FLEET REROUTING STRATEGIES WITH REAL-TIME TRAFFIC INFORMATION

Barceló, Jaume. Department of Statistics and Operations Research and CENIT (Center for Innovation in Transport), Universitat Politècnica de Catalunya.

Orozco, Jesús Arturo. Department of Statistics and Operations Research and CENIT (Center for Innovation in Transport), Universitat Politècnica de Catalunya.

ABSTRACT

The design and evaluation of City Logistics applications requires an integrated framework in which all components could work together. Therefore, City Logistics models should account for vehicle routing applications and fleet management models capable of including also the dynamic aspects of the underlying road network, namely when ICT applications are taken into account. This paper develops a methodological proposal based on an integration of vehicle routing models and real-time traffic information. In the computational experiments conducted in this paper, a dynamic traffic simulation model has been used to emulate the actual traffic conditions providing, at each time interval, estimates of the traffic state on each link of the road network that are then used, by a real time fleet management system, to determine the optimal dynamic routing and scheduling of the fleet.

Keywords: real-time traffic information, time-dependent travel times, dynamic vehicle routing, tabu search

1. INTRODUCTION

Fleet management in urban areas has to explicitly account for the dynamics of traffic conditions leading to congestions and variability in travel times severely affecting the distribution of goods and the provision of services. An efficient management should be based on decisions accounting for all factors conditioning the problem: customers' demands and service conditions (time windows, service times and others) fleet operational conditions (positions, states and availabilities of vehicles) and traffic conditions. Instead of making decisions based on trial and error and operator's experience a sounder procedure would be to base the decisions on the information provided by a Decision Support System. Regan et al. (1997, 1998) provide a conceptual framework for the evaluation of real-time fleet management systems (see Figure 1), which considers dynamic rerouting and scheduling decisions implied by operations with real-time information as new orders or updated traffic conditions.

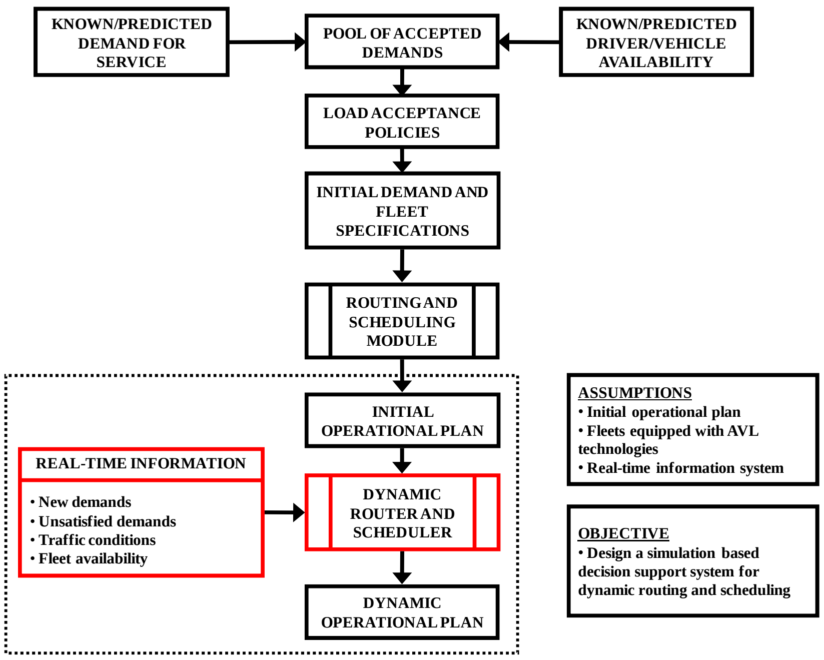

KNOWN/PREDICTED DEMAND FOR SERVICE
POOL OF ACCEPTED DEMANDS
KNOWN/PREDICTED DRIVER/VEHICLE AVAILABILITY
LOAD ACCEPTANCE POLICIES
INITIAL DEMAND AND FLEET SPECIFICATIONS
ROUTING AND SCHEDULING MODULE
INITIAL OPERATIONAL PLAN
REAL-TIME INFORMATION
• New demands
• Unsatisfied demands
• Traffic conditions
• Fleet availability
DYNAMIC ROUTER AND SCHEDULER
DYNAMIC OPERATIONAL PLAN
ASSUMPTIONS
• Initial operational plan
• Fleets equipped with AVL technologies
• Real-time information system
OBJECTIVE
• Design a simulation based decision support system for dynamic routing and scheduling

Figure 1 – Conceptual scheme for the evaluation of real-time fleet management systems

Unlike the classic approach, where routes are planned with the known demand and they are unlikely to be changed throughout the planning period, the real-time fleet management approach (inside the dotted frame) assumes that real-time information is constantly revealed to the fleet manager who has to decide whether the current routing plan should be modified or not.

The logic process, depicted in the diagram of Figure 1, assumes a partial knowledge of the demand. The service starts at the beginning of the time period (i.e. one day) proposing an initial schedule of the available fleet to service the known demand. This is the initial operational plan which will be modified accordingly later on, when the operation is ongoing and new real-time information is available. This information can correspond to new demands, unsatisfied demands, changes in the routes due to traffic conditions, changes in the fleet availability (i.e. vehicle breakdowns) and others, which feeds a dynamic router and scheduler that computes and suggests a new dynamic operational plan. In Figure 2, we depict an example of how the conceptual process described in Figure 1 can be handled by a dynamic router and scheduler system.

The various colors identify the initially assigned routes to a set of five vehicles and the order in which customers will be served according with the initial schedule. We also assume that vehicles can be tracked in real-time. At time *t*, after the fleet has started its operational plan, a new customer calls requiring a service which has not been scheduled. If real-time information, such as positions and states of the vehicles and current and forecasted traffic conditions, is available to the fleet manager, he can use it to make a better decision on which vehicle to assign to the new call and whether the new assignment results in a direct diversion from a route (vehicle 2) or a later scheduling (vehicle 1).

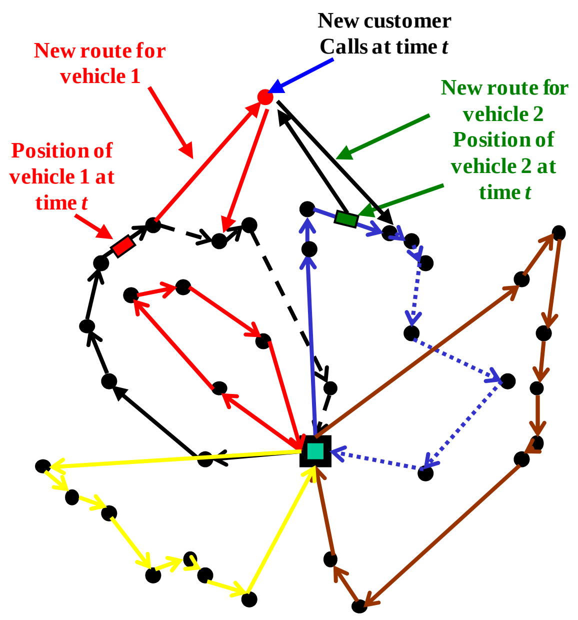

New customer Calls at time *t*
New route for vehicle 1
Position of vehicle 1 at time *t*
New route for vehicle 2
Position of vehicle 2 at time *t*

Figure 2 – Dynamic vehicle rerouting in a real-time fleet management system

1.1 Motivation for the Dynamic Approach

Although the static version of the vehicle routing problem (VRP) has been widely studied in the literature, the developments in ICT have accelerated the research in the dynamic case of the problem. The dynamic VRP assumes a partial knowledge of the demand and that new real-time information is revealed to the fleet manager throughout the operational period. A comprehensive review of the dynamic problem can be found in Ghiani et al (2003).

The VRP with time-dependent travel times acknowledges that traffic conditions have an influence on the routing planning. An introduction to the problem and a heuristic development is made in Malandraki and Daskin (1992). Further algorithm developments for the static VRP with time-dependent travel times can be found in Ichoua et al (2003), Fleishman et al (2004) and Tang (2008). To our knowledge, the dynamic VRP with time-dependent information has only been addressed by Chen et al (2006) and Potvin et al (2006). In our approach, time-dependent information is generated by means of traffic simulation of a real-world urban network in order to emulate the role of an Advanced Traffic Information System providing reliable real-time information on traffic conditions and short-term forecasts. As far as we know, the combination of dynamic traffic simulation tools and vehicle routing models has not been addressed in the literature.

Thus, the methodological proposal object of this paper assumes the availability of such real-time information and develops ad hoc version of decision rules and time-dependent route scheduling algorithms as core components of such dynamic router and scheduler.

In order to test the management strategies and algorithms assuming real-time information, we have used a dynamic traffic simulation model, emulating in this way the estimates of the current travel times as a function of the prevailing traffic conditions and the short-term forecast of the expected evolution of travel times that an advanced traffic information system would provide. This is the basis for more realistic decisions on the feasibility of providing the requested services within the perceptive time windows. Furthermore, simulation can also emulate real-time vehicle tracking giving access to positions and availabilities of the fleet vehicles, which is the information required by a dynamic router and scheduler.

