Chapter 3

Literature Review

This chapter includes a detailed literature review of publications on Dynamic Fleet Management. At the beginning, some general statistics on the surveyed literature are given (Section 3.1). This is followed by some exemplary publications showing the variety of dynamic applications in real-life (Section 3.2). Afterwards, algorithm orientated papers are presented categorized into three groups depending on the knowledge of the future (Section 3.3). The remaining sections review the most popular dynamic test instances in the literature (Section 3.4) and outline the results of some papers that do not primarily focus on the algorithmic performance, but on the acceptance of dynamic planning applications in real-life (Section 3.5).

3.1 Statistical Analysis of the Surveyed Publications

Dynamic aspects in transportation have attracted increasing attention in the research community and in practice over the last years. A dynamic transportation problem was *considered first* by Wilson (Wilson et al., 1971; Wilson and Weissberg, 1976; Wilson and Colvin, 1977) at the Massachusetts Institute of Technology (MIT) in Boston. For a dynamic Dial-A-Ride Problem at the city of Rochester (USA), the authors develop an insertion heuristic, which, after the occurrence of a new order, evaluates all possible insertion positions in the existing tours. According to a special selection criterion, the new order is inserted at the best position.

After this early work, it took several years for *dynamic transportation* to become a popular field of research. Today, many publications are available, investigating dynamic real-life applications and proposing new efficient solution methods. Figure 3.1 shows the number of publications concerning "dynamics in transportation" over the *course of time*, based on the 64 sources cited in this survey.[4] In the mid-nineties, the number of publications started to increase, reaching a peak of 12 publications in 2004.[5] Afterwards, a medium level was maintained but with a decreasing trend.

---

[4] Other surveys with classification schemes and literature different from this survey have been published, for example, by Psaraftis (1995), Ghiani et al. (2003), Cordeau et al. (2007), and Larsen et al. (2008).

[5] This finding may be attributed to a special issue on "Real-Time Fleet Management" in Transportation Science in 2004

Steffen Schorpp, *Dynamic Fleet Management for International Truck Transportation,*
DOI 10.1007/978-3-8349-6675-9_3,

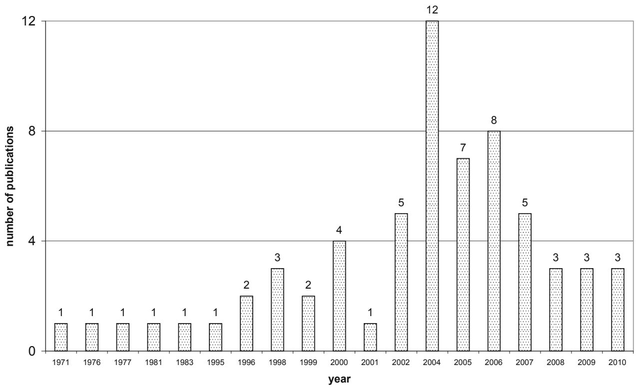

Figure 3.1: Publications on dynamic transportation over the course of time

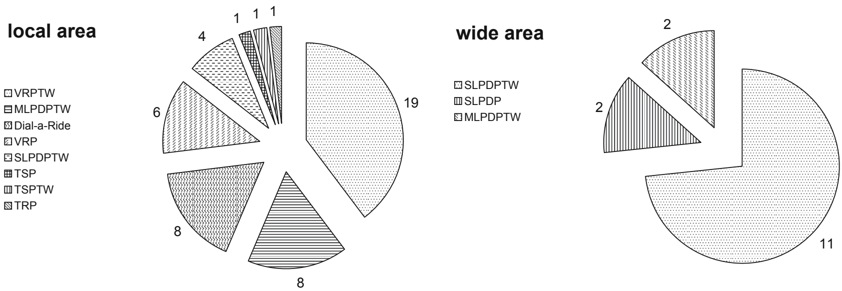

Figure 3.2: Standard problems considered in the selected dynamic publications

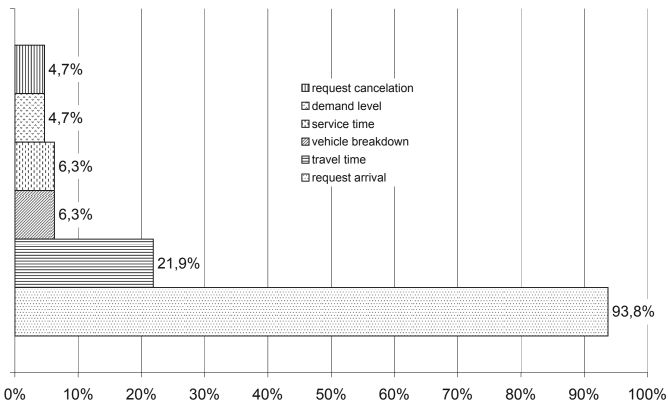

Figure 3.3: Sources of dynamism considered in dynamic publications (in %)

It is also interesting to analyze which *basic theoretical problems* the surveyed publications have dealt with (cp. Figure 3.2). In total, 48 papers considered local area applications, 15 considered a wide area environment, and one paper could not be attributed. Within the local area group, 27 papers are depot bound (VRP, VRPTW, TSP, TSPTW) and 21 papers are depot free (TRP, SLPDPTW, MLPDPTW, Dial-A-Ride). All wide area applications are depot free (SLPDPTW, SLPDP, MLPDPTW).

When evaluating *sources of dynamism* (cp. Figure 3.3), it can be observed that nearly all publications (93.8%) consider the dynamic arrival of requests. Also quite popular are dynamic travel times (21.9%), while vehicle breakdown, service time (each 6.3%), dynamic demand levels, and request cancelation (each 4.7%) are only considered by a few authors.

Before algorithmic solution procedures come to the fore in Section 3.3, several publications that focus specifically on dynamic practical applications will be presented.

3.2 Practical Applications

The following publications primarily focus on dynamic applications in real-life. They give a detailed description of the associated practical planning problems and present solution approaches that have actually been implemented. However, additional real-life applications with distinctive focus on algorithmic solution concepts can also be found in Section 3.3. This section's purpose is to outline the variety of dynamic real-life applications, which, for example, can be seen at the different objects or services provided by vehicles.

