Approaches for Tackling Dynamic Vehicle Routing Problems

Michel Gendreau
CIRRELT and MAGI
Polytechnique Montréal

EWO Seminar
Carnegie-Mellon University
Pittsburgh – November 1st, 2016

Acknowledgements

The work described was performed with a large number of co-authors, in Montreal and abroad:

- Jean-Yves Potvin
- Soumia Ichoua
- P. Badeau, F. Guertin, E.D. Taillard, R. Séguin
- M. Tagmouti, N. Azi
- F. Ferrucci, S. Bock
- V. Pillac, C. Guéret, A. Medaglia
- And more...

Outline

1. Introduction
2. A simple approach
3. Dealing with some key issues
4. More involved problems and approaches
5. Recent references
6. Future research paths

INTRODUCTION

What is Dynamic VRP?

- An extension of classical vehicle routing problems in which the some of the problem data is not deterministic and known in advance!
- Furthermore, routes must be updated while they are being performed, as new information is obtained.
- Usually, the main information which is divulged over time is the set of requests to be serviced by the fleet of vehicles.
- But, this is by no means limitative:
  - Vehicle breakdowns
  - Changes in traffic conditions (weather, congestion, etc.)
  - Customer demands

Important developments

- Development of Global Positioning System
- Introduction of two-way communications with drivers (smartphones and other devices)
- Rapid increase of computing power
- Recent algorithmic developments
  - It is now possible to consider deploying highly efficient fleet management systems.
  - (An interesting area to apply parallel computing technique!)

Two key dimensions

- Information evolution:
  - Input known beforehand (Static)
  - Input changes over time (Dynamic)
- Information quality
  - Deterministic input
  - Stochastic input
- The four combinations can be encountered in real-life applications
  - DVRPs correspond to situations where the input changes over time

DVRPs vs. Stochastic or Robust VRPs

- Most of the Stochastic VRP models encountered in the literature correspond to the "static and stochastic" case of the previous taxonomy
  - In particular, those based on the "a priori" optimization paradigm.

- Robust routing models are also exclusively defined in the context of the "static and stochastic" setting.

Some applications

- Typical applications:
  - Local pickup for long-distance courier companies
  - Local pickup and delivery (courier)
  - Local pickup for LTL companies
  - “Dial-a-ride”
  - Travelling repairmen
- Less typical applications:
  - Dispatching and routing of winter maintenance vehicles (M. Tagmouti)
  - Management of papers “second delivery” (Ferrucci and Bock)
  - E-supermarket deliveries (N. Azi)

Related (fleet management) problems

- Managing fleets of emergency vehicles (ambulances)

- Dynamic vehicle allocation problems
  - Truckload trucking (Powell)
  - Containers
  - Etc.

Measuring dynamism

- Two dimensions:
  - The frequency of changes
  - The urgency of requests
- ***Degree of dynamism***: the ratio
  (number of dynamic requests) / (total number of requests)
  (Lund et al., 1996)
- ***Effective degree of dynamism:*** normalized average of request disclosure times (Larsen, 2001)
- ***Level of urgency:*** normalized average of request reaction times (latest service time - disclosure time) (Larsen, 2001)

A SIMPLE APPROACH

General idea

- No comprehensive (and complex) model of the whole dynamic problem.
- The dynamic problem is tackled by solving continuously a series of related **STATIC** problems.
- The static problems only use the information currently available (requests).
- This idea will be illustrated on a couple of problems.