2. THE DYNAMIC ROUTER AND SCHEDULER

The Dynamic Router and Scheduler (DRS) configure the core of the proposed Decision Support System. It is the main objective of this paper and it consists of a set of algorithmic tools to decide which vehicle will be assigned to the new task and to provide an on-line solution to the underlying real-time vehicle routing problem. Under the proposed platform, the DRS is triggered by events generated by a discrete-event simulator, which emulates the dynamics of the operation of a fleet of vehicles throughout the planning period. We distinguish two types of events, external and internal. The external events depend mostly on the demand and other factors which are independent of the traffic conditions experienced in the road network, e.g. a new customer request, a cancelation of an order, changes in demand or time windows, breakdown of vehicles, etc. Internal events, in contrast, depend on the observed traffic flow dynamics which could be the cause, for instance, of delays in the estimated arrival times or service start times.

The interactive logic of the process is conceptually illustrated in the diagram of figure 3. The dynamic events simulator emulates the evolution of general traffic flows in the network, the progress of the tracked fleet and the random events experienced by the fleet manager. An event-based mechanism within the simulation platform generates the external events that are likely to change the fleet operational plan. Internal events are triggered when certain condition or criteria is met due to the prevailing traffic conditions of the network. Internal and external events and the associated information are the primary inputs to the DRS. Whenever an event is triggered, the DRS re-computes routes on the basis of a modified *tabu search* heuristic, derived from the *tabu search* methodology used in the computation of static operational plans.

Additionally, several solving strategies can be considered at this stage to propose a solution and evaluate the best alternative. For example, under a one-by-one treatment, a new solution is proposed for every customer request received. An alternative might be to “pool” several requests using certain criteria (e.g. distance, proximity, etc). These strategies are considered to be reactive as they react to a request or set of requests. Preventive strategies might look to relocate vehicles in anticipation of new requests. By using waiting strategies, a fleet manager could intentionally delay or anticipate a vehicle’s move to the next location.

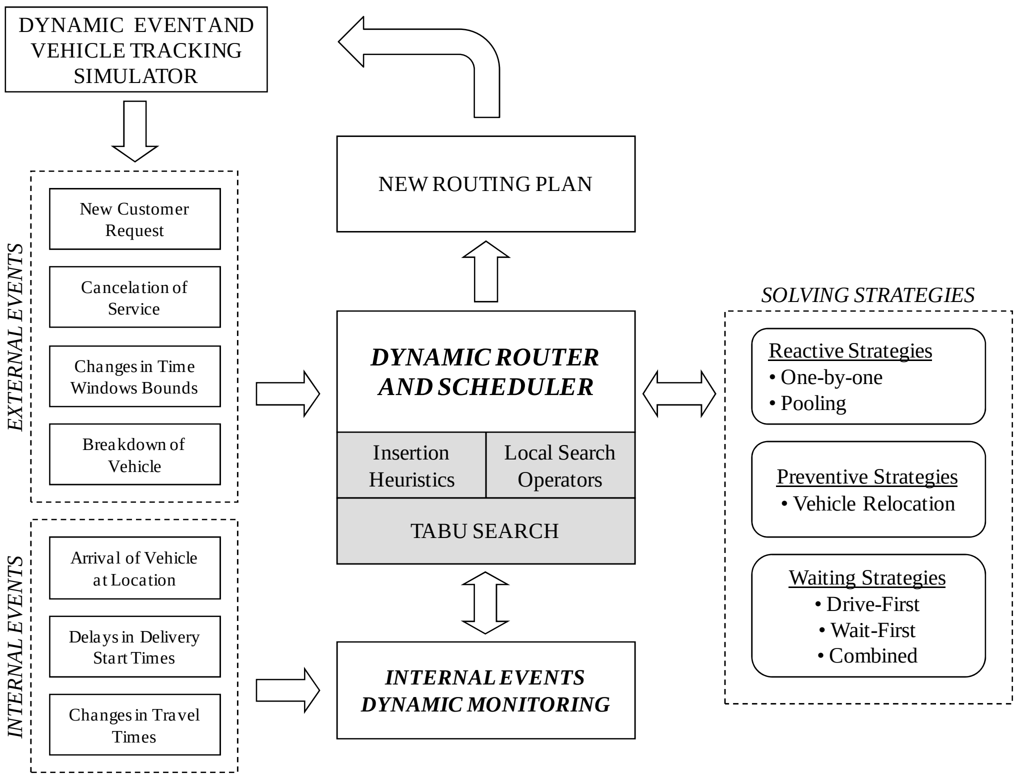


Figure 3 – Processes at the Dynamic Router and Scheduler

3. DEALING WITH TIME-DEPENDENT TRAVEL TIMES

Typical urban networks are subject to numerous random events resulting in traffic conditions with high levels of variability. This is particularly visible in commuting hours (morning and afternoon extended rush hours) when travel times in many streets rise significantly in comparison with other time slots due to traffic congestion on the roads. Thus, one vehicle traveling from origin *O* to destination *D* and leaving at time t_1 will likely have a different travel time when leaving at time t_2.

In order to illustrate this argument, we have calculated the travel time averages and variances of all links for several replications of a dynamic traffic simulation model of the downtown area of Barcelona (Barcelona's model and its calibration were available to us as a result of previous research projects.). The results, computed from 7:00 to 17:00 hours every five minutes, are shown in Figure 4. As not all links on the road network have the same lengths and capacities, some variance is expected. Thus, it is possible to identify two time regions where the variability of link travel times is significantly different. The first period, approximately from 7:30 to 10:00, corresponds to the morning extended rush hour, where the network, which begins simulation with light traffic conditions, passes rapidly to a heavy congestion state resulting in a higher variability of the travel times. The same behavior can be observed in the afternoon extended hours (last 2 hours of simulation). Between these two periods, we have a time slot where variability is stationary due to normal traffic conditions.

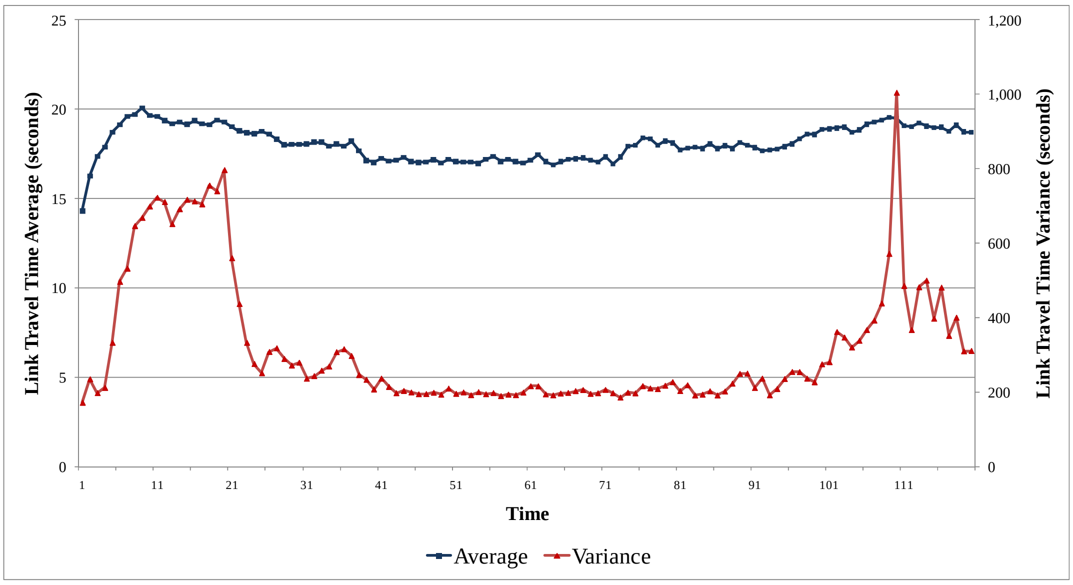


Figure 4 – Average travel times and travel time variances in Barcelona's downtown

Moreover, there is evidence that a large proportion of fleet operations take place during rush hours, as pointed out by Robusté (2005), where the results from a survey, in 60,000 retail outlets in the city of Barcelona, found that at 11:00 hours more than 50% of the goods have been delivered with an average service time between 13 and 16 minutes.

The above results state that working exclusively with average travel times may lead to significant deviations in city logistics problems where temporality is an important factor such as the vehicle routing and scheduling problem with time windows. In our research, the proposed platform is able to emulate an advanced traffic information system (ATIS) which provides link time-dependent data, which in turn is used to build routes and guide vehicles through the urban network. Therefore, given that time-dependent data is available, the decision support system is able to compute time-dependent shortest paths between any pair of nodes of the road network and provide them to a user, which, in this case, is the dynamic router and scheduler.