The first five papers (see Table 3.1) consider various objects (from petroleum to human patients) that have to be transported. A further paper deals with vehicles providing roadside assistance service, while, in the last application, customer service is constituted by the use of a specific recreational vehicle itself.

authors	**investigated real-life topic**
Brown and Graves (1981)	dispatching of petroleum tank trucks
Bell et al. (1983)	distribution of industrial gases in a VMI environment
Savelsbergh and Sol (1998)	truck dispatching at shipping company
Magalhaes and Sousa (2006)	distribution of pharmaceutical products
Beaudry et al. (2010)	transportation of patients in a hospital
Krumke et al. (2002)	dispatching of mobile roadside assistance units (ADAC)
Ernst et al. (2007)	dispatching of recreational vehicles

Table 3.1: Variety of dynamic real-life applications

Brown and Graves (1981) consider *fleet dispatching of petroleum tank trucks at "Chevron Corp."*. Starting from 80 US terminals, 2,600 loads per day have to be scheduled, delivering motor gasoline, weed oils and jet fuels. Vehicles perform a series of successive "single load" pendular tours between depot and customers. A vehicle consists of several compartments that have to be filled with different types of gasoline. The order quantity is assumed to be static, while new orders occur dynamically. The objective is the minimization of transportation costs and an equitable workload distribution.

The planning problem has to be solved for each depot in a rolling horizon framework. At first, authors model an *integer linear program*, but find out that it cannot be solved exactly within a reasonable time. This is because there is just a single central computer that has to handle a subproblem query from one of the 80 terminals within a fraction of a second.

Hence, a *heuristic* is proposed, which solves a sequence of embedded network flow problems and successively fixes order to truck assignments. Subsequently, the solution is *improved by load exchanges* between two trucks and *Best Re-Insertion*. The new dispatch module produces excellent solution quality and a reduction of transportation costs by about three percent. Additionally, the new planning approach achieves extremely uniform distributions of workload among vehicles.

Bell et al. (1983) report on a *dynamic distribution problem (capacitated VRPTW) at "Airproducts and Chemicals Inc."*, which sells industrial gases (oxygen, nitrogen, argon and carbon monoxide) from 23 depots to 3,500 customers in the US. Ten to thirty vehicles are assigned to each depot. The inventory of storage tanks, located at the customer locations, is monitored by the distributor (Vendor Managed Inventory) and must be maintained above a specified safety stock level.

The customer's demand is dynamic. The only indication for future demand levels is an estimation based on a historical 15-day horizon. Customers with high variability can additionally be phoned to detect their exact inventory levels. On the basis of this demand information, vehicle routes, schedules and quantities for Delivery are planned for each depot. Nevertheless, demands may deviate from the estimated amounts. In addition, unaccounted emergency orders, which have to be served immediately, may occur during the day. The authors model the problem as a *mixed integer problem* and develop a *Lagrange relaxation based algorithm*, which is solved in a rolling horizon manner (with a two to five days horizon). As a result, the authors observe savings of up to $1.72 million annually when replacing the current manual planning with the new planning system.

Beaudry et al. (2010) investigate *dynamic transportation of patients between health care units and service areas in a large hospital in Southern Germany (dynamic Dial-A-Ride)*. The hospital complex consists of 100 buildings and a road network of 15 km. A heterogeneous fleet of 11 ambulances, each carrying special equipment, is responsible for picking up and delivering people in given time windows. Some people require individual transportation, other people can be combined. A vehicle may carry different load combinations: one bed, one wheelchair, one seated person at a time, or up to three wheelchairs. Some transportation tasks require the vehicle to go back to the depot for disinfection afterwards. Other transportation tasks require the Pickup of an accompanying person, who needs to be picked up before the patient and perhaps needs to be brought back after the completion of the transport.

Ninety-six percent of the requests are called in dynamically. Further possible dynamic events are cancelations and updates of requests, as well as late arrivals and vehicle breakdown. The objective function prioritizes patient convenience over both travel time and prevention of early arrival. The authors develop a rolling horizon *Tabu Search metaheuristic* based on the *neighborhoods best re-insertion* and *intraroute re-arrangement*. Tests with a 20-day historical data horizon from the hospital reveal significant reductions in waiting times for patients and a reduction of the number of vehicles.

Krumke et al. (2002) examine a *dynamic service vehicle dispatching problem at the German Automobile Association (ADAC)*, which provides roadside assistance to people whose car has broken during their journey. Service vehicles possess individual capabilities (spare parts, repair kit), cost parameters and home location. In addition to ADAC owned vehicles, it is also possible to access subcontractor's vehicles. The problem can be considered as a multi depot Vehicle Routing Problem, with dynamic requests (100%) and dynamic service times. If a "customer" calls in, the objective is to guarantee service within a short period of waiting time (lateness cost), while keeping operational costs for the service vehicles (driving cost, overtime cost) as low as possible.

The authors propose a *Column Generation approach*, which is applied in a heuristic way. By solving linear subproblems, dual prices are obtained and are used to generate feasible tours with reduced cost. For a test set with 770 events and 200 mobile units, solution quality is within 5% from optimum within 15 seconds and within 2% from static optimum after one minute.

Savelsbergh and Sol (1998) describe a *dynamic planning problem at "Van Gend and Loos BV", the largest road transportation provider in Benelux.* The paper is focused on direct transportation (no consolidation of orders at a depot or hub), which is carried out for order sizes ranging from four pallets to a full truckload. The problem can be considered as an MLPDPTW, with the special characteristic that one request can have several Delivery locations in a predetermined order. Van Gend and Loos BV exclusively use rented vehicles (on average 100) for direct transportation: 50 are rented permanently and the remaining vehicles are rented on a daily basis. The number of vehicles that are rented on a daily basis has to be specified at the beginning of a working day.

The primary planning objective is to minimize the number of vehicles. The secondary objective is the minimization of total traveled distance. Especially the estimation of the right number of vehicles is difficult, because just 40% of a day's orders are known in advance, while 60% arrive during the day of execution. The authors develop a *Column Generation* based solution approach. Since fast reaction times ($<$ 5min) have to be ensured, the underlying pricing problem is solved by an approximization algorithm. Encouraging results of the new solution approach are reported, leading to reductions in total costs ranging from 3.7% to 4.7% a week. On the other hand, the number of vehicles used is slightly higher than before.

Magalhaes and Sousa (2006) deal with a *dynamic application at Cofanor, a distributor of pharmaceutical products, operating in Portugal.* As pharmacies organize stock with Just-In-Time policies, they tend to place several orders with rather small quantities during a single day. These orders are digitally transferred to Cofanor, where a human operator confirms them. Afterwards, a picking process in the distributor's warehouse is started, until the orders (on average 400 per day) can be distributed by vans. The objective is a quick response to customer demands (short lead-time) and keeping Delivery costs (traveled distance) low.

