Dynamic Intra-Modal Carpooling with Transhipment : Formalization and First Combinatorial Exact Solution

Mohamed Hassine, Philippe Canalda*, Idriss Hassine
Emails: mohamedhassinef35@gmail.com, philippe.canalda@femto-st.fr, drisshassine@yahoo.fr
*Institut FEMTO-ST (UMR CNRS 6174) – Centre de Développement Multimédia NUMERICA
1, Cours Louis Leprince-Ringuet, 25200 Montbéliard, France

***Abstract*—In this work, we propose a modeling and an implementation of an algorithm for the computerization of the N-intra-modal carpooling with transhipment. It's a new type of carpooling that allows modal shifts and subsequently a variability between the offer and the demand. Another advantage of our calculator is that it checks the possibility of proposing a return itinerary for a driver who has already carried out a modal shift on his outward journey. Also, the calculator guarantees the good operation for the classic carpooling without modal shifts. Some dynamic constraints are managed by our proposed solution. A system modeling work and managed data, multiple constraints formalization and system multiple objectives as well as implementing a matching motor will be presented in this paper. This motor is based on a greedy combinatorial optimization algorithm. Various tests are run, that prove the possibility of solving this problem based on an exact approach in a reasonable time.**

Key words: dynamic carpooling, intelligent transport, dynamic time windows, itineraries with vias, greedy algorithm

I. Project genesis

Transport has a great importance in our lives. We move to work, to study, or to travel ... . Aside public transport modes, many people use the individual vehicle. In this context, carpooling presents a solution to reduce the cost's trip for different stakeholders, minimize the number of circulating vehicles and as a consequence ensure an ecological gain (preserving against pollution). Most carpooling solutions propose a matching between a driver and same passengers. This driver uses only his own vehicle to deposit passengers to their destinations as well as achieving his final destination. In this paper, we propose a formalization of the dynamic intra-modal carpooling with transhipment. The N-intra-modality makes it possible to carry over passengers or drivers for one vehicle to another vehicle. So, a carpooling's participant can choose to use a single means of transport or a maximal number of modal shifts. To summarize, the transhipment authorizes that a driver becomes a passengers.

Our mathematical modeling of this problem presents formally the different stakeholders in the system. It has a multi-constraints and multi-objectives aspect and is based on an extended origin-destination matrix. Rows and columns present different positions of the concerned territory and a matrix cell gives information on the displacement between two positions, mainly the shortest travel time and the shortest travel distance. This formalization is associated with a first Java implementation of a resolution version of the combinatorial problem. The implemented calculator validates the correct operation for carpooling's real example in the Montbliard-Basel perimeter.

II. State of the art

We present the works dealing with issues similar to our approach. We enumerate and describe relevant characteristics in order to have an overall idea of the main approaches used to solve intelligent transport problems, specifically the carpooling with transhipment (or multi-hops).

A. Synthesis of existing works' characteristics

[*Time Windows (TW):*] TW characterize temporality of an itinerary's position. It is a time interval, specified by the user, which consists of two terms (the earliest date of arrival and the latest date of departure) ([2,4,7]–[11]). A time window can be static meaning it remains unchanged along the process of the concerned algorithm ([4,7]–[10]). Also, it can be dynamic in the case of taking into account advances/delays ([2]) or if there is a normalization process following itineraries' generation ([11]). In [14], the time windows' dynamism arises from the fact that a passenger has to wait for the parcels delivery time before continuing on his way (here passengers' transport and parcels' transport are performed simultaneously in the same vehicle).

[*Vias:*] an itinerary contains vias if it consists of positions other than origin and destination. On the one hand, vias can be inter-modal, that means they are performed using the same vehicle ([8,11,14]). On the other hand, they can be intra-modal ([5]). Other works only take into account the origin and the destination ([1]–[4,6,7,9,10]).

[*Modal shifts:*] a modal shift implies the means of transport's change during a trip. This aspect is absent in the majority of our study's works. It is present only in [5]. In fact, it is the problem of transporting people with reduced mobility (PRM) at airports. It is a multimodal problem, for example a PRM wishes to move along this path: plane1-terminal1-terminal2-plane2. In this case, there are 4 sections of itinerary and in each position the PRM will be transported by a suitable means of transport adequate to its situation. It can be seen here that modal shifts are mainly related to demand requests.

[*Drivers/Passengers:*] These two roles constitute a pillar of the concept of offers and demands for transport ([2,4]–[6,8, 10,11,13]). Other works use the notion of individuality [7] or the notion of agent [1]. A first observation is that most works use these notions of roles (driver / passenger, individual, agent ...), but we note the lack of variability of role. Indeed, a driver can not be a driver and passenger at the same time and vice versa.

[*Multi-constraints and multi-objectives formalization:*] this aspect requires a mathematical formalization of the problem with respect of certain constraints and aims at optimizing a set of objectives either academic [2,5,8,10,13] or also operational [11]. Solutions based on mathematical formalization can be generated from solvers like CPLEX [12,13] or LocalSolver. Other works do not adopt this type of formalization ([4,7]). These works adopts a simple formalization based on the graph theory.

B. Positioning of our approach to the literature

Our new carpooling approach addresses the concept of dynamic time windows with 4 terms (earliest arrival date, earliest departure date, latest arrival date and latest departure date). The dynamism comes from the normalization process and propagation carried out on the time windows in order that they become adaptable to the trips' generation satisfying all stakeholders. Such time window gives birth to another notion, it is the processing time in a position (a single return/position). So we will be able to offer a return itinerary for a driver or a passenger. Also, our approach allows Vias inter-modal, that is to say in Vias we have the possibility to change the vehicle. This leads us to carry out modal shifts along a trip. This aspect is present in [5] to plan the movement of the PRM in an airport, but it is absent in the other works dealing with the problems of carpooling. To carry out modal shifts, we introduce a role variability. We then authorize a passenger to be a driver and vice-versa. The problem's formalization is multi-constraints and multi-objectives as in ([2], [5], [8], [10], [11]) and based on an origin/destination matrix as in [11]. In the term of resolution's method, we adopt an exact greedy approach as in [11] which combines the PDP/DARP/VRP/TSP problems. Also, we present 8 instances of tests. The first validates the algorithm's functional aspect for two proposed scenarios and others ensure the scaling-up.

III. Intra-modal carpooling's formalization

In this section, we formalize the problem's input data. The IM carpooling system involves drivers who present the service providers. A driver can choose between two modes. He can be an only driver, that is to say he does not accept to make a modal shift during a trip. He can also be a rather driver, that means he can use his own vehicle for a trip's portion. After that, he deposits his vehicle and accompany another driver for another trip's portion. A passenger can be just a passenger when he does not own a vehicle or a $\approx$ passenger when he owns a vehicle [1]. Also, we distinguish the operator (it is the system administrator that manages matches). Our formalization is multi-constraints and multi-objectives. It is also based on an extended origin-destination matrix M_{o-d} which returns information on the displacement between different positions (the shortest distance, the shortest time ...). Rows and columns of this matrix are the system's different positions (origin, destination or Vias). The travel time can be initialized using the average speed, and for the distance we use geographic data (Google Maps ...).