Local pickup for long-distance courier companies

DISPATCHING AND ROUTING

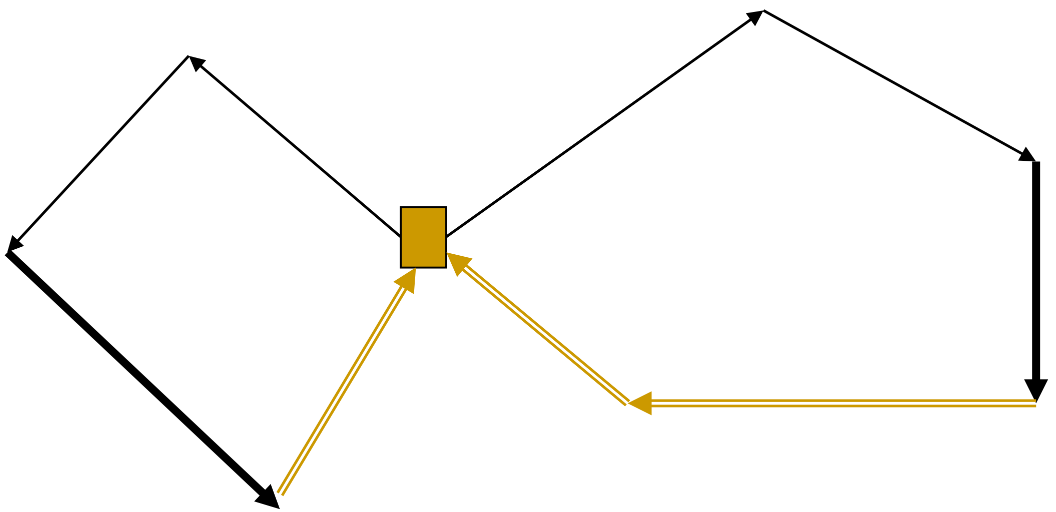
Local pickup only operations

- Customers phone the dispatch centre to ask for the pickup of small parcels and envelopes.
- Each request is characterized by:
  - a geographical location
  - an earliest pickup time (rigid)
  - a latest pickup time (flexible → lateness penalties)
- No capacity constraints.
- Vehicles must return to the depot at a fixed time (rigid constraint).
- Requests can be turned down, if servicing them would imply a late arrival at the depot.

SOLUTION STRATEGY

- A "current best solution" is maintained at all times: set of feasible routes servicing the accepted requests.
- Each vehicle moves towards the first unserviced request in its planned route (this decision cannot be changed unless diversion is allowed).
- The static problem solved at anytime is a VRP with soft time windows (rigid at the depot) and additional constraints (current destination of each vehicle).
- The static problem is modified
  - when a new request is accepted,
  - when a vehicle completes a pickup.