The authors propose a four-phase heuristic (for a capacitated VRP). First, the orders are *clustered by increasing angle*. Then, a route of orders is generated in each cluster using *Best Insertion*. Thirdly, the urgency of all routes is checked and only routes with an "urgent" order are released to phase four (an order is "urgent" if the time between order placement and expected Delivery is greater than a predefined threshold). In phase four, the chosen routes are improved with 2-opt and finally released for execution. The postponement of tours in phase three is used with the intention of receiving further compatible orders. Results of the new heuristic are compared to the results of manual planning: average lead time to pharmacies is reduced by 8.1%, however, traveled distance is increased by 1.9%.

Ernst et al. (2007) report on a *dynamic planning problem at Tourism Holdings Limited, a New Zealand-based company that operates a fleet of more than 4000 recreational vehicles (motor homes and camper vans)* at 10 locations in Australia and 4 locations in New Zealand. The vehicle fleet consists of 50 distinct vehicle types, e.g. varying in the number of berths, number of doors and power engine. The dispatching of vehicles is performed on a 200-day "active scheduling horizon" and can be described as an SLPDPTW. When a customer calls in, an *acceptance/rejection-decision* is made within five seconds. If the request cannot be accepted, alternatives have to be suggested, e.g. different dates or similar products. A further planning task is the adjustment of the plan to dynamic events, like late return, vehicle breakdown, etc.

The authors employ *two planning levels*: the objective of the *first level* is to maximize the number of accepted bookings, while, in the *second level*, the cost for handling the accepted bookings is minimized. Operation costs consist particularly of empty relocation, free upgrade to a higher valued vehicle, accelerated cleaning to hold appointed allocation time, etc. The dynamic first level problem is solved with a linear assignment algorithm. For the second level, a *relaxed linear programming formulation* is solved with *ILOG*. Afterwards, the relaxed conditions are heuristically re-incorporated. Idle time between dynamic events is used to improve the current plan. The application of the new system resulted in 2% savings in operating costs. Simultaneously, human planners won time to intensify their efforts handling exceptions.

In most cases, practical applications do not perfectly coincide with one of the standard problems. Because many additional requirements often have to be accounted for, this section's primary function has been to show the widespread appearance of dynamic problems in real-life. The following section will cover algorithmic procedures (usually related to standard problems), which provide a pool of generic concepts that can be adapted to more specific practical applications.

3.3 Algorithmic Solution Concepts

The reviewed publications within this section can be divided into three groups. The first two groups do not have any knowledge of the future and therefore only perform "myopic" planning. In contrast to the first group (Section 3.3.1), the second group (Section 3.3.2) anyhow tries to anticipate the future. Stochastic information about the future is available only to third group publications (Section 3.3.3), which make explicit use of it with different concepts.

3.3.1 Dynamic Approaches without Knowledge of the Future

There are generally many possible ways of grouping dynamic myopic publications: by *investigated standard problems*, by *sources of dynamism*, by *degree of dynamism*, by *geographical area*, by *associated groups of authors*, etc. In the following, however, the *main algorithmic solution concepts* have been chosen for classification:

- **local search approaches** (3.3.1.1),
- **metaheuristics**, guiding the local search out of local optima (3.3.1.2),
- **heuristic applications of exact procedures** (3.3.1.3),
- **rule-based approaches** (3.3.1.4), and
- **multi-agent systems** (3.3.1.5).

At the beginning of each of the following subsections, a short summary of the selected publications' properties is given in the form of a table (see Table 3.2 for an example). This table includes the dynamic aspects considered in the associated publication.

An "X" in the first column indicates dynamically occurring requests. If available, the degree of dynamism is given in subsequent brackets. The second column reports on further sources of dynamism (e.g. travel time). Afterwards, the considered standard problem is specified: capacitated/uncapacitated (column four), actual problem (column five), time window characteristic hard/soft - if time windows are available at all (column six), and geographical extension (column seven). An "X" in column eight indicates that en route diversion is allowed, and column nine gives information about the employed dynamic test data sets.

	dynamic aspects		**capaci-tated**	**problem**	**TW constr.**	**area**	**en route diversion**	**dynamic test data**
	orders	**other**						
Shieh and May, 1998	X (50%)		cap.	VRPTW	hard	local		Solomon, 1987
Du et al., 2005	X (100%)		cap.	VRPTW	soft	local		self-generated
Tang and Hu, 2005	X (50%)		uncap.	VRPTW	hard	local		Solomon, 1987
Potvin et al., 2006	X (50%)	travel time	uncap.	VRPTW	soft	local		Solomon, 1987
Chen et al., 2006	X (78%)	travel time	cap.	VRPTW	hard	local		Solomon, 1987 + real-life
Branchini et al., 2009	X (60%)		cap.	VRPTW	soft	local	X	self-generated

Table 3.2: Local search approaches

3.3.1.1 Local Search Approaches

The local search approaches that are subsequently presented consist mainly of two parts. The first part is applied to construct a feasible solution (e.g. with Best Insertion). The second part uses classical techniques (like Re-Insertion or 2-opt) to improve this initial solution. Measures to escape from local optima are not applied.

Table 3.2 summarizes the properties of five selected papers. All of them consider the local area VRPTW with dynamically occuring requests. The degree of dynamism varies between 50% and 100%. In addition, dynamic travel times are included by two publications as a second source of dynamism. The latest publication by Branchini et al. (2009) allows for en route diversion.

Shieh and May (1998) consider a capacitated VRP with hard time windows, where up to 50% of customers occur dynamically. The objective is the minimization of traveled distance. Orders may be rejected. The authors propose a heuristic that uses *Best Insertion* for constructing a feasible solution, *followed by intra- and interroute improvement with OR-opt and 2-opt.* The improvement part is run continuously between the occurrence of two requests.

For testing purposes, the static VRPTW instances of Solomon (1987) are extended by a new column with random Call-In times. Analyses are carried out, comparing the results of the dynamic approach with the best known solutions of the static Solomon instances. The authors run their *simulations in real time* and report an increase in the number of used vehicles by factors of 1.12 to 2.14 and an increase in traveled distance ranging from 7% to 25%. It has to be mentioned that for some problem sets, not all requests could be serviced due to possible late Call-In and hard time window constraints.