In the literature, the most studied form of shortest path problems are static, that is, they assume that link travel times or costs are constant. When link travel times change over time, shortest paths are said to be dynamic. Cooke and Halsey (1996), Orda and Room (1990), Ziliaskopoulos and Mahmassani (1993), among others, have proposed algorithms that take into account the variability of link travel times. In this research, we have selected the algorithm proposed by Chabini (1998, 1999). His algorithm, known as Decreasing Order of Time (DOT), computes shortest paths from all nodes to one destination for all departure times. DOT assumes that after certain time period T_{max}, all link costs remain constant. The algorithm starts calculating static shortest paths at time T_{max} and moves backwards calculating shortest paths for the previous departure time and so forth.

Let us now describe the DOT algorithm. Let $G^R = (N^R, A^R)$ be a directed graph that represents the road network where $N^R = \{1,\ldots,n\}$ is the set of nodes and $A^R = \{(i,j) \mid i,j \in N^R, i \neq j\}$ is the set of links. Let $d_{ij}(t)$ be the travel time between nodes *i* and *j* when departure time is *t*, $t \in \{0,1,\dots\ T_{max}\}$. Let us assume that we are interested in compute the shortest paths from all nodes of the network to certain destination node *q*. Let *C* be a matrix where entry a_{ij} denotes the cost (or travel time) of an optimal path from node *i* to node *q*, when departing time is *j*. Let *N* be a matrix where entry a_{ij} represents the next node to visit on the optimal path from node *i* to node *q* departing at time *j*. The following is a pseudo-code for DOT algorithm:

```
1. Initialization
      For $i$ = 1 to $n$ do:
            For $t$ = 0 to $T_{max}$ do:
                  $C[i][t] = \infty$
                  $N[i][t] = 0$
      For $t$ = 0 to $T_{max}$ do:
            $C[q][t] \leftarrow 0$
      Compute static shortest paths at $t = T_{max}$
2. Main Loop.
      For $t = T_{max} - 1$ down to 0 do:
            For $(i,j) \in A^R$ do:
                  $t' = \min\{T_{max}, t + d_{ij}(t)\}$
                  if $C[i][t] > C[j][t'] + d_{ij}(t)$ then:
                        $C[i][t] = C[j][t'] + d_{ij}(t)$
                        $N[i][t] = j$
```

The output of DOT has two components. First, we can determine the estimated travel time from any point in the urban network to a certain customer location given that a vehicle departs at a specific time instant. Second, DOT provides the sequence of nodes (representing streets and turnings) that a vehicle must follow in order to arrive from any point of the network to a customer address. The first output helps us determine the optimal routing of the fleet. The latter emulates the use of a GPS device that guides a vehicle to a given destination point.

4. HEURISTICS OF THE DYNAMIC ROUTER AND SCHEDULER

In this section, we describe the heuristics used by the Dynamic Router and Scheduler when a new customer request takes place. We assume that requests are attended one-by-one and that the initial routing plan has been previously computed using some optimization technique as tabu search. First, we describe a method to generate a quick solution to the problem. Then, we use this solution as an initial input for an optimization method based on tabu search.

4.1 Dynamic Insertion Heuristic (DINS).

The heuristic used in the DRS is derived from the basic insertion heuristic proposed in Campbell and Savelsbergh (2004), where every unrouted customer is evaluated at every insertion point. The evaluation of this movement consists in checking the feasibility and profitability of every insertion, which is the negative of the amount of additional travel time added to the route. The customer with a feasible insertion having the maximum profitability is then selected and inserted in the route. The authors proved that the total complexity for this algorithm is given by $O(n^3)$. We have adapted this heuristic for the case of the Dynamic Vehicle Routing Problem with Time Windows (DVRPTW) by introducing some new elements to reflect the dynamics of the operations of a fleet in an urban environment. The objective of DINS heuristic is to insert a new client into the current routes plan once the vehicles have started the services. The general idea of the algorithm is simple. When a new client arrives to the system, the algorithm checks the current state of the vehicles. Then, routes with insufficient time to visit the new client within their schedule are rejected. Finally, the algorithm checks, on the candidate routes, for the least cost feasible insertion. If no feasible insertion is possible and there are idle vehicles on the depot, a new route is created including the new customer.

4.1.1 Notation.

We assume that there are n routed customers in R different routes at the beginning of the time planning horizon. We also suppose that there is one single Depot with a homogeneous fleet of vehicles with capacity Q. A route is a sequence of customers beginning and ending at the Depot. For convenience, the Depot is identified with *0* (start of route) and *n+1* (end of route). The following additional notation is required for the construction of the algorithm:

- $T_{i,j}(t)$ = travel time between customers i and j when vehicle departs at time t.
- V_r = vehicle assigned to route r, $r = 1, 2,..,R$.
- q_r = total demand assigned to route r.
- d_i = demand of customer i.
- E_i = time window lower bound of customer i.
- L_i = time window upper bound of customer i.
- e_i = earliest service start time at customer i.
- l_i = latest service start time at customer i.
- S_i = service time at customer i.

The time window of a customer i is denoted by (E_i, L_i) and we assume that the time window of the depot is given by $(0, T)$, where T is the end of the planning period. The values e_i and l_i, refer to the earliest and latest time a service can take place at customer i, and they must satisfy the following condition: $E_i \leq e_i \leq l_i \leq L_i$. When a new call is received, the system must decide where to insert the new customer. Let us assume that a new customer w arrives to our system at time instant $t > 0$. When this happens, the vehicles of the fleet may have one of the following three states:

1. The vehicle is in service at some customer i (SER).

2. The vehicle is moving to the next planned customer on the route or waiting at the customer location to start service within the time window. (MOV).

3. The vehicle is idle at the Depot, without a previously assigned route (IDL).

We call this situation the *status* of a vehicle. This status will let us know when a vehicle should be diverted from its current route, be assigned to a new one if is idle or keep the planned trip. Whenever a new customer arrives, the status of each vehicle must be known to compute travel times from their current positions to the new customer.

4.1.2 Algorithm.

The first iteration of the algorithm consists of rejecting those routes that are definitely infeasible. To do this, it is necessary to compute the total available time of the routes, i.e. the sum of the available time (if any) of a vehicle on the assigned route. We define the available time of a vehicle as those periods of time where: a) a vehicle is waiting to provide service to a customer or, b) a vehicle has finished the scheduled route and is returning to the depot. This is, in fact, the maximum slack time a vehicle has available to travel and service a new customer call.

The calculation of the available route time can be done by estimating the arrival times at each of the scheduled customer locations that has not been served at the time of the new request *w*. The arrival time a_i at customer *i*, is computed from the current position of the vehicle and we assume that the vehicle will start service as soon as it arrives if the customer's time window is already open; otherwise, the vehicle waits. The calculation of arrival times is done with the following procedure. If, at instant *t*, the vehicle is moving toward the next customer then the estimated arrival time a_i is given by $a_i = t + T_{V_r,i}(t)$. The arrival times at subsequent customers on the route are computed as follows:

$$a_i = \max\{E_{i-1}, a_{i-1}\} + S_{i-1} + T_{i-1,i}(\max\{E_{i-1}, a_{i-1}\} + S_{i-1}) \qquad (1)$$

Hence, the available time of a vehicle in route *r* at some node *i* (including the depot) is the difference between the associated lower limit of the time window and the arrival time of the vehicle to that location. That is, $W_i^r = \max\{E_i - a_i, 0\}$. Therefore, the total available time of a route *r* is given by $AT_r = \sum_i W_i^r$.

From each route, we choose the unvisited node that is the closest to the new client. If the travel time from that node to the new customer is greater than the total available time AT_r, then we reject the route. If every route is rejected, a new route must be created for this new customer. The routes obtained, as a result of this previous iteration, are *possibly feasible* in the sense that vehicles serving those routes have enough time to travel to the new customer. This is a necessary condition, but not sufficient, because we have to check for time windows feasibility. The second major iteration of the algorithm consists, therefore, in checking the feasibility and profitability of an insertion in every arc of the possible feasible routes. The feasible insertion with the highest profitability is then selected and the corresponding route must be updated.

Feasibility. In order to evaluate the feasibility of the insertion of customer *w* between nodes *i* and *i+1*, we must compute e_w and l_w, the expected earliest and latest service start times, respectively. The earliest service start time at the new customer *w* is given by:

$$e_w = \begin{cases} \max\{E_w, t + T_{V_r,w}(t)\}, & \text{if } V_r = i \\ \max\{E_w, e_i + S_i + T_{i,w}(e_i + S_i)\}, & \text{otherwise} \end{cases} \tag{2}$$

The latest service start time is calculated through a backward procedure starting with the depot (the last node to be visited by the vehicle) as follows:

$$l_w = \min\{L_w, \tilde{t}_w - S_w\} \tag{3}$$

where $\tilde{t}_i = \arg\max_t\{t + T_{i,i+1}(t) - l_{i+1}\}, \forall t \le l_{i+1} - T_{i,i+1}(t)$ is the latest possible departure time that allows the vehicle to arrive to the next scheduled customer *i+1* at time l_{i+1}. If $e_w \le l_w$ and $D_w < Q - q_r$, the insertion is feasible.