ILLUSTRATION

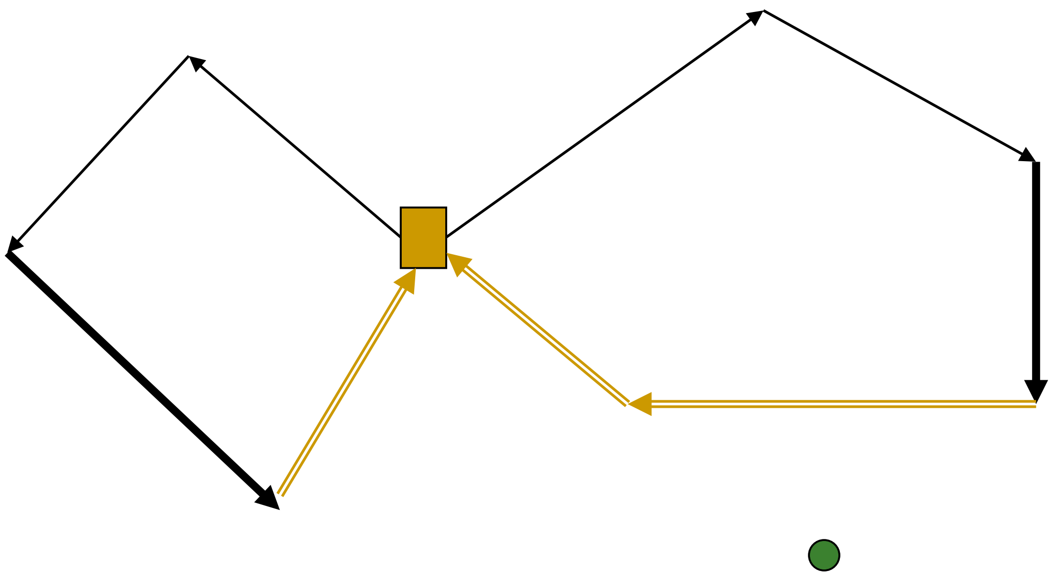


Movements already executed

Movement currently performed

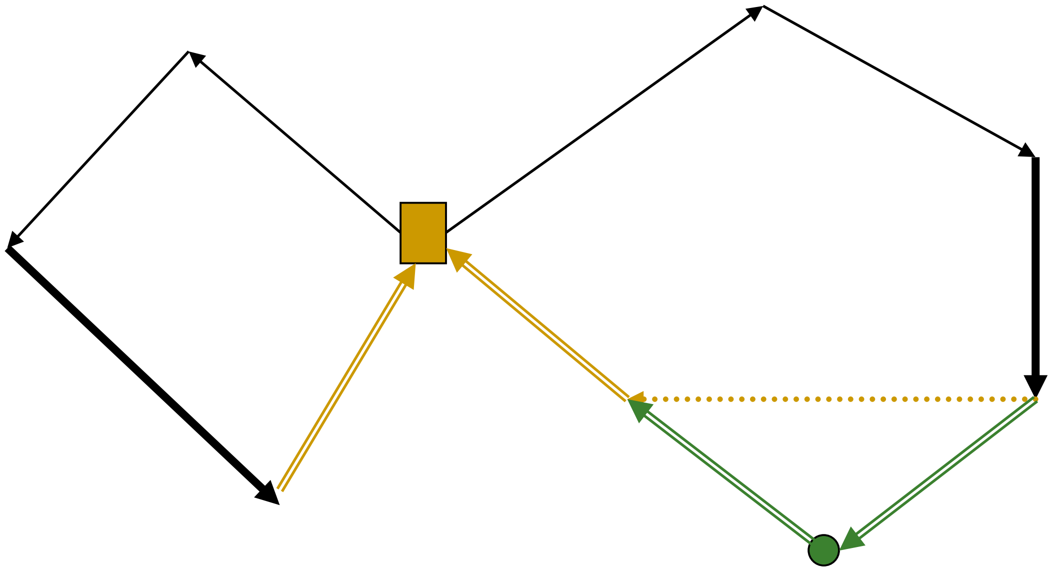
Planned movements

INSERTION OF A NEW REQUEST



- New request

INSERTION OF A NEW REQUEST

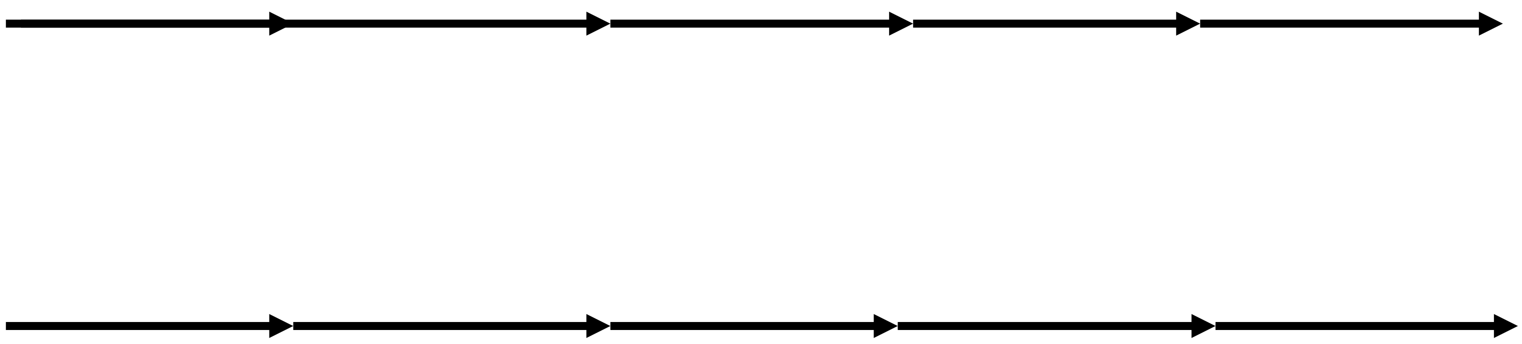


- New request

SOLVING THE CURRENT STATIC PROBLEM

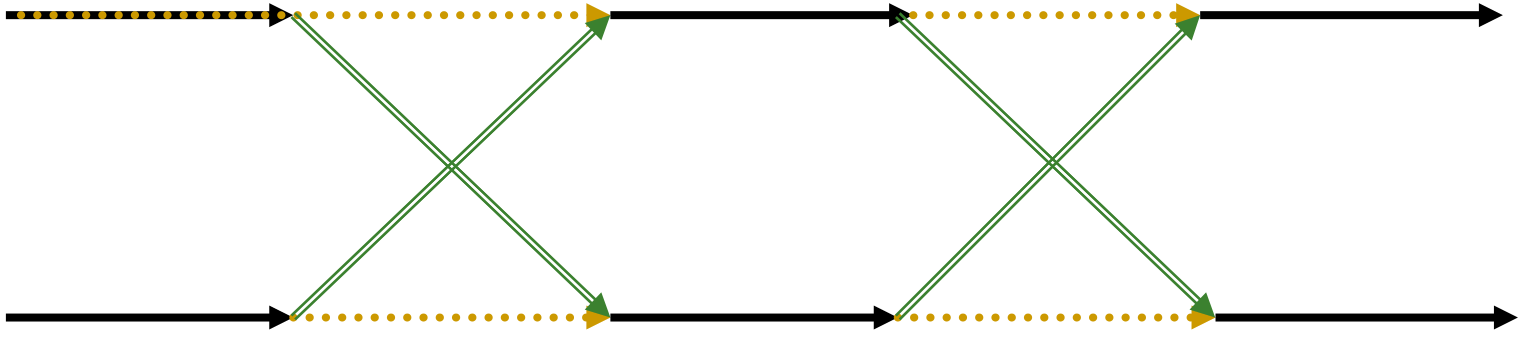
- One could simply insert new requests in the current routes, but it is better to re-optimize the planned portion of the routes.
- Various heuristics from the simplest to the most sophisticated can be used.
- Simple heuristics are cheap to implement, but the sophisticated ones may require using parallel computing.
- We used a parallel tabu search method with adaptive memory and geographical decomposition:
  - R simultaneous search threads on the whole problem
  - Each thread decomposes the problem into D geographical subproblems.

CROSS-EXCHANGE



becomes

CROSS-EXCHANGE



CROSS-EXCHANGE

- Preserves the orientation of the routes.
- Special cases : - 2-opt* (Potvin and Rousseau)
  - Or-opt
- Because of soft time windows, move are always feasible BUT lateness penalties are difficult to evaluate.

  → We use an approximation.

DECOMPOSITION

- Based on Taillard's (1993) work for standard VRP's.
- Problem space is partitioned into sectors around the depot which contain approximately the same number of routes.