Du et al. (2005) regard a capacitated VRP with soft time windows, in which up to 100% of customers occur dynamically. The objective function consists of two levels. The first level goal is to minimize the total distance traveled. When no feasible insertion position can be found, the goal is to minimize delay. The authors propose a heuristic with construction and improvement parts. In total, four construction methods are presented, partially depending on geographical order clustering, similar to the *"sweep algorithm"*:

- find the cluster to which the request belongs to and append the new order at the end of the associated vehicle's queue;
- assign the order to the vehicle with the smallest distance between the last order in the vehicle's queue and the new order's location;
- apply *Best Insertion* with regard to all vehicles;
- find the cluster the order belongs to and apply *Best Insertion* to the associated vehicle queue. If necessary (due to capacity constraints), take a new vehicle from depot; if necessary (due to absence of additional vehicles at the depot), check insertion cost for vehicles in close regions.

The improvement component consists of interroute changes with *Best Re-Insertion and 2-Exchange* (each of two routes is cut into two segments, then the second segments are exchanged) and intraroute changes with *Or-opt and 2-Swap* (exchange of two nodes within a vehicles route). Improvement is executed as pure local descent, i.e. changes are only accepted when an improvement of the objective function is found. Tests are conducted with self-generated data sets, showing best results for construction with Best Insertion followed by improvement with Best Re-Insertion and OR-opt.

Tang and Hu (2005) deal with an uncapacitated VRP with hard time windows, in which up to 50% of the customers occur dynamically. The main goal is to maximize the number of serviced customers. This is achieved by accepting as many customers as feasible. Further goals are the minimization of customer waiting time (defined as the time gap between Call-In and start of service) and the minimization of traveled distance, with higher priority being attributed to the reduction of customer waiting time.

The authors propose a rolling horizon based approach, which is triggered by the occurrence of new orders. For the acceptance decision, not only *Best Insertion* is used: When the first attempt does not result in a feasible plan, additional adjustments with *Best Re-Insertion and OR-opt* are applied in order to create a feasible insertion position for the new order. The order is only rejected if all these attempts fail. Subsequent improvement is carried out with a version of OR-opt, examining the relocation of three, two, or one consecutive nodes in the vehicle's current tour. Occasionally, a number of requests is extracted and re-inserted.

The Solomon (1987) instances are used as test data, with the extension of dynamic Call-In times and modified time window characteristics. For test instances with wide as well as narrow time windows, the authors report "dramatic benefits" with the new approach when compared to a benchmark procedure, based on Best Insertion and OR-opt (objective: minimization of travel time). General results show high quality solutions within a limited computing time.

Potvin et al. (2006) deal with an uncapacitated VRP with soft time windows, in which 50% of the customers occur dynamically. A special focus is placed on *travel times which are subject to several fluctuations*:

- Depending on the time period of the day, the average travel time is multiplied by prespecified coefficients ("long-term forecast"). This is an a-priori known information.
- The moment a vehicle starts traveling on a link, a short-term bias to the travel time coefficient is revealed ("short-term forecast"). This value is chosen according to a uniform random distribution in the interval $[-0.1, +0.1]$.
- The arrival time at a link's destination is furthermore distorted by unforeseen events that may occur along the travel leg. These are modeled as normally distributed perturbations with a mean of 0 and standard deviations ranging from 1 to 32. Here, only delays to the current schedule are considered, thus a negative value is simply reset to 0. Information about the extent of such variations is first known to the algorithm when the vehicle finally arrives at the destination.

The general objective is to minimize an equally weighted sum of travel time, lateness and overtime. The authors propose a solution approach based on *Best Insertion and subsequent improvement with Cross-Exchange* ("two segments of routes are exchanged between two different routes by removing two arcs in each route and by appropriately reconnecting the two segments.") and *Intraroute Exchange*. The procedure is applied at the beginning and at the occurrence of the following events:

- the arrival of a new order,
- when the short-term forecast on travel time is introduced at vehicle departure, and

- when the arrival time at a location is delayed by a "tolerance time limit".

In the last case, the order is reassigned to another vehicle. If the original vehicle arrives before the new vehicle, the algorithm tries to cancel the re-assignment. If the new vehicle has not yet started traveling to the observed location, the associated node is simply removed from its schedule, otherwise the new vehicle reaches the location without serving it (no en route diversion).

As test data, the authors use dynamic extensions to the Solomon (1987) data sets. They observe that an increasing magnitude of dynamic leg travel time perturbations results in harder to solve problems. After considering different levels for the "tolerance time limit", a short waiting time shows the best performance. Events of small magnitude are caught, and reaction is only performed on events of larger magnitude.

Chen et al. (2006) investigate a capacitated VRP with hard time windows, in which up to 77% of orders occur dynamically. *Travel times are also subject to dynamic fluctuations*, modeled as distortions to (a-priori known) time-dependent travel times. The objective is the minimization of a weighted cost function, containing travel and waiting time. New orders may be rejected if a feasible insertion position into the current plan cannot be found. Rejection is also possible if fluctuations in travel time make it impossible to serve an order within its time windows.

The authors propose a heuristic approach which uses *Best Insertion* for route construction, *followed by an improvement routine with OR-opt.* Planning runs are triggered by the occurrence of new dynamic information and by execution of irreversible planning events (e.g. permanent order-to-vehicle assignment or when the vehicle starts traveling to a specific order).

Extended Solomon (1987) data and some real-life data from a logistics company located in Taiwan are used for testing purposes. Dynamic travel time fluctuations are modeled with two types of random variables. First, the interval lengths of time-dependent travel times are distorted, then the corresponding travel times. For reasons of comparison, the authors apply a solution approach that is not able to include dynamic updates in travel times. As expected, the solution approach that considers those changes in travel time significantly surpasses the benchmark procedure in performance. For the second data set, the new algorithm is benchmarked with the manual planning of human dispatchers: The results for a fleet of six vehicles show a reduction in total travel time from originally 875 minutes down to 832 minutes.

Branchini et al. (2009) consider a capacitated VRP with soft time windows, in which *up to 60% of customers occur dynamically.* The objective is the minimization of traveled distance. The authors propose a construction heuristic that uses the initial static requests to distribute the available vehicles equally across the whole service region, in order to accommodate future dynamic customers more easily.