Profitability. The profit of an insertion is defined as the negative of the additional travel time incurred from inserting the new customer in a route. If a new customer *w* is to be inserted between nodes *i* and *i+1*, the profitability of this insertion is given by:

$$\begin{aligned} profit = -\big[&T_{i,w}(\max\{E_i, a_i\} + S_i) + T_{w,i+1}(\max\{E_w, a_w\} + S_w) - \\ &T_{i,i+1}(\max\{E_i, a_i\} + S_i)\big] \end{aligned} \tag{4}$$

If no insertion is possible and there are available vehicles at the depot, then a new route that includes customer *w* is created. If the whole fleet of vehicles is already occupied, then the call is rejected and postponed for service within the next planning period. Given the arrival of a new client *w* at time *t* and a set *R* of pre-planned routes, the pseudo-code of the heuristic is as follows:

```
1. For each route $r$ in $R$ do:
        Compute available time $AT_r$
        Find travel time from $w$ to closest neighbor
        If available time is not enough, then reject route $r$
        Else, add route $r$ to $R'$, the set of possible feasible routes
2. For each route $r$ in $R'$ do:
        Check the status of the vehicle and set its position as the starting node of the route.
        For each arc ($i$, $i+1$) of the remaining route sequence do:
                If insertion of $w$ is feasible and profit is improved then:
                        Store current profit and insertion places
3. If insertion is possible then:
Insert $w$ in least cost arc and update selected route.
4. Else: create a new route with an available idle vehicle (if any).
```

4.2 Dynamic Tabu Search (DTS).

In order to optimize the solution obtained by the DINS heuristic, we have implemented an adaptation of the *Unified Tabu Search (UTS)* heuristic proposed by Cordeau et al (2001) for the vehicle routing problem with time windows. The heuristic was modified to take into account the dynamic aspects of the problem. Mainly, we have changed the heuristic to perform local searches only on the unvisited nodes of the scheduled routes and added a stopping criterion which takes into account the rate of improvement of the search process. First, we will introduce the UTS heuristic and then we will describe the adaptations made for our algorithm.

4.2.1 The Unified Tabu Search

The UTS meta-heuristic is an iterative procedure that drives the search from a current solution *s* to the best solution found in a subset of the solution space called the neighborhood of *s*, or *N*(*s*), by performing local searches. *N*(*s*) consists of the set of solutions that are reached when a switch operator is applied to the current solution *s*. As it is based on tabu search, UTS uses anti-cycling rules (tabu lists) and diversification mechanisms that allows the search to explore other areas of the solution space while escaping from local optima.
Let *S* be the set of solutions of the vehicle routing problem. Each $s \in S$ satisfies the following constraints: i) every route starts and ends in the depot, and ii) every customer is assigned to exactly one route. During the search process, a solution *s* may be unfeasible as it may violate the time windows or capacity constraints.

For the evaluation of solutions, Cordeau et al (2001) proposed the cost function $f(s) = c(s) + \alpha q(s) + \beta d(s) + \gamma w(s)$, where *c*(*s*) denotes the estimated total travel time of the routes; *q*(*s*) denotes the total capacity violation of the vehicles; *d*(*s*) denotes the total duration violation of the routes; *w*(*s*) denotes the total violation of the time windows constraints, and α, β and γ are positive parameters.

The *N*(*s*) is generated by applying a shift operator to the current solution *s*. A shift operator removes a customer *i* from route *k* and inserts it in route *k'*. In the search process, all insertion positions are evaluated. When customer *i* is removed from route *k*, its reinsertion in that route is forbidden for the next θ iterations. Let *B*(*s*) = {(*i*,*k*): customer *i* is visited by vehicle *k*} denote the attribute set of a solution *s*. To diversify the search, every solution *s'* such that *f*(*s'*) ≥ *f*(*s*) is penalized with a function $p(s') = \lambda c(s')\sqrt{nm}\sum_{(i,k)\in B(s')} \rho_{ik}$, where n is the number of customers, m is the current number of routes, λ is a positive parameter and ρ_{ik} is defined as the number of times the attribute (*i*,*k*) has been added to a solution.

The method begins with the construction of the initial solution by an insertion algorithm. This solution is used by the tabu search procedure as the starting point of the iterative process. In every iteration, the algorithm selects the best non-tabu solution $s' \in N(s)$ that minimizes the values of the cost function *f*(*s*) or satisfies an aspiration criteria. The criteria used by Cordeau et al (2001) is that the tabu status of a solution may be revoked if that would lead to a solution of smaller cost than the currently best solution identified having the attribute *B*(*s*). When the selected solution *s'* is feasible, the value of the best feasible solution is updated and the values of parameters α, β, γ are reduced by dividing the current values by 1 + δ; otherwise, the values are increased by multiplying the current values by the factor 1 + δ.

4.2.2 A Dynamic Tabu Search for the Vehicle Routing Problem with Time Windows.

The proposed heuristic is based on the UTS algorithm, but incorporates the dynamic elements of the problem. As in the DINS algorithm, we first need to know the position of the vehicles and their current state (in transit, in service, waiting or idle). Based on this information, the algorithm uses a modified cost function $f_D(s) = c_D(s) + \alpha q(s) + \beta d_D(s) + \gamma w_D(s)$ that takes into account the current network conditions, where $c_D(s)$ denotes the estimated total travel time of the routes on the pending scheduled customers; $d_D(s)$ denotes the total duration violation of the routes, and $w_D(s)$ denotes the total violation of the time windows constraints of the customers to be visited.

As real-time traffic information is assumed to be available, the computation of $c_D(s)$, the total traveling time, must take into account the departure times of vehicles from a given location. Therefore, before any evaluation of cost function $f_D(s)$, the algorithm must compute an estimation of the arrival and departure time of each vehicle for each one of the scheduled customers in the route. In our algorithm, these calculations assume that a vehicle departs as soon as it finishes a service in a customer, or, as soon as the depot is open. That is, it is assumed that vehicles will follow a *drive-first* policy.

The neighborhood *N*(*s*) of a solution is generated through shift operations with some rules. The possible list of insertion positions in a route starts from the current position of the vehicle and not from the depot. If a vehicle is waiting to start service or heading to a customer's location, the heuristic allows the diversion of the vehicle, that is, the route can be modified to visit a different customer than the originally scheduled. Figure 5 depicts the shift operation in a dynamic environment. In this figure, the shift of customer *i* from route *k* implies that vehicle *k* diverts its current scheduling to visit customer *i*+1. The vehicle in route *k'* is not diverted but is ordered to make a change in his scheduled route.

As in the UTS heuristic, we keep *B*(*s*), the attribute set of solution *s*. The number of iterations that a move is forbidden is given by $\theta = 7.5\log_{10} n_u$, where n_u is the current number of customers that have not been serviced when the algorithm is executed. The choice of this number is based on the results provided by Cordeau et al (2001). During the search process, if a solution $s' \in N(s)$ is such that $f_D(s') \geq f_D(s)$, we penalize it with function $p_D(s') = \lambda c_D(s')\sqrt{n_u m}\sum_{(i,k)\in B(s')} \rho_{ik}$.

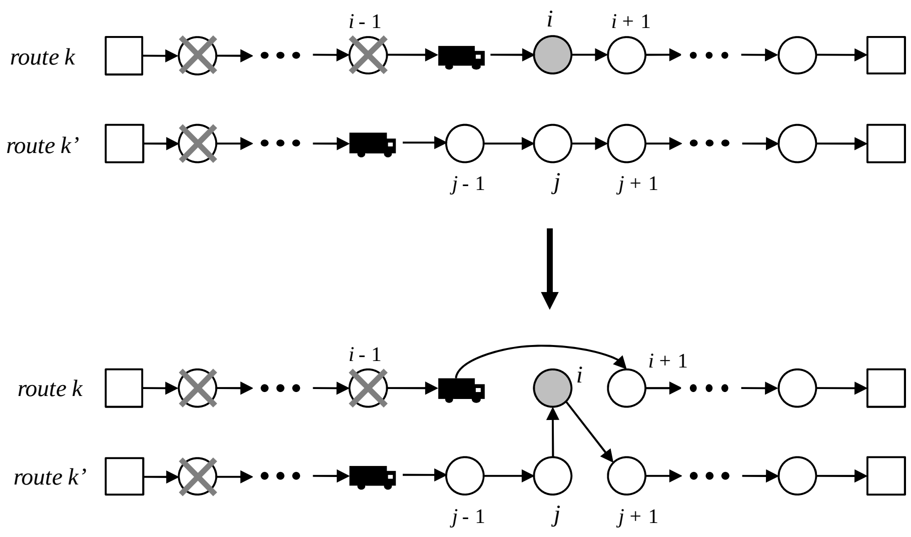


Figure 5 – Shift operation for the dynamic vehicle routing problem

The solution provided by the dynamic insertion heuristic represents the initial solution for the tabu search heuristic. The iterative steps are the same used in UTS, except for two criteria. First, our aspiration criterion accepts a solution if its cost improves the best solution found so far. Second, as fast on-line solutions are required by a fleet manager, we added a criterion that stops the search if a certain number of iterations κ has been reached without observing an improvement larger than ε.