A. Drivers and offer requests

Drivers intervene in the system by sending a set of offer requests O. This list is composed by $\overline{R_o}$ requests. Each offer request $R_{o,i}$ consists of a sequence of $\{Itin_{o,i}, C_{o,i}, V_{o,i}, AS_{o,i,initial}, AS_{o,i,current}, St_{o,i}, StD_{o,i}, RM_{o,i}, GC_{i,i}\}$ with :

- $Itin_{o,i}$: indicates the driver's itinerary [2](formula 1). This itinerary consists of a positions' set (source, [vias]*, destination). Each position contains the earliest departure date $h^{--}_{o,i,pos}$, The date of departure at the latest $h^{-+}_{o,i,pos}$, The earliest arrival date $h^{+-}_{o,i,pos}$, The date of arrival at the latest $h^{++}_{o,i,pos}$, the number of boarding passengers $PU_{o,i,pos}$ (Pick-Up) and the number of descending passengers $D_{o,i,pos}$ (Delivery).
  The time window consists of 4 terms. However, considering that the processing time in a position is negligible (instantaneous PU and D), this window only reduces in two terms (h^{+-}=h^{--} and h^{-+}=h^{++}). Also, if we have a pivot position (single and return), there will be a processing time ($h^{+-} \neq h^{--}$ and $h^{-+} \neq h^{++}$).

$$Itin_{o,i} = \{Pos_{o,i,1}[h^{+-}_{o,i,1}, h^{--}_{o,i,1}, h^{++}_{o,i,1}, h^{-+}_{o,i,1}, PU_{o,i,1}, D_{o,i,1}], ..., Pos_{o,i,\overline{Itin_{o,i}}} [h^{+-}_{o,i,\overline{Itin_{o,i}}}, h^{--}_{o,i,\overline{Itin_{o,i}}}, h^{++}_{o,i,\overline{Itin_{o,i}}}, h^{-+}_{o,i,\overline{Itin_{o,i}}}, PU_{o,i,\overline{Itin_{o,i}}}, D_{o,i,\overline{Itin_{o,i}}}]\} \quad (1)$$

- $D_{o,i}$ presents the driver Dr_d associated to the offer request $R_{o,i}$. We define the set $Driver$ sush as $Driver = \{Dr_d, d \in [1, \overline{Driver}]\}$ with
  Dr_d=(driver_id, driver_name, city, postal code, registration date).
- $V_{o,i}$ presents the vehicle Vh_v associated to the offer request $R_{o,i}$. We define the set $Vehicle$ such as $Vehicle = \{Vh_v, v \in [1, \overline{Vehicle}]\}$ with
  Vh_v=(vehicle_id, vehicle_mark, vehicle_capacity, comfort type).
- $AS_{o,i,initial}$ (Available Seats) : is the number of available seats at the initialization of the offer request $R_{o,i}$.
- $AS_{o,i,current}$: is the current number of available seats.
- $St_{o,i}$: indicates the offer request's state (initialized, validated, partially contracted, contracted, I'm leaving now, I deposit my vehicle, I achieve my destination...)
- $StD_{o,i}$: indicates the mode chosen by the driver (rather driver or only driver).
- $MS_{o,i}$: is the maximum number of modal shift tolerated by a driver. If $MS_{o,i} > 0$ so, in a step's trip, the offer request will be considered as a demand request.

[1]The fact that a passenger can use his own vehicle during a trip gives rise to variability between offers and demands.

[2]this itinerary may contain, in addition to the single itinerary, a coming itinerary. This case is studied in order to offer a return itinerary to the driver making a modal shift in his single itinerary, and that to give him the possibility to recover his vehicle already deposited during the modal shift.

- $GC_{o,i}$ (single and return) : it is a boolean term, it takes a true value when the driver wants that the system propose to him a return itinerary. This choice is adaptable with the modal shift's scenario to ensure that a driver can recover his vehicle in a planned return itinerary.

B. Passengers and demand requests

Passengers intervene in the system by sending a deman requests list D. A passenger can also have a vehicle and eventually takes a rather passenger's profile ($\approx$ passenger) . Otherwise, he has the state "only passenger". This list is composed by $\overline{R_d}$ demand request. Each demand request consists of a sequence of $\{Itin_{d,j}, [D_{d,j}, V_{d,j}], St_{d,j}, StP_{d,j}, MS_{d,j}, SN_{d,j}\}$ [3] with :

- $Itin_{d,j}$: is the passenger's itinerary. It consists of a source position, a destination position and Vias.

$$Itin_{d,j} = \{Pos_{d,j,1}[h^{+-}_{d,j,1}, h^{--}_{d,j,1}, h^{++}_{d,j,1}, h^{-+}_{d,j,1} PU_{d,j,1}, D_{d,j,1}], ..., Pos_{d,j,\overline{Itin_{d,j}}} [h^{-}_{d,j,\overline{Itin_{d,j}}}, h^{+-}_{d,j,\overline{Itin_{d,j}}}, h^{--}_{d,j,\overline{Itin_{d,j}}}, h^{-+}_{d,j,\overline{Itin_{d,j}}}, PU_{d,j,\overline{Itin_{d,j}}}, D_{d,j,\overline{Itin_{d,j}}}]\} \quad (2)$$

- $St_{d,j}$: indicates the demand request state (initialized, validated, contracted [4])
- $StP_{d,j}$: indcates the passenger state (only passenger or $\approx$ passenger). When a passenger chooses the $\approx$ passenger mode, there must be a vehicle. In other terms, he can be a driver for a trip's step.
- $MS_{d,j}$: indicates the maximum modal shift number tolerated by a passenger.
- $SN_{d,j}$: indicates the seats number reserved by a passenger.

C. Operator and organizations

The calculator provides an organization list $Orgs$ consisting of $\overline{Orgs}$ possible organizations. Each present organization $O_{trip \in [1,\overline{Orgs}]}$ in this list is formed by a trip list. Each of these trips consists of an offer request (contracted or partially contracted), matched requests list (offer or demand) and a proposed itinerary (with or without modal shift). Each trip T_k is a sequence of :

- $Itin_{trip,k}$: presents the itinerary k of the organization O_{trip} affected to an offer request and matched requests.

$$Itin_{trip,k} = \{Pos_{trip,k,1}, \ldots, Pos_{trip,k,p}, \ldots Pos_{trip,k,\overline{Itin_{trip,k}}}\}$$

- $ItinRet_{trip,k}$: presents a return itinerary proposed by the calculator. This itinerary is generated under two conditions. On the one hand, the trip must allow at least a modal shift. So, we have a driver who deposits his vehicle along the trip. On the other hand, it is necessary to ensure the existence of an offer request that satisfies the driver's return itinerary.
- $Ro_{trip,k}$, is the matched offer request in a trip T_k. This request becomes contracted when the number of reserved seats in the matched offer request becomes equal to the initial number of free seats $AS_{o,i,initial}$.
- $lstMR_{trip,k}$: is a list of demand requests or offer requests matched with the offer request.

$$lstMR_{trip,k} = \{R_{trip,k,1}, ..., R_{trip,k,p}, ..., R_{trip,k,\overline{R_{trip,k}}}\}$$

- $St_{trip,k}$: indicates the trip's state (partially contracted, contracted, in progress, realized, not realized, archived...).
- $AS_{trip,k}$ (Available Seats) : indicates the available seats number in a trip (if $AS_{trip,k}$¿0, so the concerned trip and the associated offer request are both partially contracted).
- $OS_{trip,k}$ (Occupied Seats) : indicates the occupied seats number in a trip.
- $MS_{trip,k}$: is the modal shifts number in a trip.
- $GC_{trip,k}$ (single and return) : If this term is true, then we find a return itinerary in this trip.