After a request arrival has taken place, the new customers are included with *Best Insertion.* In addition, an improvement procedure is continuously run, investigating *2-opt, OR-opt and Cross-Exchange* neighborhoods. Depending on arrival intensity, these neighborhoods are dynamically reduced ("adaptive") in order to concentrate on high quality solutions. Hence, "long" arcs with small probabilities for improvement are neglected. The authors call their approach *granular search.* The approach further includes a basic "wait first" strategy, en route diversion, as well as vehicle re-positioning to strategic waiting places.

For testing purposes, three instances are generated with a personal data generator. The basic parameter settings are taken from real-life transportation companies. The authors apply several nine-hour *real time simulations* and compare their new approach with Best Insertion and Nearest Neighbor. As expected, significant profit gains were achieved when employing the new sophisticated approach. In addition, a 4% better objective value is reported when all information is made known to the algorithm a-priori.

3.3.1.2 Metaheuristics

The second group of dynamic approaches, the "Metaheuristics", can be seen as a *control level* above one (or several) basic local search procedures (improvement neighborhoods). Metaheuristics usually allow for temporary deterioration of the objective function value in order to escape from local optima. Five representatives are subsequently considered: *Tabu Search, Evolutionary Approaches, Variable Neighborhood Search, Ant Colony, and the concept of a Second Objective Function.* Table 3.3 summarizes the properties of the associated publications.

Tabu Search

In a Tabu Search (cp. Glover, 1989), the algorithm moves towards the best available solution, generated by its underlying neighborhood. This is possible as long as a solution is not stored within the "tabu list", which includes (usually for a given time horizon) already visited solutions. Since the algorithm is not allowed to choose such a tabu list solution, it is forced to explore regions of the search space that would otherwise be left unexplored. Temporary worsening of the objective function value is explicitly allowed in order to escape from local optima.

Five Tabu Search publications, all considering a local area geographical extension, are presented: two focusing on the uncapacitated VRPTW, two focusing on the Dial-A-Ride problem and one dealing with the uncapacitated MLPDPTW. An interesting aspect of the second paper by Ichoua et al. (2000) is the detailed investigation of the impact of "en route diversion".

Gendreau et al. (1999) deal with an uncapacitated VRP with soft time windows for a Courier service application, in which 50% of the customers occur dynamically. New orders may be rejected if they cannot be handled within a feasible solution, e.g. because of hard time window restrictions at the depot. The objective is to minimize a weighted cost function, including total distance traveled and time window violations.

The authors introduce a *parallel Tabu Search algorithm with Adaptive Memory*, based on Taillard et al. (1997). The Adaptive Memory, similar to the concept of Genetic Algorithms, contains a set of feasible solutions. These are generated in the beginning by

	dynamic aspects: orders	dynamic aspects: other	capacitated	problem	TW constr.	area	en route diversion	dynamic test data
Tabu Search								
Gendreau et al., 1999	X (50%)		uncap.	VRPTW	soft	local		Solomon, 1987
Ichoua et al., 2000	X (75%)		uncap.	VRPTW	soft	local	X	Solomon, 1987
Attanasio et al., 2004	X (50%)		cap.	DARP	hard	local		Cordeau and Laporte, 2003
Fabri and Recht, 2006	X (100%)		cap.	DARP	hard	local		Caramia et al., 2002
Gendreau et al., 2006	X (100%)		uncap.	PDPTW	soft	local		self-generated
Evolutionary Approaches								
Haghani and Jung, 2005	X (55%)	travel time	cap.	VRPTW	soft	local		self-generated
Pankratz, 2005	X (100%)		cap.	MLPDPTW	hard	local		Solomon, 1987
Hanshar and Ombuki-B., 2007	X		cap.	VRP		local		Kilby et al., 1998
Cheung et al., 2008	X (16%)	travel time	cap.	MLPDPTW	hard	local		self-generated
Okhrin and Richter, 2008		travel time	cap.	VRPTW	hard	local		Solomon, 1987
Variable Neighborhood Search								
Angellelli et al., 2004	X		uncap.	VRPTW	hard	local	X	
Bock, 2010	X	several	cap.	MLPDPTW	soft	wide	X	self-generated
Ant Colony								
Montemanni et al., 2005	X		cap.	VRP		local		Kilby et al., 1998; real-life
Guntsch and Middendorf, 2002	X			TSP				TSPLIB
Second Objective Function								
Xiang et al., 2008	X	several	cap.	DARP	soft	local		self-generated

Table 3.3: Metaheuristics

a stochastic insertion heuristic and subsequent improvement with Tabu Search (neighborhood: Cross Exchange). Based on the solutions found in the Adaptive Memory, new solutions are repeatedly composed and improved with Tabu Search (cp. Figure 3.4).

Afterwards, the best new solutions are added to the Adaptive Memory. In a dynamic environment, the improvement procedure is run until a new event occurs. If the new event is the occurrence of a new request, the latter is inserted into each Adaptive Memory solution. If the event represents the end of service at a customer location, the driver's next destination is identified using the best solution stored in the Adaptive Memory. In order to keep solutions consistent, the other solutions are updated accordingly. Then, the overall improvement process is restarted.

The Solomon (1987) data sets, extended to dynamic aspects, are used for testing purposes. The new approach is benchmarked with several "easier" heuristics: simple successive in-

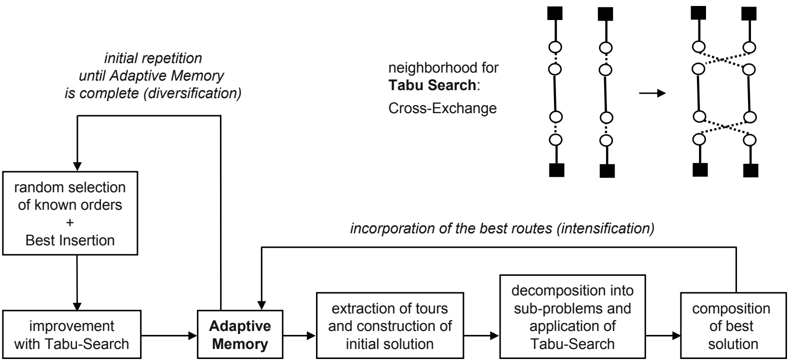

Figure 3.4: Tabu Search algorithm with Adaptive Memory

sertion, successive insertion and improvement with cross exchange, complete solution rebuild with insertion, rebuild and improvement with cross exchange, and parallel Tabu Search with stop at the first local optimum. As expected, results show best performance when the complete approach is run. In addition, the authors show that it is beneficial to optimize the planned routes between the occurrence of two events. With increasing computation time better average results are achieved. Finally, the *parallelization of the algorithm on up to 16 processors* is investigated, resulting in more customers being serviced and in a reduced sum of distance and lateness.