The pseudo-code for the dynamic tabu search (DTS) procedure is described as follows:

```
1. **Find** a solution $s$ using DINS algorithm.
2. **If** solution $s$ is feasible **then** make $s^* = s$, $\alpha = 1$; $\beta = 1$; $\gamma = 1$
3. **For** $i = 1,\dots, \eta$ **do**:
3.1 Choose solution $s' \in N(s)$ that minimizes $f(s') + p(s')$ such that $s$ is not tabu or
satisfies aspiration criteria
        3.2 If $s'$ is feasible and $c(s') < c(s^*)$, then $s^* = s'$; $c(s^*) = c(s')$
        3.3 If $q(s') = 0$ then $\alpha = \alpha/(1+\delta)$, else $\alpha = \alpha(1+\delta)$
    If $d(s') = 0$ then $\beta = \beta/(1+\delta)$, else $\beta = \beta(1+\delta)$
    If $w(s') = 0$ then $\gamma = \gamma/(1+\delta)$, else $\gamma = \gamma(1+\delta)$
3.4 Set $s = s'$
3.5 If (number of iterations without improvement is $\kappa$) and (last improvement < $\varepsilon$) then STOP
```

5. PLATFORM FOR THE DECISION SUPPORT SYSTEM

Our proposed decision support system is based on the combination of dynamic traffic simulation and the algorithms defined on the dynamic router and scheduler. The platform in which all components are integrated is able to emulate the operation of an Advanced Traffic Information System providing real-time data and the operations of fleet management center.

Figure 6 shows the different components of the platform and how they are connected to each other. Next, we will briefly explain the role and interactions of each one of these components.

5.1 Dynamic Traffic Simulation

The first component of our approach is dynamic traffic simulation, which is used as a supporting tool to model the road network and generate databases of time-dependent link travel times. As a modeling tool, dynamic traffic simulation helps us to build and model urban networks by defining the set of sections, intersections, traffic controls, origin-destination matrices and other important characteristics. We also use this tool to generate two types of databases. The first one is created from several replications of a traffic simulation and it represents the historical records of a traffic information database. The second is associated to a single replication and it is used to represent the dynamics of the traffic flows within the simulation of the fleet operational plan. It is important to remark that dynamic traffic simulation has a subsidiary role in our model and its usage is motivated by the fact that it can emulate with great detail and realism the traffic conditions of a road network.

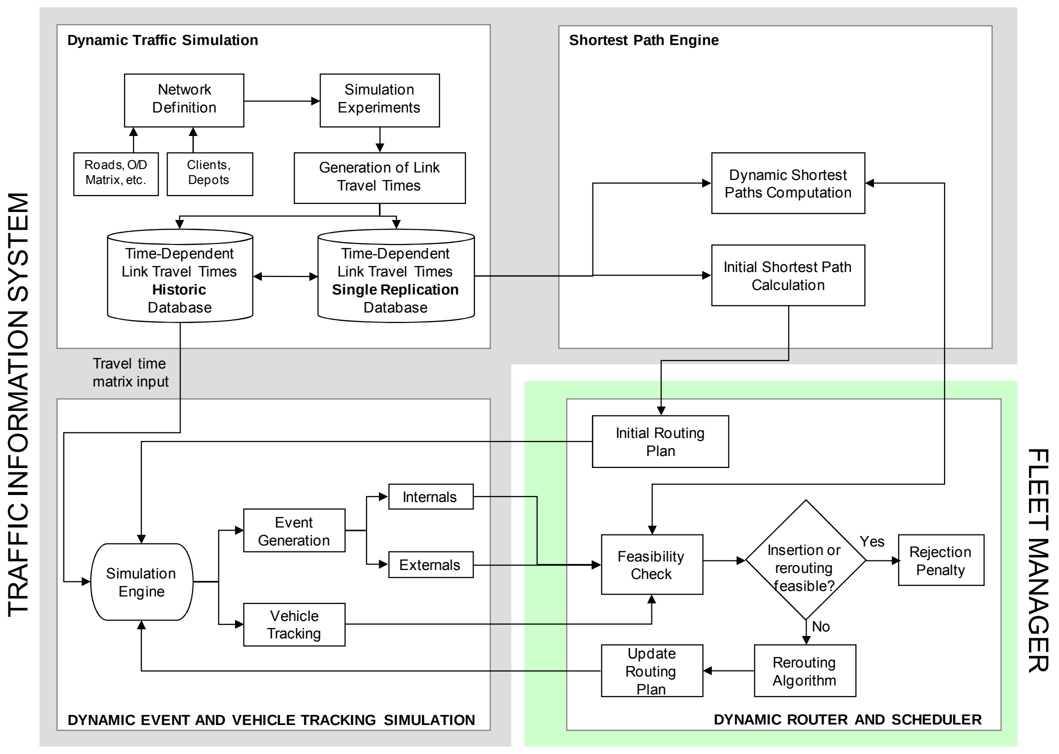

TRAFFIC INFORMATION SYSTEM
Dynamic Traffic Simulation
Network Definition
Simulation Experiments
Roads, O/D Matrix, etc.
Clients, Depots
Generation of Link Travel Times
Time-Dependent Link Travel Times Historic Database
Time-Dependent Link Travel Times Single Replication Database
Shortest Path Engine
Dynamic Shortest Paths Computation
Initial Shortest Path Calculation
Travel time matrix input
Initial Routing Plan
Internals
Event Generation
Externals
Simulation Engine
Vehicle Tracking
Feasibility Check
Insertion or rerouting feasible?
Yes
No
Rejection Penalty
Update Routing Plan
Rerouting Algorithm
DYNAMIC EVENT AND VEHICLE TRACKING SIMULATION
DYNAMIC ROUTER AND SCHEDULER
FLEET MANAGER

Figure 6 – Integrated platform for a decision support system for real-time fleet management.

5.2 Shortest Path Engine

A second component is represented by the shortest path engine, which calculates the optimal paths of the vehicles among customers and the corresponding travel times. A path is represented by the sequence of nodes belonging to the road network that a vehicle follows when travelling among customers. As in the case of a real-life traffic information system, we assume the existence of short and long-term travel time forecasts. Short-term forecast come out of the current replication database and they represent the travel time information for the next 15 minutes. Long-term forecasts, in turn, are provided by the historical database and estimate the travel times beyond the 15-minute period of the short-term forecasts.

The calculations of shortest paths are carried out by an implementation of the DOT algorithm, which receives, as input, the combined information of short and long-term forecasts of travel times. Shortest paths calculations are made whenever a vehicle departs from the depot or from a customer's location and when required by the DRS. DOT provides two types of output: i) the optimal paths followed by the vehicles among the scheduled locations and, ii) the current estimated travel times among customers and vehicles.

5.3 Dynamic Event and Vehicle Tracking Simulation

The core of the simulation is represented by the third component, the dynamic event and vehicle tracking simulator, which i) randomly generates the different external and internal events defined in section 2 and, ii) moves vehicles along the road network according to the operational plan keeping track of their positions and activities at every time step. It is based on the general approach of a next-event time-advance simulation by Law and Kelton (2000) with some additional features according to our needs.

In an initialization step, the simulator loads all the required data for the emulation of the systems, which include the road network, the clients and depot attributes and, the current and historical travel times generated by the dynamic traffic simulation. Once the required data has been loaded, the simulator asks the DRS to calculate an initial route assignment, which is used to set the initial vehicle attributes which are, among others, the sequence of customers to visit and the path on the road network that the vehicle will follow. Additionally, it is set the list of events and their time of occurrence.

A timing routine controls the following event that has been set to occur, advancing the simulation clock and executing the corresponding set of instructions. Whenever there is a change in the simulation clock, the simulator "moves" the vehicles along the network, using the average link travel times from the current replication database. The simulation ends when the simulation clock reaches an ending time.

In this paper, we deal with two types of events: the departure of a vehicle from a customer's location and, the arrival of a new customer request. When a vehicle departs from a location, the simulator records the exit time of the vehicle and computes the time-dependent shortest path to the next customer in the route sequence. The simulator then changes the vehicle's internal state to "in transit" and updates the path on the network that the vehicle must follow to arrive to the next scheduled customer. When a new request is received, the simulator processes and passes all the required information to the DRS, which include current vehicle positions, customers that have not been fulfilled, and the time-dependent travel times computed from the current network conditions. If a solution is found, the route plan is updated accordingly and the next new customer event is scheduled. A penalty may be applied if the algorithm is not able to find a feasible solution that includes the new customer.

5.4 Dynamic Router and Scheduler

The last component of our platform is the DRS, which interacts with the event simulator and the shortest path engine in order to recalculate routes and propose new solutions if needed.
It is important to remark that the proposed platform has been designed in such a way that their components can be replaced by real-time systems while keeping the logic and interactions among customers. For instance, the dynamic traffic simulation component might be replaced by any real-time traffic information system capable of providing travel times and vehicle guidance, while the dynamic event and vehicle tracking simulator can be replaced by any real-time fleet tracking and communication system. Therefore, the framework in which the DRS is developed can be implemented in any fleet management system.