D. Multi-constraints and multi-objectives aspect

The intra-modal carpooling system involves several actors. These actors are linked to several constraints. Thus, the organizations calculated by the solver have several objectives. In this section we list some constraints and some objectives of this system.

Normalization constraint of time windows: A request initialized by the driver becomes validated after checking time windows' consistency. We pose $M(Pos_{o,p}, Pos_{o,p+1}).t$ the time move between the position p and the position $p+1$. The addition of the departure earliest date (respectively at the latest) of the position p with the trip's time must be less or equal than the earliest departure date (respectively at the latest) of the position $p+1$.

$$\forall \quad R_{o,i \in [1,\overline{Ro}]} \in O \quad \wedge \quad St_{o,i} = "validated", \forall \quad p \in [1, \overline{Itin_{o,i}}], \quad h^{--}_p + M(Pos_p, Pos_{p+1}).t \leq h^{--}_{p+1} \quad \wedge \quad h^{-+}_p + M(Pos_p, Pos_{p+1}).t \leq h^{-+}_{p+1} \quad (3)$$

This constraint is valid $\forall \quad R_{d,j \in [1,\overline{R_d}]} \in D$.It is valid for arrival dates too.

Capacity constraint: In a proposed organization, the maximum number of occupied seats must be strictly less to the vehicle capacity.

$$\forall \quad T_k \quad \in \quad O_{trip}, \quad OS_{trip,k} \quad < \quad Vehicle.cap \quad (4)$$

Constraints of modal shift's authorization: If a trip contains a modal shift, the associated offer request must have a driver with the "rather driver" state. That means, he accepts to deposit his vehicle and continue the trip with another driver.

$$\forall \quad T_k \quad \in \quad O_{trip} \quad and \quad T_k.MS \quad = \quad 1, \quad T_k.Ro.StD \quad = \quad "rather \quad driver" \quad (5)$$

Moreover, at least one matched demand request must have a driver. This happens when its passenger state is "$\approx$ passenger".

$$\forall T_k \quad \in \quad O_t \quad \wedge \quad T_k.MS \quad = \quad 1, \exists r \quad \in \quad lstMR_{t,k} | r.StP \quad = \quad " \quad \approx \quad passenger" \quad (6)$$

Identification constraint of a return itinerary: If a trip includes a return itinerary, the matched offer request in this trip must contain a position called a pivot position where there is a processing time. That means, the intersection between the interval consisting of the earliest and latest arrival dates and the interval consisting of the earliest and latest departure dates is equal to the empty set.

$$\forall \quad R_{o,i} \in Offer \quad et \quad R_{o,i}.GC = "yes", \quad \exists \quad pos_{o,i,p} \in R_{o,i}.Itin | [pos_{o,i,p}.h^{+-}, pos_{o,i,p}.h^{++}] \cap [pos_{o,i,p}.h^{--}, pos_{o,i,p}.h^{-+}] = \varnothing \quad (7)$$

Dynamic constraint of deposited vehicle's recuperation: The proposal of a return itinerary allows the driver who has deposited his vehicle on outward itinerary to recover it on return. To guarantee this, the return itinerary must contain the position of the modal shift executed during the outward itinerary. It is the first itinerary's position of the passenger who have a vehicle (rather passenger).

$$\forall T_k \in O_t | T_k.ItinRet \neq \varnothing, \exists pos_{t,k,p} \in T_k.ItinRet, \exists r \in T_k.lstMR | r.StP = " \approx passenger" \wedge pos_{t,k,p} = r.Itin.Pos_1 \quad (8)$$

[3] Elements in brackets are optional, they will be added in case the passenger wishes to use his own vehicle ($\approx$ passenger) and they have the same definition of an offer request but replacing i by j

[4] In this formalization, we exclude the partially contracted state for a demand request (that is, our calculator does not satisfy a demand request by several offer requests, besides it always proposes the chosen destination by the passenger (for example, it does not propose a position close to the desired destination))

Constraints of inclusion between organization's trip: Trips are initialized from requests that have a vehicle (rather driver, only driver or rather passenger). The matching process in our transhipment approach can lead to inclusions between trips. This trips' inclusion is in fact mainly of an inclusion between their itineraries as indicated in the figure 1.

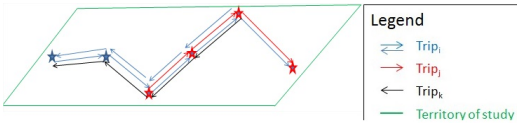

Fig. 1. inclusion between organization's trip

we suppose that the trip j is included in the trip i if the itinerary of the trip j is included in the itinerary of the trip i (case of the red and blue trips in the figure). In fact, the red trip presents a portion of the blue one. This case is conceivable in the case where a modal shift is made in the itinerary's first position of the trip j.
Also, one can have an inclusion relation if the itinerary of a trip k is included in the return itinerary of a trip i (case of the black and blue trip in the figure).

$$\forall T_i, T_j \in O_t|T_j \subseteq T_i, (T_j.itin \subseteq T_i.itin) \vee (T_j.itin \subseteq T_i.itinRet) \quad (9)$$

This inclusion gives a trips' redundancy. To remedy this problem, a redundant trip will be removed from the generated organization.

$$\forall T_i, T_j \in O_t|T_j \subseteq T_i, O_t \leftarrow O_t \setminus \{T_j\} \quad (10)$$

Maximization of contracted requests number: This objective aims for maximizing the contracted offer requests number and contracted demand requests number. That means, maximize $R_{o,con}$ et $R_{d,con}$ and as a result, minimizing $R_{o,!con}$ et $R_{d,!con}$. This objective must be realized with respect of requests' list FIFO.

Occupancy rate maximization: This objective aims for maximizing the vehicles' occupancy rate. That means, maximize the number Occupancy Rate (OR) of positions' Pick Up of all existing itineraries in generated trips.

$$\forall O_{t\in[1,\overline{Orgs}]}, \forall Itin_{t,k\in[1,\overline{O_t}]}, OR = \sum_{v=\overline{T_1}}^{\overline{T_{O_t}}} \sum_{p=1}^{\overline{Itin_{v,k}}} Pos_{v,k,p}.PU \quad (11)$$

Occupancy rate maximization must respect the capacity constraint.