Ichoua et al. (2000) extend the scope of action for the problem and algorithm considered in Gendreau et al. (1999) by allowing **"en route diversion"**. *A driving vehicle may be directed away from its current destination in order to serve a request that has just occurred in the vicinity of its current position.*

The authors introduce a framework with variable anticipation horizon δt. Every time t the solution approach is restarted, decisions in the horizon $t + \delta t$ are "frozen", so that the current algorithm run may schedule planning updates soonest for time $t + \delta t$. During the frozen period, new requests may arrive, which have to be accepted or rejected (only if no feasible insertion position is available). For this purpose, a copy of the current plan is held ready, which is updated if a new best solution is found. In this situation, the authors do not consider the case of two similar orders arriving in the same frozen interval. Both could be accepted separately, but together they could render the plan infeasible.

In order to determine reasonable computation times, the authors suggest three variable rules for calculating the value of δt:

- δt is chosen in such a way that the solution procedure ends before any vehicle arrives at its current destination,
- δt is chosen to be proportional to a moving average of the last l inter-arrival times,
- δt is chosen to be the length of some time horizon X divided by the number of requests on the planned routes found within that time horizon.

In first tests, the third rule succeeded in reducing both the number of unserved customers, as well as total objective value.

The same test data sets as in Gendreau et al. (1999) are used, providing insights into potential gains by applying "en route diversion". Compared to the original heuristic, the number of unserved customers is indeed reduced by 16.8% up to 100%. In addition, the objective function value is decreased by 2% up to 4.3%, indicating substantial benefits through the exploitation of en route diversion.

Attanasio et al. (2004) deal with a capacitated Dial-A-Ride problem, where 50% of the requests occur dynamically. Orders possess one hard time window, which is either the Delivery time window (for outbound trips) or the Pickup time window (for inbound trips). In addition, a maximum ride time of 90 minutes has to be considered. New orders are accepted if they can be feasibly inserted into the current plan without violating any hard constraints. The objective is the minimization of traveled distance.

The authors develop a *Tabu Search approach*, which is a dynamic and parallelized extension to Cordeau and Laporte (2003). An initial solution is generated by relaxing several hard problem restrictions (capacity, maximal route duration, time windows, and user ride time constraints). Then, the Tabu Search, based on a *Best Re-Insertion neighborhood*, explores the solution space including infeasible solutions. After each iteration, the objective function cost parameters are adjusted, raising and decreasing the weight for restrictions that have been violated and complied with, respectively. With the help of a tabu list, solutions are penalized by a factor proportional to the frequency of the addition of its distinguishing attributes. If a new best solution is found, intraroute exchanges are performed.

As test data, the authors use 26 dynamic instances from real-life applications in Montreal/Canada and from a Danish company. A parallelization strategy is applied with an increasing number of processors ranging from one to eight. Results show the benefits of enhanced computing power via parallel computing. A performance statement on how well the proposed algorithm works in dynamic environments (e.g. by comparison with static data or with other dynamic approaches) is not explicitly mentioned.

Fabri and Recht (2006)[6] investigate a capacitated Dial-A-Ride problem with 100% dynamically occurring customers. When a request arrives, it is accepted if a feasible insertion position into the current plan (in compliance with hard time windows for Pickup and Delivery) is available. The objectives are the minimization of rejected orders and the minimization of traveled distance.

The authors employ a dynamic solution approach that extends the work of Caramia et al. (2002) by introducing explicit Delivery time windows and by allowing waiting times. When a new order occurs, all vehicles are successively inspected for a feasible insertion position. If such a position exists, the order is assigned to the cheapest vehicle.

The single vehicle subproblem is solved as follows. A network of possible status vectors is established, in which each order may have the status already delivered ("0"), already

[6] Based on the author's dissertation: Fabri (2008)

picked up ("1"), or waiting for Pickup ("2"). Two vertices are connected by an edge whenever the subsequent vector can be obtained from its predecessor by subtracting 1 to exactly one vector element. The source vertex is the vector $(2, 2, \ldots, 2)$, i.e. all accepted demands are waiting for Pickup, the sink vertex is the vector $(0, 0, \ldots, 0)$, i.e. all accepted demands have been delivered. The problem is to find a shortest path from the source to the sink, subject to time windows and capacity constraints. This task is performed with an *A* algorithm* that reduces computation time by using a lower bound approximization function to estimate the cost of the route from the current vertex to the sink. Between two events, the solution is improved with a Tabu Search algorithm based on the *neighborhoods Best Re-Insertion and* $(1, 1)$*-Exchange.*

As test data, the authors use adapted data from Caramia et al. (2002). The "stretch factor", which originally implies the maximum acceptable ratio between actual and minimum expected travel time, is converted into a Delivery time window. Tests with 20 vehicles show significant improvements, ranging from 3.83% up to 10.74% in comparison to results of the original algorithm. It is observed that the new approach produces better solutions for problems with a small number of vehicles and a high number of orders.

In a further work, **Gendreau et al. (2006)** investigate an uncapacitated MLPDP with soft time windows, in which up to 100% of the orders occur dynamically. The objective is the minimization of an equally weighted cost function consisting of travel time, lateness and overtime. The authors rely on the same optimization framework proposed in Gendreau et al. (1999). The *parallel Tabu Search with Adaptive Memory* is primarily changed in terms of the basic neighborhood. Instead of Cross Exchange, an *Ejection Chain* procedure is applied. *A request (Pickup and Delivery) is taken from one route and moved to another route, thus forcing a request from that route to move to yet another route, and so on.* The chain may be of any length and may be cyclic or not.

For testing purposes, three main scenarios, each including five instances with increasing requirements to the solution procedure ("temporal utilization" of vehicles 28%, 57% and 78%, respectively) are generated. See Section 4.4.2 for a detailed presentation of these dynamic test sets. A comparison of the new approach is carried out with the adapted benchmark heuristics from Gendreau et al. (1999). It is worth mentioning that *simulation is run in real time*, producing realistic time intervals between events for running the improvement.

Best results are achieved with the application of the new approach. The results of "complete solution rebuild" show worst performance because previously obtained solution structures get lost. With increasing stress, the solution quality of Tabu Search with Adaptive Memory and benchmark heuristics gets progressively closer. The authors explain this finding by a lack of computation time between two consecutive events. When parallelization with 16 processors is applied, additional improvement ranges from 2.2% to 5.7%.