ADAPTIVE MEMORY

- Rochat and Taillard (1995)
- Pool of routes from previously visited solutions sorted according to the objective value of the solution (can accomodate up to *T* routes).
- New solutions are created, one route at a time, using biased sampling.

  With *T’* routes,

$$p_R = \frac{2(T'-k+1)}{T'(T'+1)}.$$

- Routes with one or more customers in common with the current partical solution are ignored.
- May terminate with some unrouted customers.

PARALLEL IMPLEMENTATION

- For static VRPTW's solution quality (for a given number of calls to the adaptive memory) is unaffected by the number of search threads.

$\longrightarrow$ effective parallel search scheme!

COMPUTATIONAL EXPERIMENTS

- Test problems:
  - based on Solomon's problems
  - 50% of requests known beforehand
  - arrival times of other requests are randomly generated in $[0, \tilde{a}_i]$

    where $\tilde{a}_i = \min \begin{bmatrix} a_i, \text{ earliest service time for customer } i \\ d_{i-1}, \text{ departure time from } i\text{'s predecessor in best known solution} \end{bmatrix}$
  - use the minimum number of vehicles.

- Competing heuristics:
  - simple insertion of new requests upon arrival
  - reconstruction of routes using Solomon's I1 heuristic after each request arrival
  - insertion of new requests using Solomon's criterion followed by descent to the first local minimum using the CROSS exchange.