Minimizing the circulating vehicles' number: One of the most important objectives of using the modal shift's concept is the minimization of the circulating vehicles' number (CV) when realizing trips. So, instead of realizing two trips with two different vehicles, we prefer realizing one trip using these two vehicles. The gain here is realizing the common itinerary between the two initial trips by using a single vehicle instead of two.

$$\forall\ O_{trip\in[1,\overline{Orgs}]} \in Orgs,\ CV = \sum_{t_{rip}=\overline{T_1}}^{\overline{T_{O_{trip}}}} Ro_{trip,k}.V \quad (12)$$

IV. Algorithm Description

In this section, We describe the algorithm's process based on two illustrative examples of displacement between Montbliard and Basel. The first example validates carpooling with detour without modal shifts. In the second example, we are going to study the modal shift. In this second one, the driver and passengers accept to change the car during the trip.

A. Carpooling with detours without modal shifts

A driver proposes an offer request $R_{o,1}$. He moves from Montbeliard to Mulhouse to work and deposits his son at the school in Fontaine (black line). The available seats number in his vehicle is equal to 5. A first passenger sends a demand request $R_{d,2}$ and the reserved seats number $SN_{d,2}$ is equal to 3. He wants to move from Hericourt to Altkirch (orange line). A second passenger sends a demand request $R_{d,3}$ with $SN_{d,2}$ = 2. He wants to move from Brevilliers to Ballersdorf (pink line). Stakeholders do not accept a modal shift ($MS = 0$) and want to be present in their different positions within specified time intervals. All this is summarized in the figure 2. The first step is to normalize time windows. We apply the function **validateInitializedRequest()** This function has as inputs initialized requests and ensures the respect of the constraint 3. Thus, we obtain validated requests. [5] with :

- $h^-_{o,1,1} = 8:05$, $h^+_{o,1,1} = 8:21$, $h^-_{o,1,2} = 8:34$, $h^+_{o,1,2} = 8:50$, $h^-_{o,1,3} = 9:15$, $h^+_{o,1,3} = 10:15$
- $h^-_{d,2,1} = 8:05$, $h^+_{d,2,1} = 8:30$, $h^-_{d,2,2} = 8:45$, $h^+_{d,2,2} = 9:08$
- $h^-_{d,3,1} = 8:00$, $h^+_{d,3,1} = 8:23$, $h^-_{d,3,2} = 8:49$, $h^+_{d,3,2} = 9:14$

Algorithm 1 Carpooling Algorithm(O_{init} O_{init}, D_{init} D_{init})

```
1: O_v ← ValidateInitializedRequests(O_i) //constraint 3
2: D_v ← ValidateInitializedRequests(D_i) //constraint 3
3: Orgs.O_1 ← Concat(Org, O_V, O_{V,rp})
4: PtripOrgs ← Orgs
5: PitinOrgs ← SingleItinerary(PtripOrgs.itin) //the single itinerary
6: if PtripOrgs.GC = "yes" then
7:     ItinRet ← ReturnItinerary(PtripOrgs.itin) //the return itinerary
8: end if
9: PpositinOrgs ← PitinOrgs
10: PD_v ← D_v
11: while PtripOrgs! = Null && !(PtripOrgs.St = "contracted") do
12:     while PD_v! = Null && !(PD_v.St.equals("contracted")) do
13:         PPosD_v ← D_v
14:         while PPosD_v! = Null && PpositinOrgs! = Null do
15:             if verifIsIinserted(PPosD_v, PpositinOrgs, PpositinOrgs.next()) //13, 14
                then
16:                 validateLocalTimeWindow(PPosD_v, PpositinOrgs.next()) //15, 16
17:                 propagationOfTimeWindow(PPosD_v, PpositinOrgs,
                    PpositinOrgs.next())//constraints 17, 18
18:                 update(PPosD_v, PpositinOrgs, PpositinOrgs.next())
19:                 PpositinOrgs ← PPosD_v
20:                 PPosD_v ← PPosD_v.next()
21:             else if PPosD_v = Null then
22:                 D_con ← D_con + PD_v
23:                 D_v ← D_v − PD_v
24:                 PpositinOrgs.Os = PpositinOrgs.OS + PD_v.SN
25:                 PpositinOrgs.AS = PpositinOrgs.AS − PD_v.SN
26:                 PD_v ← PD_v.next()
27:                 PPosD_v ← PD_v
28:                 if PpositinOrgs.RM <> 0 et PpositinOrgs ≠ ItinRet then
29:                     listeSubDemandFromOrg(PositinOrgs)
30:                     goto 16 //with this condition !(PD_v.C.equals("")))) //constraints
                        5 and 6//
31:                     fusionTwoItinerary(PpositinOrgs.itin, itinSousDemand)
32:                     if PtripOrgs.GC = "yes" then
33:                         PitinOrgs ← ItinRet
34:                         goto 16 //with respecting 8//
35:                     end if
36:                 end if
37:             else
38:                 PD_v ← PD_v.next()
39:                 PPosD_v ← PD_v
40:             end if
41:         end while
42:         PD_v ← PD_v.next()
43:     end while
44:     PtripOrgs ← PtripOrgs.next()
45:     PitinOrgs ← PtripOrgs
46:     PpositinOrgs ← PitinOrgs
47: end while
48: return Orgs
```

[5] In the following, if the earliest arrival date is equal to the earliest departure date and the arrival date at the latest is equal to the departure date at the latest, we present by h^- the earliest arrival and departure date and by h^+ the arrival and departure date at the latest.

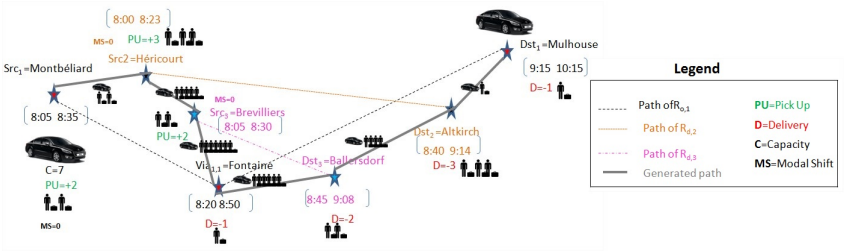

Fig. 2. Overview of carpooling without modal shifts

Then, organisation's trips will be constructed from requests having a vehicle, thus we obtain for our example an organization consisting of a single trip:

$T_1 = \{Itin_{trip,1} = \{Pos_{trip,1,1} = \{(Montbeliard[8h05,8h21], Pu = 2, D = 0),$
$Pos_{trip,1,2} = (Fontaine[8h34,8h50] Pu = 0, D = 1), Pos_{trip,1,3} = (Mulhouse[9h15,10h15] Pu = 0, D = 1)\}, R_{o,1}, lstMR_{trip,1} = , St_{trip,1} = Valide, AS_{trip,1} = 5, OS_{trip,1} = 2, MS_{trip,1} = 0\}$

This trip is constructed from $R_{o,1}$, It is only validated and it awaits matching with demand requests.