Evolutionary Approaches

This group of metaheuristics is inspired by *biological evolution.* Candidate solutions to the optimization problem play the role of individuals in a population, and each individual's fitness is determined by its associated objective function value. The procedures initially generate a diversified pool of solutions (parent generation), which is used as origin for improvements (child generations). The improvements are achieved by using mechanisms like "selection", "recombination", or "mutation". The idea of "Genetic Algorithms" goes back to Holland (1975).

Again, all five selected papers deal with a local geographical area. Interestingly, three of five publications explicitly consider dynamic travel times: two in combination with dynamic requests, and one focusing only on dynamic travel times.

Haghani and Jung (2005) consider a capacitated VRPTW with 55% dynamic requests and dynamic travel times. The dynamism of travel time is modeled by varying link travel speed, which is calculated as the link's average speed multiplied by a dynamic factor, depending on the time of day. Information about time-dependent variations in travel time is not ex-ante known to the planning algorithm. The objective is the minimization of a weighted cost function with costs for used vehicles, traveled distance and violation of time windows.

The authors present a rolling horizon based *Genetic Algorithm* with the following *encoding*. A feasible solution for the VRPTW consists of a sequence of four-digit numbers. Each number belongs to a real order, in which the first digit indicates the assigned vehicle, and the last three digits are used as sorting keys for each vehicle's routing. At the beginning, an initial parent population is randomly generated and evaluated for fitness. Afterwards, new individuals (children) are generated by applying *two-point crossover, mutation and vehicle merging*. Individuals of a new generation are selected from both the parent and children generation. The best solution is always passed onto the next generation (*elitist strategy*), while the remaining solutions are chosen with probabilities depending on their *fitness*.

Solution quality is evaluated by a comparison with exact CPLEX solutions (for very small problems with less than 10 demand nodes) and lower bounds (based on a relaxed MIP formulation). Results show gaps of less than 5% for 5 to 25 demand node problems. For a 30 demand node problem with 30 time periods, the gaps increase up to 7.9%. Overall, the authors report excellent results within very short computation times. In addition, the new approach is applied to a larger case study's data, which shows the benefits of reacting to dynamic changes in travel time by comparing the performance of a plan revision strategy with a non-revision strategy.

Pankratz (2005)[7] investigates a capacitated PDP with hard time windows and up to 100% dynamically occurring customers. The objective is the minimization of total traveled distance. All orders have to be served and, if necessary, an additional vehicle is introduced. As solution approach, the author proposes a *"Grouping" Genetic Algorithm*.

Pankratz argues that in an MLPDPTW, the assignment problem of orders to vehicles has more influence on solution quality than the routing problem. This is motivated by the fact

[7] Based on the author's dissertation: Pankratz (2002)

that time windows and precedence constraints considerably restrict the number of routing alternatives for a given allocation of requests. In the presented approach, an MLPDPTW solution ("chromosome") is therefore encoded by clusters of requests assigned to a single vehicle ("genes"). Additionally, a chromosome contains routing information for each gene, which is hidden from the Genetic Algorithm and cannot be directly manipulated by the genetic operators. The encoding is visualized in Figure 3.5.

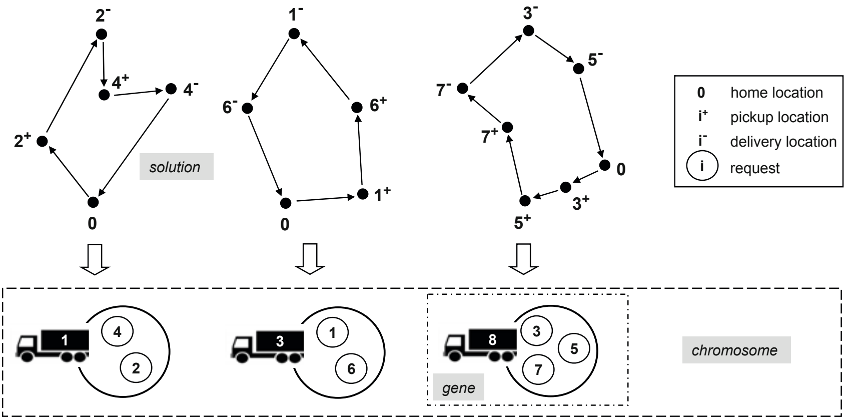

Figure 3.5: Encoding of the "grouping" Genetic Algorithm (cp. Pankratz, 2005)

An initial population is generated by repeated *Best Insertion* of all requests in random order until the desired population size is reached. The algorithm successively selects pairs of individuals with regard to their fitness and generates two children by applying *crossover and mutation operators*. For Crossover, two crossing sections are specified in each parent. Then each parent's section is inserted into the other parent. To yield feasible solutions, some repair actions, e.g. elimination of vehicles or elimination of orders occurring twice, are performed. A subsequent mutation randomly eliminates a gene and re-inserts the associated requests.

If a new request occurs, all irreversible decisions of the "plan in execution" up to this point in time are discovered, and a synchronization of all individuals of the population is carried out. Then, the new request is inserted into all individuals. Subsequently, the Genetic Algorithm is restarted, and, after termination, the best solution is picked as new "plan in execution". Dynamic test data sets are derived from the static PDPTW instances of Li and Lim (2003) (cp. Section 3.4 for details). Two insertion heuristics without improvement are used for comparison: (i) incremental insertion and (ii) total plan revision ("from scratch") every time a new request occurs. Both methods are significantly outperformed by the GA, which produces up to 5% reductions in traveled distance. However, with an increasing degree of dynamism, the gap between GA and insertion heuristics shrinks.

Hanshar and Ombuki-Berman (2007) report on a capacitated VRP with dynamically occurring customer requests and the objective of minimizing traveled distance. A rolling horizon based *Genetic Algorithm* is presented. Similar to Montemanni et al. (2005), dynamic information is batched up to the end of equal time slices and processed in the following time slice. During time slices, "optimization" is run based on known data. At the end of each time slice, the best known solution is chosen for execution. Decisions that have a processing time starting within the next time slot are permanently fixed.

The *encoding* of a VRP solution is performed by a series of positive and negative integer numbers, describing the sequence of orders. While positive numbers indicate open (not fixed) orders, negative numbers are used for identification of order bundles that have already been assigned to a specific vehicle. The series of numbers is traversed from the left to the right, successively assigning orders to vehicles. When a vehicle's capacity is reached, or when a negative number occurs, a new vehicle is introduced.

