Investissements d'Avenir".

References

[1] W. J. Mitchell, *Reinventing the automobile: Personal urban mobility for the 21st century*. MIT Press, 2010.
[2] S. A. Shaheen, D. Sperling, and C. Wagner, "A short history of carsharing in the 90's," 1999.
[3] D. Jorge and G. Correia, "Carsharing systems demand estimation and defined operations: a literature review," *EJTIR*, vol. 13, no. 3, p. 201220, 2013.
[4] J. Firnkorn and M. Mller, "What will be the environmental effects of new free-floating car-sharing systems? the case of car2go in ulm," *Ecological Economics*, vol. 70, no. 8, p. 15191528, 2011.
[5] A. Fink and T. Reiners, "Modeling and solving the short-term car rental logistics problem," *Transportation Research Part E: Logistics and Transportation Review*, vol. 42, no. 4, pp. 272–292, Jul. 2006.
[6] Y. Yang, W. Jin, and X. Hao, "Car rental logistics problem: A review of literature," in *Service Operations and Logistics, and Informatics, 2008. IEEE/SOLI 2008. IEEE International Conference on*, vol. 2. IEEE, 2008, pp. 2815–2819.
[7] H. Wang, R. Cheu, and D.-H. Lee, "Dynamic relocating vehicle resources using a microscopic traffic simulation model for carsharing services," in *Computational Science and Optimization (CSO), 2010 Third International Joint Conference on*, vol. 1, 2010, p. 108111.
[8] T. Cucu, L. Ion, Y. Ducq, and J. Boussier, "Management of a public transportation service: Car-sharing service," 2009.
[9] R. Nair and E. Miller-Hooks, "Fleet management for vehicle sharing operations," *Transportation Science*, vol. 45, no. 4, p. 524540, 2011.
[10] S. L. Smith, M. Pavone, M. Schwager, E. Frazzoli, and D. Rus, "Rebalancing the rebalancers: Optimally routing vehicles and drivers in mobility-on-demand systems," *arXiv preprint arXiv:1303.3522*, 2013.
[11] W. D. Fan, R. B. Machemehl, and N. E. Lownes, "Carsharing: Dynamic decision-making problem for vehicle allocation," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2063, no. 1, p. 97104, 2008.
[12] G. H. d. A. Correia and A. P. Antunes, "Optimization approach to depot location and trip selection in one-way carsharing systems," *Transportation Research Part E: Logistics and Transportation Review*, vol. 48, no. 1, pp. 233–247, Jan. 2012.
[13] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network flows: theory, algorithms, and applications*. Englewood Cliffs, N.J.: Prentice Hall, 1993.
[14] "AMPL." [Online]. Available: http://ampl.com/
[15] "GLPK - GNU linear programming kit." [Online]. Available: http://www.gnu.org/software/glpk/