Now we start browsing demand requests to look for a possible match. This matching requires a process's application on the positions of demand requests and offer requests that can be matched :

* to insert a position 3 between positions 1 and 2, the following constraint must be respected :

$$h_1^- + M(1,3).t \leq h_3^+ \text{ and } Max(h_3^-, h_1^- + M(1,3).t) + M(3,2) \leq h_2^+ \quad (13)$$

* concerning distance, we pose $M(Pos_{o,p}, Pos_{o,p+1}).d$ the shortest distance between the position p and the position $p+1$. We tolerate a detour distance of 10 km. Considering always the triplet 1, 2 and 3, so that 3 is inserted between 1 and 2, it is necessary to respect the following constraint :

$$M(1,2).d - [M(1,3).d + M(3,2).d] \leq 10 \quad (14)$$

* if, 13 et 14 respected, We proceed to a local validation of the time windows of triplet 1, 2 and 3 (figure 3).

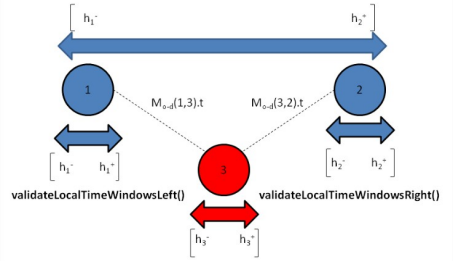

Fig. 3. Time windows' local validation

Local validation follows the next process and is performed by both functions **validateLocalTimeWindowsRight()** (step 15) et **validateLocalTimeWindowsLeft()** (step 16):

$$\begin{array}{l} If\ \ h_3^+ + M(3,2.t) > h_2^+\ \ so\ \ h_3^+ \leftarrow h_2^+ - M(3.2).t \\ If\ \ h_3^- + M(3,2).t > h_2^-\ \ so\ \ h_2^- \leftarrow h_3^- + M(3.2).t \end{array} \quad (15)$$

$$\begin{array}{l} If\ \ h_1^+ + M(1,1.t) > h_3^+\ \ so\ \ h_1^+ \leftarrow h_3^+ - M(1.3).t \\ If\ \ h_1^- + M(1,3).t > h_3^-\ \ so\ \ h_1^- \leftarrow h_3^- + M(1.3).t \end{array} \quad (16)$$

By respecting 13 and 14 and applying 15, we check the insertion of requests demand's positions in the offer request. The processing described above focuses on the function **verifIsInserted()** in the algorithm. In our example, the two demand requests $R_{d,2}$ and $R_{d,3}$ will be matched with $R_{o,1}$. Before the new itinerary's generation, a propagation of time windows is performed (figure 4).

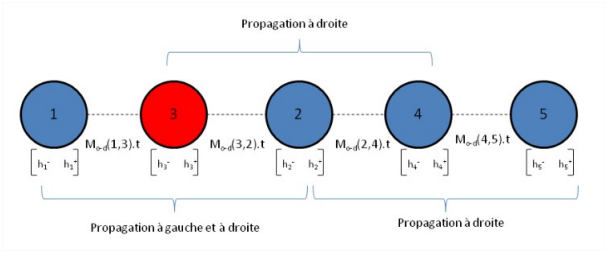

Fig. 4. Time windows' propagation

Considering the same triplet of positions, propagation follows this process: (right and left propagation 17 and 18). If the earliest date of the current position plus the distance from that position to the next position is greater than the earliest date of the next position, that date will be updated. This propagation is realized recursively on each positions' triplet.

$$h_3^- + M(3.2).t \geq h_2^- \Rightarrow h_2^- = h_3^- + M(3.2).t \wedge h_3^+ = h_2^+ - M(3.2).t \quad (17)$$

If the latest date of the current position to be inserted minus the distance between that position and the previous position is less than or equal to the latest date of that position, that date will be updated. This propagation is realized recursively on each positions' triplet.

$$h_3^+ - M(1.3).t \leq h_1^+ \Rightarrow h_1^+ = h_3^+ - M(1.3).t \wedge h_3^- = h_1^- + M(1.3).t \quad (18)$$

The last step is to generate the new route by calling the function **Update()**, so our final trip (presented with green in the illustrative example) takes this form :

$T_1 = \{Itin_{trip,1} = \{Pos_{trip,1,1} = \{(Montbeliard[8h05,8h07], Pu = 2, D = 0),$
$Pos_{trip,1,2} = (Hericourt[8h15,08h17], Pu = 3, D = 0), Pos_{trip,1,3} = (Brevilliers[8h23,8h26], Pu = 2, D = 0), Pos_{trip,1,4} = (Fontaine[8h46,8h49], Pu = 0, D = 1), Pos_{trip,1,5} = (Ballersdorf[9h06,9h09] Pu = 0, D = 1), Pos_{trip,1,6} = (Altkirch[9h11,9h45] Pu = 0, D = 1), Pos_{trip,1,7} = (Mulhouse[9h41,10h15] Pu = 0, D = 1)\}, R_{o,1}, ListRA_{trip,1} = \{R_{d,2}, R_{d,3}\}, St_{trip,1} = contractualisée, AS_{trip,1} = 0, OS_{trip,1} = 7, RM_{trip,1} = 0\}$

In this generated trip, we notice well the treatment carried out on time windows, so that they are adaptable to different positions' insertion. This insertion respects the shortest times and the shortest distances between the different positions. The two matching demand requests become contracted as well as the offer request. In fact, at the beginning $R_{o,1}$ proposes 5 free seats, 3 are occupied by $R_{d,2}$ and 2 are occupied by $R_{d,3}$. Since $R_{o,1}$ occupies 2 seats at the beginning, we get 7 occupied seats in the generated trip ($OS_{trip,1} = 7$) and 0 free seats ($AS_{trip,1} = 0$) because the vehicle capacity is equal to 7. As a result, this trip is contracted.

B. Carpooling with modal shifts

In this case, we keep the same illustrative example above, but by modifying $R_{d,3}$ et $R_{o,1}$. In fact, the new $R_{d,3}$ has a driver and a vehicle and it provides a link between Dannemarie and

Basel (blue line in the figure). Also, in $R_{o,1}$, we evolved the itinerary to ensure the return (black line between Mulhouse and Montbeliard). In addition, we added an another offer request $R_{o,4}$. This request connects Mulhouse and Montbliard via Dannemarie (purple line). Another important detail, is that all requests allow to make a modal shift (RM =1). also, there is a small change in the time windows.

The vehicle associated with $R_{o,1}$ (black vehicle) has a capacity 7 and the free seats number is equal to 5 (since the driver reserves 2 seats for him from the beginning). The vehicle associated with $R_{d,3}$ (blue vehicle) has a capacity 5 and the free seats number is equal to 2 (since the driver reserves 3 seats for him from the beginning).

The vehicle associated with $R_{o,4}$ (purple vehicle) has a capacity 6 and the free seats number is equal to 4. It carries the link between Mulhouse and Montbliard via Dannemarie. All this is summarized in figure 5 page 5.

At the beginning, the calculator begins by processing $R_{o,1}$. the driver accepts to make a modal shift, since $StD_{o,1}$ = "rather driver" and he also wants to have a return itinerary proposed by the system ($GC_{o,1}$ = "*yes*"). The request's itinerary is as follows:

$Montbeliard \rightarrow Fontaine \rightarrow Mulhouse \rightarrow Mulhouse \rightarrow Montbeliard$

To simplify the task, this itinerary is divided into two sub-itineraries: a single itinerary and a return itinerary. For this, we call two functions **SingleItinerary()** et **ReturnItinerary()**. These two functions have as input the complete itinerary and they respectively return the single itinerary and the return itinerary [6].