6. COMPUTATIONAL EXPERIMENTS

A set of computational experiments has been designed and conducted to evaluate the performance of a fleet using the simulation platform to emulate daily traffic city conditions, under different scenarios. For this paper in particular, we are interested in assessing: i) the added value of real-time traffic information in the performance of the fleet and, ii) the quality of the solutions provided by the proposed algorithms when new customer orders are received during the operational period.

All algorithms and the integrated simulation platform have been implemented in C++. The computational tests described in this section were executed on a PC with Intel Core 2 Duo 3.16Ghz and 3.25 GB in RAM.

6.1 Test Instances.

For our computational tests, a microscopic simulation model of the downtown area of Barcelona has been used. It covers 746.5 hectares and it is one of the most important districts in the city due to the high volume of commercial activity (see Figure 7). The graph that represents the road network consists of 1,570 nodes and 2,797 arcs.

Real-time Traffic Information System
In order to emulate the real-time traffic information system of the urban network, we generated a database of historic travel times through Aimsun, a microscopic traffic simulation software. This database, which represents the long-term travel time forecasts, contains the average travel times of all links on the network for every 5-minute interval of 15 replications of a simulation under normal traffic conditions. We additionally generate a set of 12 replications that are used by the dynamic event and vehicle tracking simulator to move vehicles along the network. These replications also represent the short-term travel time forecasts that combined with the long-term forecasts of the historic travel time database

provides the input to the time-dependent shortest paths calculations that are used to guide vehicles through the network.

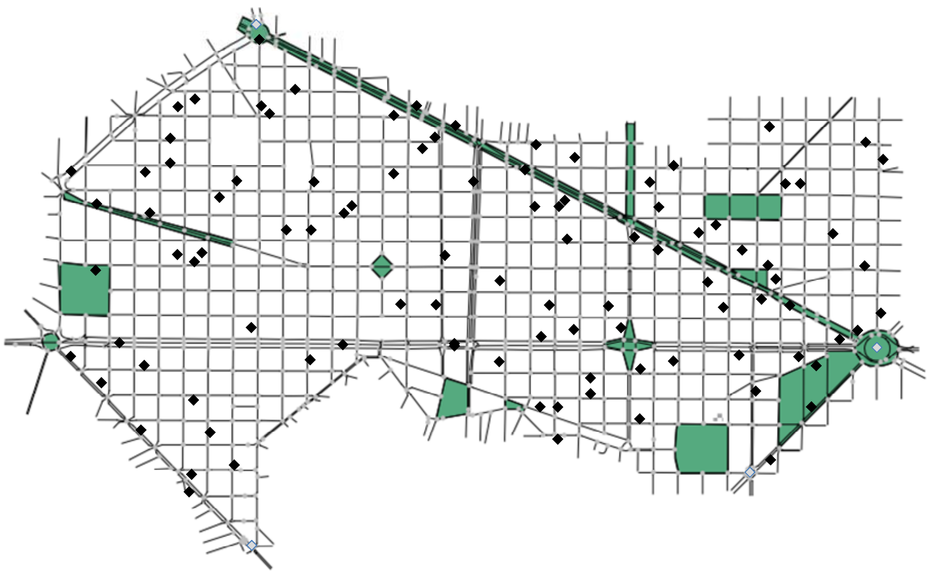


Figure 7 – Barcelona's Downtown Network

Customer Datasets
We have generated two datasets of 100 customers with different geographic locations on the network. The first dataset contains customers that are randomly distributed across the network (see Figure 7 where customers are represented by the black dot). For the second dataset, we have used commercial data from the Council of Barcelona to generate customers that are distributed in clusters. Besides their geographic location, we also distinguish customers by their time windows characteristics, which can be: i) *narrow*, that have a time length between 0.5 and 1 hour, and ii) *wide*, with a time length between 4 and 6 hours. For both datasets, we assume that there is a logistics center with one depot and a fleet of 7 homogenous vehicles with a capacity of 1,000 units. The depot service runs from 08:00 to 16:00 hours. It is also assumed that all customers have a constant demand of 10 units and a service time of 10 minutes[1].

Scenarios
We have defined two main scenarios. In the first one, we assume that all demands are known in advance and that there are no new customer requests. We then compare the performance of the fleet when routing plans are computed and carried out assuming two types of traffic information: static and real-time time-dependent travel times. In the second scenario, we deal with the dynamic problem where new customer orders arrive during the day. We assume four degrees of dynamism (20%, 40%, 60% and 80%), which is defined as the proportion of new orders that are received by the fleet manager during the day. In this

[1] Beginning June 1st, 2010, datasets can be downloaded in the following webpage: http://www-eio.upc.es/~jorozco/

scenario we compare the performance of the fleet when a greedy heuristic (DINS) is used to insert new orders in the routes and when a metaheuristic (DTS) is used for this purpose. The following table summarizes the considered scenarios and the corresponding factors.

Table 1– Main scenarios for the computational experiments

Scenario	Factors	Levels
No new customers requests (S1)	Link travel times	Static; Time-dependent
	Shortest paths calculations	At the beginning; When vehicle departs
	Time windows width	Narrow (0.5 - 1 hrs); wide (4 - 6 hrs)
New customer requests (S2)	Link travel times	Time-dependent
	Shortest paths calculations	When a vehicle departs; when a new client arrives
	Time windows width	Narrow (0.5 - 1 hrs); wide (4 - 6 hrs)
	DRS solving algorithm	DINS; DINS + DTS
	Degree of dynamism	20%; 40%; 60%; 80%

In order to compare scenarios, the following key performance indicators have been defined:

- Service level (*SL*).
- Average travel time of vehicles (*ATT*).
- Average waiting time of vehicles (*AWT*).

The *SL* is defined as the percentage of scheduled customers that were visited by the vehicles within the time windows constraints. *ATT* is the average time that a vehicle spends traveling in the network. *AWT* is the average time that a vehicle spends waiting at a customer location to start service.

Initial Operational Plan Computation

Before starting simulations, we proceeded to calculate the initial routing plan for each one of the instances using a variation of the UTS heuristic (Cordeau et al, 2001). Unlike this heuristic, we have used the insertion algorithm of Campbell and Savelsbergh (2004) to generate an initial solution. When then applied the UTS iterative process with the following parameters:

Table 2 – Parameters used by the UTS to compute initial routes

Parameter	Description	Value
α	Coefficient of capacity violation function (initial value)	1.00
β	Coefficient of duration violation function (initial value)	1.00
γ	Coefficient of time windows violation function (initial value)	1.00
δ	Update parameter of violation functions parameters	0.50
θ	Number of iterations for which a move is forbidden	$7.5\log_{10}(100)$
λ	Diversification parameter in penalty function	0.015
η	Maximum number of iterations	10000

We assume two different kinds of inputs to calculate routes in scenario S1: static and time-dependent travel times. When static data is assumed, routes are computed using the average travel times observed during a 10-hours period. That is, we assume that travel times among customers remain static during the planning period. Therefore, vehicles are guided through the network with shortest paths computed from static information. When time-dependent information is assumed, routes are calculated using the historic average travel times observed for different departure times. In this case, vehicles are guided by time-dependent shortest paths which depend on the estimated departure time from a customer.
In real-life applications of vehicle routing models, companies usually estimate their travel times by adding a "buffer factor" to their calculations in order to protect themselves against unexpected variability in the traffic conditions. Therefore, we additionally tested the performance of the vehicles when a buffer factor is considered. In our case, we have set a buffer factor of 1.20, which is the average ratio of the observed mean and standard deviation in all links of the network. Thus, when computing routes, travel time estimates were increased in 20%.

6.2 Results of Scenario S1

We first simulated the initial routing plans assuming that there are no new customer orders during the operational period. In this case, the only source of variability faced by the fleet is given by the current traffic conditions in the service area. We assume that vehicles always follow shortest paths between customers in their route sequence. However, we use two different approaches related to shortest path calculations. First, we assume that there is no real-time traffic information system providing on-line travel times. Therefore, vehicles are guided through the network using static shortest paths computed with average travel time data. The second approach considers that real-time time-dependent link travel times are available and they are used to compute new time-dependent shortest paths that guide vehicles to their next destination.

An additional assumption is that vehicles depart as soon as they are ready from their current position following a *drive-first* policy. That is, after finishing a service, a vehicle departs immediately to the next assigned customer.

Customers in random locations

We first analyze the results for the set of instances where customers are randomly distributed along the service area. Table 3 shows the results obtained under normal traffic conditions.