ASSESSING SOLUTIONS

- Route length
- Time window penalty
- Number of rejected requests

COMPUTATIONAL RESULTS

Typical results obtained with 1 request per minute:

	Insertion	Insertion + descent	Recon-struction	Tabu Search
Distance	1350	1081	1157	1039
Delay	485.2	55.2	171.6	30.7
Rejected customers	2.05	0.57	0.57	0.09

Local pickup and delivery

THE PROBLEM

- To dispatch the vehicles of a local courier company same day pickup and delivery of small size goods.
- Fleet of *m* vehicles (fixed size).
- Requests are received continuously.
- For each request, we have
  - a pickup location
  - a delivery location
- For each location *i*, we are given
  - a time-window to begin service $[e_i, \ell_i]$
  - a service time s_i
- Vehicles must return to a central depot for a specified time.
- No capacity constraints on vehicle loads.

THE OBJECTIVE FUNCTION

- Minimize a weighted sum of:
  - total travel time (distance),
  - lateness penalty at request locations,
  - an overtime penalty.

$$\text{Lateness}_i = \max(0, (\text{arrival time at } i) - \ell_i);$$

$$\text{Overtime}_k = \max(0, (\text{arrival time at depot})_k - \ell_o^k)$$

- All requests received are served

$\longrightarrow$ no penalty for unserviced requests

MODUS OPERANDI

- Vehicles have at all times a *planned route*.
- When they are received, requests are assigned to (inserted in) the planned route of one of the vehicles.
- After completing service at a location, a vehicle starts moving towards the next location on its planned route) (*This cannot be changed* $\longrightarrow$ *no diversion*).
- Requests can be moved to another route until their assigned vehicle has started moving towards their pickup location.
- Requests whose pickup location has already been visited can have their delivery location rescheduled within the same route.

SOLUTION APPROACH

- Similar to the one used by Badeau, Gendreau, Guertin, Potvin & Taillard for pickup only case.
- Requests are inserted in the current (best) planned solution as soon as they are received.
- The planned solution is re-optimized (and updated) between request arrivals
  - *solely on the basis of the currently available information.*
  - *no provision for future requests which have yet to arrive.*
- Approximate costs formulas are used to evaluate potential modifications to routes (exact evaluation is too demanding).

MODIFYING A SOLUTION

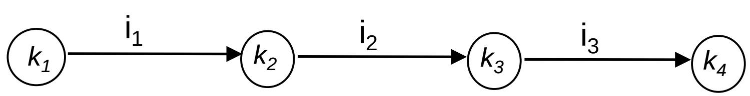
We consider two types of modifications (moves):

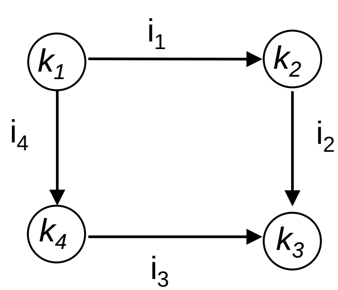
- ***Intra-route moves***
  - Except for the next location to be visited, any unserviced location may be rescheduled within its assigned route subject only to “pickup precedes delivery” constraint.
- ***Inter-route moves***
  - Apply only to requests for which both *pickup and delivery locations belong to the planned portion of the route.*
  - *Based on the concept of ejection chains* proposed by Glover and used by Rego and Roucairol.
  - Basic idea: move a request to a new route, forcing out a request which is moved to a third route, where a third request is force out to a fourth route, and so on…

EJECTION CHAINS

- An ejection chain may involve several routes (all of them, if desired)

  BUT a route may appear only once in a chain, except for the first one.
- *Open chains* : chains where no request is ejected from the last route involved.
- *Closed chains* : chains that stop by having a request inserted in the route of the request which started the chain.





A closed (cyclic) chain

INSERTING A REQUEST IN A ROUTE

- The two best pickup positions within the route are determined using an approximation of the objective function.
- For each of those two pickup locations, the best delivery location is determined using the approximation of the objective.
- $\longrightarrow$ we obtain two pairs (p_1, d_1) and (p_2, d_2) of tentative positions within the route.
- The exact cost of inserting the request is evaluated for the following 18 combinations of pickup and delivery positions

$$(p_1 \pm 1, d_1 \pm 1) \qquad (p_2 \pm 1, d_2 \pm 1)$$

- The combination yielding the lowest cost is stored.

CHAIN EVALUATION

For evaluation purposes, a chain can be broken down into its component links.