The application of these two functions provides two itineraries:

- single itinerary : $Mulhouse \rightarrow Fontaine \rightarrow Mulhouse$
- return itinerary : $Mulhouse \rightarrow Montbeliard$

Subsequently, the treatment will split in two parts. A first part will executed by considering only the single itinerary of $R_{o,1}$ including a modal shift. A second part will ensure the proposal of a return itinerary, so that the driver of $R_{o,1}$ can recover his car on a planned return itinerary.

For the first part of the processing, the same time window normalization treatment described in the previous scenario is applied, so validated requests are obtained with:

- $h^{-}_{o,1,1} = 8:05$, $h^{+}_{o,1,1} = 8:21$, $h^{-}_{o,1,2} = 8:34$, $h^{+}_{o,1,2} = 8:50$, $h^{+-}_{o,1,3} = 9:15$, $h^{--}_{o,1,3} = 13:15$, $h^{++}_{o,1,3} = 10:15$, $h^{-+}_{o,1,3} = 13:45$
- $h^{-}_{d,2,1} = 8:00$, $h^{+}_{d,2,1} = 8:25$, $h^{-}_{d,2,2} = 8:49$, $h^{+}_{d,2,2} = 9:15$
- $h^{-}_{d,3,1} = 8:30$, $h^{+}_{d,3,1} = 9:03$, $h^{-}_{d,3,2} = 9:40$, $h^{+}_{d,3,2} = 11:02$

Afterward, the solver generates a first trip that matches $R_{o,1}$ et $R_{d,2}$. This is effected by respect of 3, 13 et 14 and by applying the treatment described in 15, 18 et 17. Since trips are generated from requests having a vehicle, two other trips will be initialized from $R_{d,3}$ and $R_{o,4}$. However, these two trips will be included at the end of processing in the first trip (constraint 9). Then they will be eliminated when the final result is generated (constraint 10). So, in the following, we focus on the treatment performed on the first trip.

$T_1 = \{Itin_{t,1} = \{Pos_{t,1,1} = \{(Montbeliard[8h05, 8h10], Pu = 2, D = 0),$
$Pos_{t,1,2} = (Hericourt[8h18, 08h23], Pu = 1, D = 0), Pos_{t,1,3} = (Fontaine[8h34, 8h45], Pu = 0, D = 1), Pos_{t,1,4} = (Altkirch[9h04, 9h15] Pu = 0, D = 1), Pos_{t,1,5} = (Mulhouse[9h19, 10h19] Pu = 0, D = 1)\}, ItinRet_{t,1} = \{\}, R_{o,1}, lstMR_{t,1} = \{R_{d,2}\}, St_{t,1} = partially\ \ contracted, AS_{t,1} = 4, OS_{t,1} = 3, MS_{t,1} = 1, GC_{t,1} = "yes", Dr_{t,1} = \{name = (HASSINE), city = (Montbeliard)\}, Vh_{t,1} = \{mark = (Audi), capacity = (7)\}\}$

In this first trip, there are still free seats ($AS_{t,1} = 4$). So, this trip is partially contracted. The occupied seats number is equal to 3 ($OS_{t,1} = 3$) (2 seats occupied by the driver and another seats reserved by $R_{d,1}$).

Another important detail is that modals shift number of T_1 is equal to 1 ($MS_{t,1} = 1$). In fact, $MS_{t,1} = Min\{(MS_{o,1}, RM_{d,2}\}$. In this case, the algorithm triggers a new processing in order to find a new matching with a modal shift to ameliorate the trip's quality. The current trip is divided into sub-demand requests from the second position. This is guaranteed by applying the function **listSubDemandFromOrg()**. This function takes as input the current organization's trip and returns sub-demands from the second position. In our case from this trip:

$Montbeliard \rightarrow Hericourt \rightarrow Fontaine \rightarrow Altkirch \rightarrow Mulhouse$

We generate three sub-demands :

- sub-demand 1 : $Hericourt \rightarrow Fontaine \rightarrow Altkirch \rightarrow Mulhouse$
- sub-demand 2 : $Fontaine \rightarrow Altkirch \rightarrow Mulhouse$
- sub-demand 3 : $Altkirch \rightarrow Mulhouse$

The first sub-demand has a seat's required number equal to 3 because $R_{o,1}.pos_1.PU = 2$ and $R_{d,1}.pos_1.PU = 1$ (2+1=3).

The second sub-demand has a seat's required number equal to 2 because $R_{o,1}.pos_1.PU = 2$, $R_{d,1}.pos_1.PU = 1$ and $R_{o,1}.pos_2.D = 1$ (2+1-1=2).

The third sub-demand has a seat's required number equal to 1 because $R_{o,1}.pos_1.PU = 2$, $R_{d,1}.pos_1.PU = 1$, $R_{o,1}.pos_2.D = 1$ and $R_{d,1}.pos_2.D = 1$ (2+1-1-1=1).

Thereafter, the algorithm browses the demands' list to search for a request having a driver and consequently a vehicle. In our case, this request is $R_{d,3}$ ($Dannemarie \rightarrow Basel$). This request behaves in this phase of the algorithm as an offer request that looks for matching with one of sub-demands already cited. If a match is made, we stop the treatment. $R_{d,3}$ offers 2 free seats. By respecting the time window constraints and the tolerated detour distance constraint, it will be matched with the second sub-demand and we obtain this itinerary:

$Fontaine \rightarrow Dannemarie \rightarrow Altkirch \rightarrow Mulhouse \rightarrow Bale$

The last step is to merge this itinerary with the old itinerary generated in the current organization :

$Montbeliard \rightarrow Hericourt \rightarrow Fontaine \rightarrow Altkirch \rightarrow Mulhouse$

This process is guaranteed by the function **fusionTwoItinerary()** which takes as input two itineraries and returns the merged itineraries. The new trip T_1' have this form :

$T_1' = \{Itin_{t',1} = \{Pos_{t',1,1} = \{(Montbeliard[8h05, 8h08], Pu = 2, D = 0),$
$Pos_{t',1,2} = (Hericourt[8h15, 08h18], Pu = 1, D = 0), Pos_{t',1,3} = (Fontaine[8h42, 8h45], Pu = 0, D = 1), Pos_{t',1,4} = (Dannemarie[8h55, 8h58], Pu = 3, D = 0), Pos_{t',1,5} = (Altkirch[9h12, 9h15], Pu = 0, D = 1), Pos_{t',1,6} = (Mulhouse[9h42, 13h15, 10h15, 13h45], Pu = 0, D = 1)\}, Pos_{t',1,7} = (Bale[10h23, 11h02] Pu = 0, D = 3), ItinRet_{t,1} = \{\}, R_{o,1}, lstMR_{t',1} = \{R_{d,2}, R_{d,3}\}, St_{t',1} = partially\ \ contracted, AS_{t',1} = 1, OS_{t',1} = 6, MS_{t',1} = 1, GC_{t',1} = "yes", Dr_{t,1} = \{names = (HASSINE, HASSINE), cities = (Montbeliard, Paris)\}, Vh_{t,1} = \{marks = (Audi, BMW), capacities = (7, 5)\}\}$

This trip matches an offer request and two demand requests and it includes a modal shift in Dannemarie. We also note the existence of two drivers and two vehicles assigned to this trip, which is an immediate consequence of the modal shift's application. In this trip, the return itinerary is not yet generated. Now, the treatment's second part will be executed in order to propose a return itinerary for the driver of $R_{o,1}$. This proposal for a return itinerary has two advantages. On the one hand, it is an obvious satisfaction of a system's participant. On the other hand, it allows the driver to recover his already disposed vehicle during the first itinerary.