Table 3 – Simulation results of scenario S1 with customers randomly distributed

Travel Times Data	Calculation Base	SP Calculation Frequency	Number of Customers	Time Windows	Number of Routes	Total Travel Time (secs)	Total Waiting Time (secs)	Service Level
Static	Average	Once	100	Narrow	5	16631.2	52188.8	100.0%
Time-Dependent	Average	When departing	100	Narrow	5	15752.2	51411.8	99.8%
Static	Average + 20%	Once	100	Narrow	5	16536.6	50325.3	100.0%
Time-Dependent	Average + 20%	When departing	100	Narrow	5	16000.2	47487.7	100.0%
Static	Average	Once	100	Wide	3	9579.6	5222.8	99.9%
Time-Dependent	Average	When departing	100	Wide	3	9270.0	5675.2	99.1%
Static	Average + 20%	Once	100	Wide	4	9088.3	15186.5	100.0%
Time-Dependent	Average + 20%	When departing	100	Wide	4	8802.9	24037.6	100.0%

From the table, we can observe that the total travel time experienced by the fleet is always lower when time-dependent data is available and shortest paths are updated when a vehicle departs to the following destination in its scheduling. On average, the total waiting time was reduced in 4% when real-time traffic information was available. Regarding service level, we found that, although routing plans were simulated under normal traffic conditions, there were some customers that couldn't be serviced, but we found that introducing the buffer factor in the estimates of travel times had a positive effect in the service levels as they reached 100% for both type of time windows. Nevertheless, in the case of wide time windows, this decision resulted in the addition of one more vehicle to the routing plan.

Customers distributed in clusters

Table 4 shows the results for the set of instances where customers are distributed in clusters in the network. In this case, we observed that real-time traffic information helped to reduce by 9% the total travel time of the vehicles, but only when the calculation base of the routing plans are the average estimates. Introducing the buffer factor in these instances also had a positive effect in the service level in the case of time windows. However, it can be observed that there was no significant difference in the total travel times between vehicles guided with static data and vehicles guided with real-time traffic information.

Table 4 – Simulation results of scenario S1 with customers distributed in clusters

Travel Times Data	Calculation Base	SP Calculation Frequency	Number of Customers	Time Windows	Number of Routes	Total Travel Time (secs)	Total Waiting Time (secs)	Service Level
Static	Average	Once	100	Narrow	4	15229.8	32836.7	99.9%
Time-Dependent	Average	When departing	100	Narrow	4	13858.3	33997.9	99.6%
Static	Average + 20%	Once	100	Narrow	5	14688.9	57670.3	100.0%
Time-Dependent	Average + 20%	When departing	100	Narrow	4	14957.9	32615.4	100.0%
Static	Average	Once	100	Wide	4	8833.9	18937.7	100.0%
Time-Dependent	Average	When departing	100	Wide	3	8076.6	6254.8	100.0%
Static	Average + 20%	Once	100	Wide	4	8048.0	19159.9	100.0%
Time-Dependent	Average + 20%	When departing	100	Wide	4	7980.8	21497.3	100.0%

6.3 Results of Scenario S2

In the next set of experiments, we evaluate the performance of the fleet in the presence of new customer orders considering various degrees of dynamism. We assume that the new customer orders arriving to the fleet manager follow an exponential distribution with a mean of 5 minutes. The sum of initial and dynamic customers is assumed to be 100. For each instance, there is a set of customers whose demand is known in advance. The initial routing plan for these customers is built using the UTS heuristic as described in section 6.1. In order to achieve a 100% service level, we have used a protection or buffer factor against variability of 20% in the estimated travel times among customers. During simulation, shortest paths calculations are made after a vehicle finishes a service and when a new order assignment takes place.

We used two approaches to assign customers to the routes: the dynamic insertion heuristic (DINS) and, the dynamic tabu search heuristic (DTS). In preliminary tests of the DTS heuristic, we observed that best-found solutions are usually no longer improved after 1,000 iterations of the algorithm. Therefore, we set DTS to stop the search process at 1,250 iterations or, if after 500 iterations, the solutions has not been improved in at least 1%. When a new order arrives, the simulation is temporarily stopped to calculate the new assignment. Simulation is resumed once the new solution is found and routes are updated. In our experiments, we have assumed that diversion is allowed, that is, if a vehicle is traveling or waiting at a customer location, a vehicle can be diverted to provide service to a new customer.

In order to measure the quality of the solutions obtained by these two methods, we also evaluate the performance of the algorithms against the *value of perfect information* (VPI), which can be defined as the fleet performance that would be observed if we knew the demand in advance when planning routes. We test our algorithms in the datasets of customers with random and clustered geographic locations.

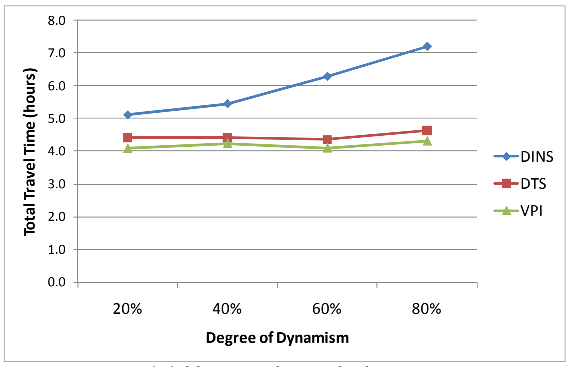
Customers in random locations
Table 5 shows the simulation results for the different degrees of dynamism when customers are assumed to be randomly distributed in the service area. Let us first consider the case of customers that require narrow time windows. When the degree of dynamism is relatively small (20%), assigning a new order with DTS reduces about 14% the total travel time of the fleet than when using a greedy heuristic as DINS. The gap between these two methods increases along with the degree of dynamism. In problems with 80% of dynamism, we observed that DTS could improve the results obtained by DINS in 36%. A similar pattern was observed in the case of wide time windows where DTS found solution that, on average, improved the total travel time experienced by the fleet in 50%.

Regarding the number of vehicles required by both methodologies, we observed that solutions proposed by DTS were always equal or lower for all degrees of dynamism. Although the objective function in DTS is in terms of travel time, the shift operation in the iterative process drives the search to “more balanced” solutions which require, on average, a lower number of vehicles.

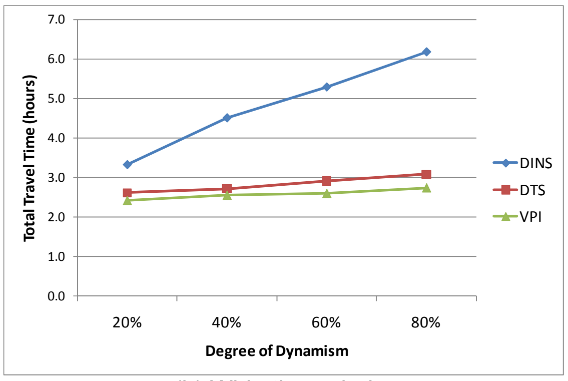
Table 5 – Simulation results of scenario S2 for customers in random locations

Customer Distribution	Degree of Dynamism	Time Windows	Solving Algorithm	Initial Routes	Final Routes	Total Travel Time (secs)	Total Waiting Time (secs)	Service Level
Random	20%	Narrow	DINS	4.0	4.0	18,400	28,465	100.0%
Random	40%	Narrow	DINS	3.0	4.0	19,625	30,795	100.0%
Random	60%	Narrow	DINS	3.0	6.0	22,629	24,711	100.0%
Random	80%	Narrow	DINS	2.0	7.0	25,926	40,032	100.0%
Random	20%	Narrow	DTS	4.0	4.0	15,892	29,918	100.0%
Random	40%	Narrow	DTS	3.0	4.1	15,890	34,610	100.0%
Random	60%	Narrow	DTS	3.0	5.5	15,678	33,171	100.0%
Random	80%	Narrow	DTS	2.0	6.3	16,672	38,600	99.9%
Random	20%	Wide	DINS	3.0	4.0	11,981	19,607	100.0%
Random	40%	Wide	DINS	2.0	3.9	16,223	10,706	100.0%
Random	60%	Wide	DINS	2.0	5.0	19,020	26,241	100.0%
Random	80%	Wide	DINS	1.0	6.3	22,205	26,478	100.0%
Random	20%	Wide	DTS	3.0	3.0	9,400	10,040	100.0%
Random	40%	Wide	DTS	2.0	3.0	9,765	12,317	100.0%
Random	60%	Wide	DTS	2.0	4.0	10,456	14,329	100.0%
Random	80%	Wide	DTS	1.0	5.0	11,071	15,795	100.0%

In order to test the quality of the solutions provided by the DTS we compared them with those obtained when we assume that perfect information is available. It is expected that perfect information will lead to near-optimal solutions as routes can be more carefully planned at the beginning of the operational period. From Figure 8, we can observe that DTS lead to relatively good results when compared to those obtained when perfect information is available. On average, DTS solutions were 6% and 8% larger than those obtained under VPI for the narrow and wide time window cases, respectively. In particular, we observed that in problems with a relatively high degree of dynamism (more than 60%), DTS seems to perform better in the case of narrow time windows than in the case of wide time windows.