There are two types of links:

- *(i, j)* with *i* and *j* being requests in different routes
  - saving obtained from removing *i* of its current route;
  - additional cost of inserting in the route of request *j*, after removing *j* from it.
- *(i, k)* with *i* a request and *k* a route
  - to complete open chains (no request is ejected from route *k*).
  - similar to the above procedure, but the current route *k* is used for the insertion step.
- Savings resulting from extractions are computed exactly.

CHAIN SELECTION

- If there are n movable requests and m routes, we obtain an n x n and an n x m cost matrices. Some entries of these matrices will, in general, be negative.
- Selecting the best chain which visits every route at most once amounts to solving a variant of a *Selective Generalized Traveling Salesman Problem*.

  $\longrightarrow$ certainly NP-hard!
- We use instead a heuristic which is based on an adaptation of the Floyd-Warschall all-pairs shortest path algorithm.

  $\longrightarrow$ The two cost matrices are used to construct an n x $(n+m)$ matrix of path (chain) costs.

  $\longrightarrow$ $0(n^2(n+m))$.

ADAPTIVE MEMORY

- A scheme originally proposed by Rochat and Taillard (1995) for VRPTW.
- Was used in TS heuristic for real-time dispatch of pickup only vehicles (1996).
- Idea : use a central repository (memory) to store the routes making up good solutions.
  - These routes can be used to rebuild new solutions from which descent or tabu search can be restarted.
  - Natural communication mechanism to share information between search threads running in parallel.
- Reconstructing a solution from memory:
  - Biased sampling from routes in memory (routes chosen must be disjoint).
  - Complete by inserting left over requests into the new solution.

COMPUTATIONAL EXPERIMENTS

- Abstract city setting.

- Generation program creates realistic instances for this setting:
  - Request arrival time
  - Request locations (pickup and delivery)
  - Request time windows (pickup and delivery)
  - Some requests may be known at the beginning of the day.

- Vehicle dispatching simulator:
  - Operates in real time.
  - Tracks down all events.
  - Runs in parallel with the solution processes.

SOLUTION STRATEGIES

- INSERTION + GREEDY DESCENT
  - Insert new request.
  - Apply local improvement operator until a local optimum is found.
- RECONSTRUCTION + GREEDY DESCENT
  - Reconstruct solution from “scratch”.
  - Apply local improvement operator as above.
- ADAPTIVE DESCENT
  - Insert new request.
  - Greedy descent to local optimum.
  - Apply descent and loop.
  - Build new solution from adaptive memory.
- TABU SEARCH
  - with adaptive memory;
  - with parallel search threads;
  - using spatial decomposition.

COMPUTATIONAL RESULTS

- Computational results show that the most refined approaches (adaptive descent and tabu search) perform much, much better than naïve ones.

- It turns out that adaptive descent does extremely well, because it is a method very well adapted to this class of problems.

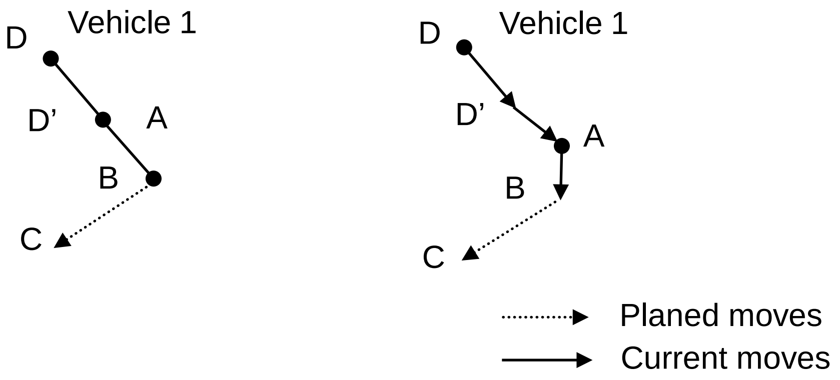
DEALING WITH SOME KEY ISSUES

MAKING THE APPROACH MORE REALISTIC AND FLEXIBLE

- Diversion
- Time-dependent travel times
- Generic information on future demands