[6] The division of the single itinerary and return itinerary is based on the presence of a position called pivot position where there is a processing time. That means, the intersection between the interval consisting of the earliest and the latest arrival dates and the interval consisting of the earliest and latest Departure times is equal to the empty set (constraint 7. In our case, this position is Mulhouse.

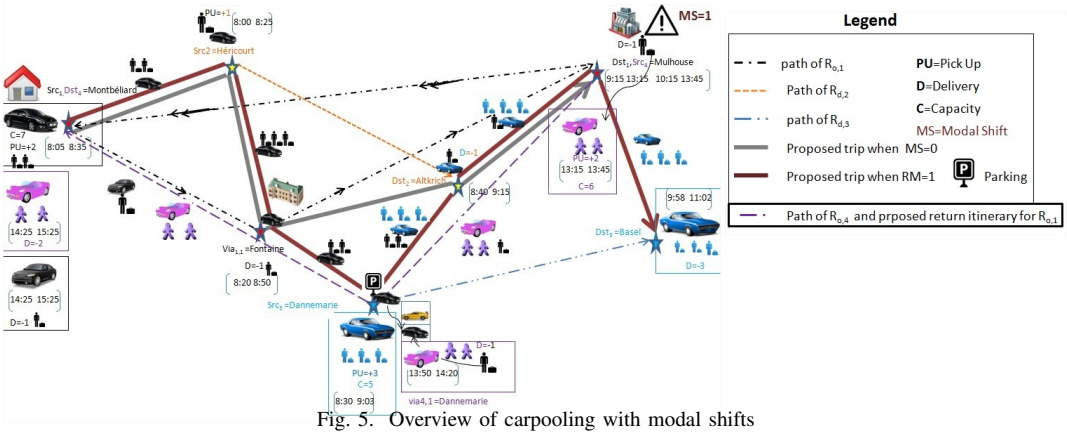

Fig. 5. Overview of carpooling with modal shifts

The return path of $R_{o,1}$ is : $Mulhouse \rightarrow Montbeliard$

The strength of our algorithm is that it keeps the single itinerary's history. Indeed, the driver of $R_{o,1}$ driver desposed his vehicle at Dannemarie on his first journey. So, the solver go through the offer requests' list in order to looking for a request that has an itinerary that goes through Dannemarie (so that the driver of $R_{o,1}$ takes his way back with his car disposed at Dannemarie).

It is the case for the request offer $R_{o,4}$ (purple line in the figure) having the following itinerary :

$Mulhouse \rightarrow Dannemarie \rightarrow Montbeliard$

With respecting constraints 3, 13 et 14 and applying the treatment described in 15, 18 et 17. The new trip $Trip''_1$ will have this form :

$T''_1 = \{Itin_{t''_,1} = \{Pos_{t''_,1,1} = \{(Montbeliard[8h05, 8h08], Pu = 2, D = 0), Pos_{t''_,1,2} = (Hericourt[8h15, 08h18], Pu = 1, D = 0), Pos_{t''_,1,3} = (Fontaine[8h42, 8h45], Pu = 0, D = 1), Pos_{t''_,1,4} = (Dannemarie[8h55, 8h58], Pu = 3, D = 0), Pos_{t''_,1,5} = (Altkirch[9h12, 9h15], Pu = 0, D = 1), Pos_{t''_,1,6} = (Mulhouse[9h42, 13h15, 10h15, 13h45], Pu = 1, D = 1)\}, Pos_{t''_,1,7} = (Basel[10h23, 11h02]Pu = 0, D = 3), ItinRet_{t,1} = \{Pos_{t''_,1,8} = \{(Mulhouse[13h15, 13h45], Pu = 2, D = 0), Pos_{t''_,1,9} = (Mulhouse[9h42, 13h15, 10h15, 13h45], Pu = 1, D = 1), Pos_{t''_,1,10} = (Dannemarie[13h50, 14h20], Pu = 0, D = 1), Pos_{t''_,1,11} = (Montbeliard[14h30, 15h25], Pu = 0, D = 1), Pos_{t''_,1,12} = (Montbeliard[14h30, 15h25], Pu = 0, D = 1)\}, R_{o,1}, lstMR_{t''_,1} = \{R_{d,2}, R_{d,3}, R_{o,4}\}, St_{t''_,1} = partially\ contracted, AS_{t''_,1} = 1, OS_{t''_,1} = 6, MS_{t''_,1} = 1, GC_{t''_,1} = "yes", Dr_{t''_,1} = \{names = (HASSINE, HASSINE, CANALDA), cities = \{(Montbeliard, Paris, Montbeliard)\}, Vh_{t''_,1} = \{marks = (Audi, BMW, Peugeot), capacites = (7, 5, 7)\}\}$

In this trip, we note the existence of an offer request $R_{o,4}$ in $lstMR_{t''_,1}$. This confirms another time the variability between offers and demands in our approach. Indeed $R_{o,4}$ behaves in this phase of the algorithm as a demand request that was matched with the return's offer of $R_{o,1}$. Also the drivers' list and the vehicles'list have now the size 3 (single itinerary, modal shift, return itinerary).

To conclude, the first driver (with a black vehicle of capacity 7) leaves Montbeliard occupying 2 places. He goes through Hricourt to recover a first passenger. Then, he goes to his first via Fontaine to dispose his companion of Montbeliard. The second driver with a blue vehicle of capacity 5 (three seats are occupied, so only 2 seats are available) waits the arrival of the first to Dannemarie. When arriving, the first driver leaves his vehicle in a parking and takes the road (him and the first passenger of Hericourt) with the second driver. The next destination is Altkirch to dispose the first passenger then Mulhouse to dispose the first driver. Then, the second driver goes to Base (him and their two accompanying of Dannemarie). Now, a third driver (the one who issues the offer request $R_{o,4}$ having a purple vehicle with capacity 5) ensure first driver's return to Dannemarie. The latter takes back his vehicle and both go to Montbeliard (each with his vehicle).

The main objective of this second scenario is to minimize the vehicles' number traveling in the road (in our example the road between Fontaine and Mulhouse). The described itinerary is shown in figure 5 page 7 by the beige color (the modal shift's case). Also, the itinerary where there is no modal shift is presented by the green color. The algorithm already described in this section is presented in algo 1.