(a) Narrow time windows



(b) Wide time windows

Figure 8 – Fleet performance for different degrees of dynamism – Customers in random locations

Customers distributed in clusters

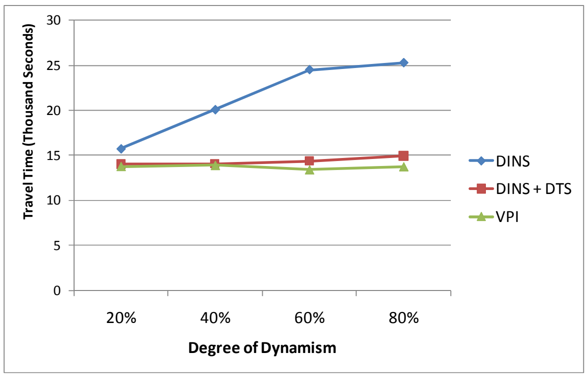
Simulation results when customers are distributed in clusters are shown in Table 6. As in the case of random locations, the gap between the solutions found by DINS and DTS algorithms increases along with the degree of dynamism. In the case of narrow time windows and for relatively low levels of dynamism (20%), results show that DTS, on average, found solutions that were 11% lower than those found by DINS. In the case of wide time windows, this difference is larger as DTS improves the solution in about 26%. In problems with a relatively high degree of dynamism, DTS was also found to outperform the solutions of DINS in 41% and 51% in the case of narrow and wide time windows, respectively.

Table 6 – Simulation results of scenario S2 for customers distributed in clusters

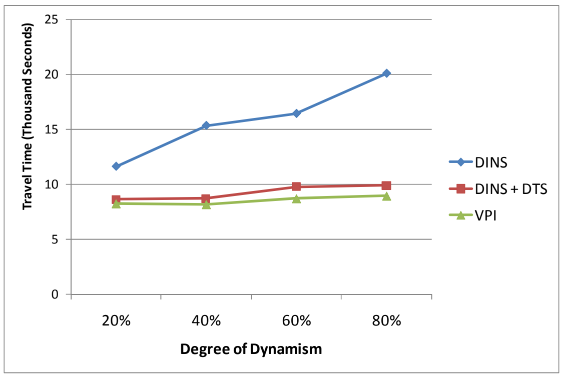
Customer Distribution	Degree of Dynamism	Time Windows	Solving Algorithm	Initial Routes	Final Routes	Total Travel Time (secs)	Total Waiting Time (secs)	Service Level
Clustered	20%	Narrow	DINS	4.0	5.0	15,763	45,349	100.0%
Clustered	40%	Narrow	DINS	3.0	5.0	20,083	41,049	99.9%
Clustered	60%	Narrow	DINS	3.0	7.0	24,503	57,254	100.0%
Clustered	80%	Narrow	DINS	2.0	7.0	25,269	38,575	99.3%
Clustered	20%	Narrow	DINS + DTS	4.0	4.1	14,034	33,928	100.0%
Clustered	40%	Narrow	DINS + DTS	3.0	4.9	14,062	42,289	100.0%
Clustered	60%	Narrow	DINS + DTS	3.0	6.1	14,365	46,313	100.0%
Clustered	80%	Narrow	DINS + DTS	2.0	6.9	14,928	40,368	99.9%
Clustered	20%	Wide	DINS	3.0	4.0	11,605	16,587	100.0%
Clustered	40%	Wide	DINS	2.0	4.0	15,332	17,815	100.0%
Clustered	60%	Wide	DINS	2.0	5.0	16,421	24,445	100.0%
Clustered	80%	Wide	DINS	1.0	6.0	20,076	24,908	100.0%
Clustered	20%	Wide	DINS + DTS	3.0	3.3	8,611	11,534	100.0%
Clustered	40%	Wide	DINS + DTS	2.0	3.5	8,692	12,928	100.0%
Clustered	60%	Wide	DINS + DTS	2.0	4.0	9,775	13,714	100.0%
Clustered	80%	Wide	DINS + DTS	1.0	4.8	9,882	15,621	100.0%

.

Figure 9 compares the simulation results of the dynamic solving strategies with the value of perfect information. Unlike the case of customer randomly located, we found that for a relatively low degree of dynamism DTS is closer to the solution found with perfect information. When narrow time windows are considered and the degree of dynamism is less or equal than 40%, the average gap between DTS and VPI was 1.5%. In the case of wide time windows, this gap added up to 5.4% on average. This result suggests that for very low degrees of dynamism, a fast insertion heuristic can be used to obtain good results without the need of more advanced algorithms that may require significant computational efforts. For a relatively larger degree of dynamism (80%), this gap was 8% in the case of narrow time windows and, 9.7% when customers are characterized by wide time windows.



(a) Narrow time windows



(b) Wide time windows

Figure 9 – Fleet performance for different degrees of dynamism – Customers distributed in clusters

7. CONCLUSIONS AND FUTURE RESEARCH

This paper provides an insight of the integration of vehicle routing models with real-time traffic information systems as a decision support platform. We propose a platform for the evaluation of city logistics applications, in the case of the vehicle routing problems with time windows and a new approach to deal with the dynamic case, through fast insertion heuristics and tabu search, when new customer requirements arrive during the operational period. The results obtained in the simulations proved that, when creating an initial routing plan, although feasible solutions are originally obtained based on some historical average data, it may happen that not all clients will be served within their time windows due to changes in traffic conditions during the day.

Additionally, when dealing with new customer orders in the vehicle routing problem with time windows, we observe that for very low degrees of dynamism, fast insertion heuristics can be used as solving methodologies without a significant loss in the solution quality. Furthermore, when dealing with high dynamism, we observed that tabu search can be used to improve the solution obtained by a fast insertion heuristic. The results suggest that the proposed tabu search algorithm has a better performance when wide time windows are considered due to the higher degree of flexibility to shift customers from one route to another than in the case of narrow time windows

The results obtained in this paper assume that the underlying road network presents expected traffic conditions during the simulation. A further step on our research is the evaluation of the solving strategies when traffic conditions are suddenly altered (e.g. traffic incidents). Another issue to be investigated is the optimal calibration of parameters and data structures in order to improve the performance of the algorithms.

8. REFERENCES

Campbell, A.M., Savelsbergh, M. (2004). Efficient insertion heuristics for vehicle routing and scheduling problems. Transportation Science, 38 (4), pp. 369—378.

Chabini, I. (1998). Discrete dynamic shortest path problems in transportation applications. Complexity and algorithms with optimal run time. Transportation Research Record, 1645, pp. 170-175.

Chabini, I., Dean, B. (2006). Shortest path problems in discrete-time dynamic networks: complexity, algorithms and implementations. Technical Report, Massachusetts Institute of Technology, Cambridge, USA.

Chen, H.K., Hsueh, C.F., Chang, M.S. (2006). The real-time time-dependent vehicle routing problem. Transportation Research Part E, 42: 383—408.

Cooke, K.L., Halsey, E. (1996). The shortest route through a network with time-dependent internodal transit times, Journal of Math. Anal. Appl., 14: 492-498.

Cordeau, J.F., Laporte, G., Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows. Journal of the Operational Research Society, 52, pp. 928-936.

Fleischmann, B., Gietz, M., Gnutzmann, S. (2004). Time-Varying Travel Times in Vehicle Routing. Transportation Science, 38(2): 160—173.

G. Ghiani. G, Guerriero, F., Laporte, G., Musmanno, R. (2003). Real-time Vehicle Routing: Solution concepts, algorithms and parallel computing strategies. European Journal of Operational Research, 151: 1—11.

Ichoua, S., Gendreau, M., Potvin, J.Y. (2003). Vehicle dispatching with time-dependent travel times. European Journal of Operational Research, 144: 379—396.

Law, A.M., Kelton, W.D. (2000). Simulation modeling and analysis. 3rd edition, McGraw-Hill series in industrial engineering and management science .

Malandraki, C., Daskin, M.S. (1992). Time dependent vehicle routing problems: formulations, properties and heuristics algorithms. Transportation Science, 26 (3): 185—200.

Orda, A., Rom, R. (1990). Shortest-path and minimum-delay algorithms in networks with time-dependent edge length. Journal of the Association for Computing Machinery, 3(3): 607—625.

Potvin, J.Y., Xu, Y., Benyahia, I. (2006). Vehicle routing and scheduling with dynamic travel times. Computers and Operations Research, 33: 1129—1137.

Regan, A.C., Mahmassani, H.S., Jaillet, P. (1997). Dynamic decision making for commercial fleet operations using real-time information. Transportation Research Record, 1537.

Regan, A.C., Mahmassani, H.S., Jaillet, P. (1998). Evaluation of dynamic fleet management systems: simulation framework. Transportation Research Record, 1645.

Robusté, F. (2005). Logística del Transporte. Edicions Universitat Politécnica de Catalunya, First Edition.

Tang, H. (2008). Efficient implementation of improvement procedures for vehicle routing with time-dependent travel times. Transportation Research Record, 2089.

Ziliaskopoulos, A.K., Mahmassani, H.S. (1993). Time-dependent, shortest-path algorithm for real-time intelligent vehicle highway system applications. Transportation Research Record 1408, pp. 94-100.