DIVERSION

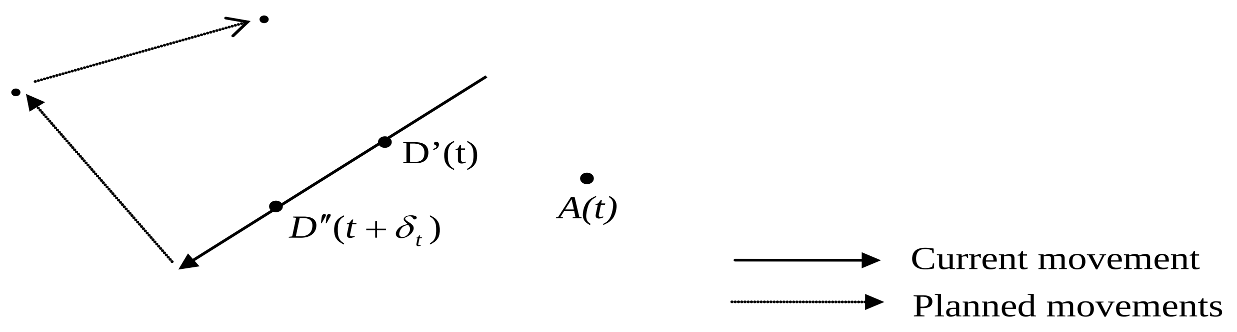
- Consider the possibility of changing the destination of a moving vehicle to go service a newly arrived request.



- A more general form : reconsider the current destination and planned routes after the arrival of new request.

IMPORTANT CONSIDERATIONS

- Assessing diversion opportunities requires time for computation.
- One must project oneself in time to assess diversion correctly.
- Critical question : how much time should be allowed for computation?



Time projection

SETTING δt

- Rule 1 : $\delta t \leq \min(t_i) - t$

  where t_i is the time at which service will begin at the current destination of vehicle *i* (too restrictive)

- Rule 2 : $\delta t \leq \alpha_1 T^-$

  where T^- is the moving average of recent interarrival times of requests

- Rule 3 : $\delta t = \alpha_2 X / \ell_X$

  where *X* is a time horizon and ℓ_X is the number of requests on the current planned route during *X*.

COMPUTATIONAL EXPERIMENTS

- Solomon's benchmark problems
- Two scenarios:
  - 50% of requests known in advance
  - 25% of requests known in advance
- 15 minute time horizon
  - 3 requests/min in scenario 1
  - 5 requests/min in scenario 2
- Network of 9 SUN UltraSparc workstations (300 MHz) under PVM

COMPUTATIONAL RESULTS

- Results show that diversion is useful, if properly calibrated.
- Rule 3 is much more effective than others.

TIME-DEPENDENT TRAVEL TIMES

- Travel times are not constant during the day.
- Using several values of travel times for different time periods leads to inconsistencies (violation of FIFO assumption).
- Instead, use different values of *travel speeds*!
- Assign travel speeds per subset of similar links and time period.
- Fairly easy to integrate in the approach described so far.
- Leads to much improved solutions if there are important variations of travel times.

INTEGRATING FUTURE DEMANDS

- Experienced dispatchers take into account general knowledge about demand patterns when dispatching vehicles.
- They will not move vehicles away from high-demand zones even if there are currently no requests in these zones.
- How to replicate this?
- Use statistical information about demand patterns (w.r.t. location and time of the day).
- After a vehicle completes a service request, hold it at its current location if the expected demand within some neighbourhood and a small time horizon is high enough.
- Simulation experiments show that holding is seldom used, but does provide small, but significant improvement

MORE INVOLVED PROBLEMS AND APPROACHES

The second paper delivery problem
(joint work with F. Ferrucci and S. Bock)

THE PROBLEM

- To dispatch a fleet of vehicles that deliver a second paper to subscribers whose “first” paper was stolen or blown away by the wind.
- Speed of delivery after receiving the phone call request is critical.
- Requests are received continuously.
- For each request, we have
  - a delivery location,
  - the arrival time of the request,
  - a time window, which is defined by a maximum allowed response time that is the same for all requests.
- If service begins after this time window, high penalty costs occur.

THE SOLUTION APPROACH

- A tabu search solves a VRPTW problem at fixed time points using information on unserviced requests + ***dummy customers***.
- The dummy customers are generated according to a statistical analysis of past requests in different sectors ***at different times***.
- They must be updated as time goes by.