V. New instances and evaluation

In order to evaluate the algorithm, We implemented a test simulator in java. This simulator generates automatically both demand requests file and offer request file by varying the requests number and their characteristics (positions, time windows, vehicle capacity, etc.)[7]. The test sets implement, on the one hand, the execution time necessary for matching and on the other hand the number of requests according to their state (contracted : $\overline{R_c}$, partially contracted : $\overline{R_{pc}}$ or only validated : $\overline{R_v}$). Also, for each test, we calculated the trips' number including a modal shift $\overline{Tr_{MS}}$ in order to evaluate the percentage of such trips (which characterize our new type of carpooling) in an organization. These tests validate the algorithm's operation for the two scenarios already described before. Thus, other scenarios were discussed exhaustively to ensure scaling up.

We define the folowing configuration : $config_{test} = \overline{R_o}/\overline{R_d}$ with $\overline{R_o}$ presents the number of offer requests initialized by drivers and $\overline{R_d}$ presents the number of demand requests initialized by passengers. From the second configuration, we

[7]The request's generation is done in a way to increase the probability of matching in order to properly evaluate the algorithm's performance.

config test	$\overline{R_{o,c}}$	$\overline{R_{o,pc}}$	$\overline{R_{o,v}}$	$\overline{R_{d,c}}$	$\overline{R_{d,v}}$	$\overline{Tr_{MS}}$	$\overline{\%Tr_{MS}}$	ms
$config1 = 3/4$	1	2	0	4	0	1	50%	11
$config2 = 30/60$	21	4	5	50	10	4	16%	33
$config3 = 60/120$	41	5	14	90	30	7	15%	70
$config4 = 120/240$	75	13	32	180	60	15	17%	99
$config5 = 240/480$	160	23	57	350	130	41	22%	276
$config6 = 480/960$	330	40	110	801	159	95	25%	677
$config7 = 960/1920$	639	63	258	1701	219	201	66%	1319
$config8 = 1920/3840$	1280	183	454	2880	960	366	24%	3566

TABLE I
FUNCTIONAL TEST (8 CONFIGURATIONS) OF N-INTRAMODAL CARPOOLING WITH TRANSHIPMENTS

chose that demand requests' number presents the half of offer requests' number. We varied configurations for $config_1 = 3/4$ (it is the configuration of our two illustrative examples) to $config_8 = 1920/3840$. The table 1 shows test' results.[8]. We note the trips' percentage including a modal shift varies between 15 % and 50 % for the different test configurations. This percentage shows the rate's trip that have amelioration on the term of matching. This validates our choice of adding the transhipment to the classic carpooling in order to satisfy the most system's participants. The figure 6 shows the execution time evolution.

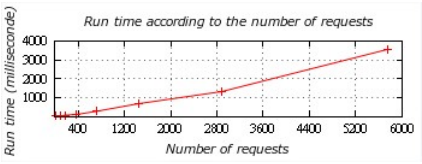

Fig. 6. Run time execution of our exact combinatorial algorithm solving N-intra-modal carpooling problem with transhipment

This curve has a linear behavior. This is adequate with the theoretical complexity of our algorithm (in the order of O (n)). For our two illustrative examples (3 offer requests and 4 demand requests), the matching execution time is **11 ms**. For 5760 requests (1920 offer requests and 3840 demand requests), the execution time is **3.5s** which is acceptable for a matching algorithm processing a large number of data (in the order of thousands).

VI. CONCLUSIONS AND PERSPECTIVES

In this paper, we discussed the problem of carpooling with and without modal shifts. Initially, we proposed a new formalization of this problem presenting on the one hand the input data (offer requests and demand requests) and on the other hand the output data (organizations). This formalization is based on an extended origin-destination matrix which makes information about displacements between positions mainly the shortest time and the shortest distance. It is also multicon-straints (time windows' constraint, capacity constraints ...) and multiobjectives (the main objective is to maximize the number of contracted requests).

Then, we described the matching algorithm by implementing two real scenarios of displacement between Montbeliard and Basel. The first scenario validates the carpooling's classic behavior without modal shifts, that means there is no vehicle's change during a trip. The algorithm's second part was addressed to the second scenario. Indeed, in this case, we note well the variability between the offer and the demand because a demand request having a driver can behave like an offer request. The fact that passengers and drivers agree to make at least a modal shift gives us the opportunity to ameliorate the concerned organization by proposing a new trip including a modal shift. This modal shift increases on the one hand the number of satisfied requests and on the other hand minimizes the number of vehicles along a travel.

In parallel with solver's implementation, we started the creation of an IOS mobile application. Several interfaces are already realized. Also, a REST web service is being implemented to try a possible integration between the calculator and the mobile application.

[8]The machine used during the tests is an ASUS K56C laptop under Windows 7 professional 64 bits equipped with an Intel Core i5-3317U (4 core), 1.7GHz and 6GB RAM.

REFERENCES

[1] Galland and al, Simulation Model of Carpooling With the Janus Multi-agent Platform, In Procedia Computer Science 19 (2013) 860-866.

[2] Bruglieri and al, PoliUniPool : a carpooling system for universities, In Procedia Social and Behaviral Sciences 20 (2011) 558-576.

[3] Zhang and al, Research on Strategy Control of Taxi Carpooling Detour Route under Uncertain Environment. In Journal "Discrete Dynamics in Nature and Society", Volume 2016 (2016), 11 pages.

[4] Cordeau and Laporte, A tabu search heuristic for the static multi-vehicle dial a ride problem, In Transp. Research Part B 37 (2003) 579-594.

[5] Blander and al, Synchronized dia-a-ride transportation of disabled passengers at airports, In EJOR 225 (2013) 106-117.

[6] Khande and al, Design and simulation of Carpool Service Problem using Soft Computing Tools, In International Journal of Recent Trends in Engineering and Research, Volume 2,Issue 4(04 - 2016).

[7] Knapen, L., Yasar, A., Cho, S. et al, Exploiting Graph-theretic Tools for Matching in Carpooling Application. In Journal of Ambient Intelligence and Humanized Computing, (2014) 5: 393.

[8] Iori, M. and Martello, S., Routing problems with loading constraints. In journal TOP July 2010, Volume 18, Issue 1, pp 427.

[9] Malandraski and al, A restricted dynamic programming heuristic algorithm for the time dependent traveling salesman problem, In European Journal of Operational Research 90 (1996) 45-55.

[10] Ropke, S. and Cordeau, J.-F. and Laporte, G.. Models and Branch-and-Cut Algorithms for Pickup and Delivery Problems with Time Windows. In Networks an Int. Journal, Vol. 49, Issue 4, July 2007, pages 258-272.

[11] Hassine and Canalda, Premire Résolution Combinatoire Gloutonne du probléme du louage incrémental, dans la Société Franaise de Recherche Opérationnelle et d'Aide *à* la Décision, Février 2017.

[12] Chen and all, Multiple crossdocks with inventory and time windows, in Computers and Operations Research 33 (2006) 43-63.

[13] Berlingerio, M. et al. The GRAAL of carpooling: GReen And sociAL optimization from crowd-sourced data. In Transp. Research Part C : Emerging Technologies, Vol. 80, july 2017, pp 20-36.

[14] Baoxiang and all, The Share-a-Ride Problem: People and parcels sharing taxis, in European Journal of Operational Research 238 (2014) 31-40.