- Simulations based on real application data showed that the usefulness of incorporating statistical knowledge did vary from instance to instance.

Planning winter maintenance routes
(joint work with M. Tagmouti and J.-Y. Potvin)

Planning e-market deliveries
(joint work with N. Azi and J.-Y. Potvin)

THE PROBLEM

- To dispatch the delivery vehicles of an e-supermarket.
- Fleet of *m* vehicles (fixed size) with given capacity.
- Requests are received continuously.
- For each request, we have
  - a delivery location
- Because of perishability considerations any order must be delivered within a rather short time from the departure of the e-supermarket.
  - Delivery routes must be short!
  - Several routes must be assigned to each vehicle to form a workday.

THE OBJECTIVE FUNCTION

- Because of the problem constraints, it might not be possible to service all customers.

- The objective is hierarchical:
  - Serve as many customers as possible;
  - Minimize distance of the driven routes.

SOLUTION STRATEGY

- The non-dynamic version of the problem is solved using an Adaptive Large Neighborhood Search heuristic that was specially designed for the problem.
- The dynamic version is solved by exploiting this heuristic, but not only with the known requests.
  - A set S of s-solutions based on possible scenarios of future requests, is used to evaluate the profitability of new requests.
  - These scenarios are based on some probabilistic knowledge.
  - The scenarios are updated as time passes to reflect actual request arrivals and routes being performed.
  - New requests are accepted only if they are expected to be profitable.

ASSESSMENT

- A simulator was built to test the proposed method on randomly generated instances with 3 to 5 vehicles.

# vehicles	# cust.	# served cust.	% served cust.	# routes per workday	# cust. per route	Profit	CPU (s)
3	72.2	44.5	61.8	2.8	5.5	447.9	0.6
	108.4	55.4	51.2	3.3	6.0	564.7	1.7
	144.2	56.1	39.0	3.3	5.8	579.0	2.7
5	72.2	51.6	71.8	1.7	6.3	534.3	1.2
	108.4	69.4	64.1	2.3	6.2	710.8	3.8
	144.2	82.3	57.2	3.0	5.6	855.3	7.6

Table 1: Simulations of 4 hours with the myopic approach, fleets of 3 and 5 vehicles and an increasing number of customers

# vehicles	# cust.	# served cust.	% served cust.	# routes per workday	# cust. per route	Profit	CPU (s)
3	72.2	48.6	67.8	2.1	8.1	508.9	204.7
	108.4	57.5	53.1	2.1	9.2	606.9	272.8
	144.2	62.6	43.6	2.4	8.6	676.3	471.1
5	72.2	55.6	77.3	1.5	7.8	587.0	305.6
	108.4	70.2	64.9	1.6	8.9	745.0	733.4
	144.2	83.1	57.9	1.8	9.2	901.6	963.7

Table 2: Simulations of 4 hours with the non myopic approach, fleets of 3 and 5 vehicles and an increasing number of customers

RECENT REFERENCES

Some useful survey papers/chapters

- V. Pillac, M. Gendreau, C. Guéret, A.L. Medaglia, (2013), "A review of dynamic vehicle routing problems", *European Journal of Operational Research* 225(1), 1-11.

- T. Bektas, P.G. Repoussis, C.D. Tarantilis (2014), "Dynamic Vehicle Routing Problems", *Vehicle Routing: Problems, Methods, and Applications, Second Edition*, P. Toth and D. Vigo (Eds.), SIAM, pp. 213-239.

- U. Ritzinger, J. Puchinger, and Richard F. Hartl, (2016), "A Survey on Dynamic and Stochastic Vehicle Routing Problems", *International Journal of Production Research* 54(1), 215-231.

FUTURE RESEARCH PATHS

Where are we going?

- Given modern communication methods, it is obvious that fleets can really be controlled in real-time.

- Among other things, it should now be possible to really track closely the vehicles.

- In the context of Big Data, the proliferation of available data on all aspects of DVRPs and their environment opens vast perspectives for improving dispatching in real-time, but insightful analysis will be required to determine how this would best be done.

Thank you for your attention!