An initial population of 400 individuals is randomly generated and evaluated. Afterwards, individuals are chosen for *Crossover* according to their fitness. For crossover, a route from each parent solution is randomly selected, and the customer orders present in each route are removed from the *other* parent. Then, the customers are reinserted with *Best Insertion*. This is repeated until a sufficient number of feasible solutions for the next generation is available. Some of the new solutions are subject to the mutation operator that reverses the sequence of orders between two randomly chosen cutting points. Finally, 1% of the worst new solutions is replaced by the 1% best solutions from the parent generation.

Test data and benchmarking results are taken from Montemanni et al. (2005), who have developed an Ant Colony based approach. In addition, the authors develop a Tabu Search approach with the neighborhoods inversion and λ - interchanges (1,0) to (3,3). Nevertheless, best results are obtained with the new Genetic Algorithm, followed by Tabu Search and Ant Colony. The GA outperforms Montemanni's Ant Colony results by 5.26% on average.

Cheung et al. (2008) deal with an MLPDP with hard time windows, in which travel times and the occurrence of new requests (up to 16%) are subject to dynamism. Dynamic orders may be rejected only if there is no feasible insertion position. The goal is the minimization of total travel time.

The authors propose a *genetic solution approach*, which is triggered by the arrival of new information (new orders, changes in travel time). The *encoding* of a solution is performed as follows. For each order, a triple of numbers ("a gene") is stored, where the third number denotes the assigned vehicle, and where the first and second numbers denote the routing positions of Pickup and Delivery in the vehicle's tour. An initial population is generated by first building pendular tours (depot - Pickup - Delivery - depot), with orders lying at a prespecified distance from the depot (subject to variation), and by subsequently inserting the remaining orders into the pendular tours.

When a sufficient population size is reached, solution pairs are selected for *Crossover* (according to fitness). In the Crossover operation, a random number of genes from parents A and B is exchanged. If the *fitness value* of one of the new emerging solutions outperforms both parents, the parents are discarded and replaced by their offspring. Otherwise, both parent solutions are kept in the population. Afterwards, each new generated solution is subject to *mutation*. For a random number of requests, the assigned vehicle is changed. The mutated solution is only accepted if it has a better fitness value than before.

The authors use some self-generated test instances with customer locations evenly distributed in a unit square. While information about the number of dynamic customers (four in all test instances) is available, frequency and magnitude of travel time variations remain undefined. The impact of dynamic data on solution quality is investigated by applying the new algorithm on the associated static case, with resulting gaps between 5% and 10%. According to the authors, the effectiveness of the dynamic re-optimization is quite high.

Okhrin and Richter (2008) consider a capacitated VRPTW with time-dependent and dynamic travel times. For each pair of nodes, four travel time values are specified, associated with different day time intervals. In addition to this a-priori known information, travel time is subject to dynamic fluctuations, which are modeled as normally distributed deviations $N(0, 4)$ and $N(0, 9)$. The objective is the minimization of total travel time.

As solution approach, the authors propose a *Genetic Algorithm*. The initial population is created by first sorting customers according to the urgency of their time windows and the subsequent application of *Best Insertion*. A new generation of the same size is generated with a selection procedure, followed by *crossover and mutation*. For selection purposes, a random number of individuals is repeatedly chosen from the whole population. In every round, one individual from the chosen subset reaches the next generation (with a probability of 80% for the fittest one, otherwise a randomly selected individual of the remaining subset).

Afterwards, 90% of the selected individuals are subject to the Crossover operator. Partial routes are randomly chosen from two individuals. Then the associated orders are removed from the other respective individual, followed by Best Re-Insertion. The mutation operator is applied to 10% of the offspring. A random customer within each individual is exchanged with the customer that has the most similar time window. Finally, the 1% best solutions from the old population are transferred to the new one, replacing the worst individuals (*"elitist strategy"*).

The authors test their approach with modified Solomon (1987) data and prove the efficiency of the Genetic Algorithm for static planning situations. In addition, the benefits of dynamic reactions to fluctuations in travel time are shown by comparison with a "no reaction strategy".

Variable Neighborhood Search

The concept of Variable Neighborhood Search was proposed by Mladenovic and Hansen (1997). The basic idea of VNS is to search for improvements from the current best solution, first using the *smallest neighborhood* in order to randomly (!) generate one new solution. This solution serves as starting point for another local search procedure, which is executed until a local optimum is found. If the local optimum is a new best solution, the search is re-started from this new solution. Otherwise, the *"radius" of the neighborhood around the original best solution is successively increased.*

Two publications using VNS in dynamic environments are selected. The first one focuses on generating a concept for how to realize VNS in a dynamic situation. The second publication also reports on computational tests. A main feature of the second approach is the explicit consideration of “transshipment options”.

Angelelli et al. (2004) investigate an uncapacitated VRP with hard time windows, in which new requests occur dynamically. Orders are classified into priority levels according to their time window’s urgency. The objective is to maximize the total priority value of the served requests. Generally, orders may not be rejected, but some orders with time windows that are more distant may be postponed to the next shift.

The authors present a concept for a rolling horizon based solution procedure that is applied in fixed specified time intervals. A plan is made feasible by inserting all unpostponable orders with the help of Best Insertion, re-arrangement of orders, and extraction of postponable orders. Subsequently, a *Variable Neighborhood Search (VNS) improvement procedure* is applied. A neighborhood is defined by the number k of postponable requests, which are extracted from the current solution. These orders are labeled “tabu” and stored in a pool of not assigned postponable orders. As many “non tabu” postponable orders as possible are then attempted to be inserted into the current plan. Computational results are not reported.

An algorithm much like “Variable Neighborhood Search” is also applied by **Bock (2010)**. The author focuses on a dynamic MLPDPTW, including several sources of dynamism: requests, vehicle breakdown, vehicle deceleration, route blockage, and traffic congestion. The main contribution, however, is the integration of **multi-modal transport chains and multiple transshipments**. The author models four *shipment scenarios* for transporting a load from a Pickup to a Delivery location (an exemplary visualization can be found in Figure 3.6):

- direct transportation,
- transportation making use of one transshipment point (depot or hub),
- transportation making use of two transshipment points (two depots, or one depot plus one hub), and
- transportation using several transshipment points (several depots, plus one hub if necessary).

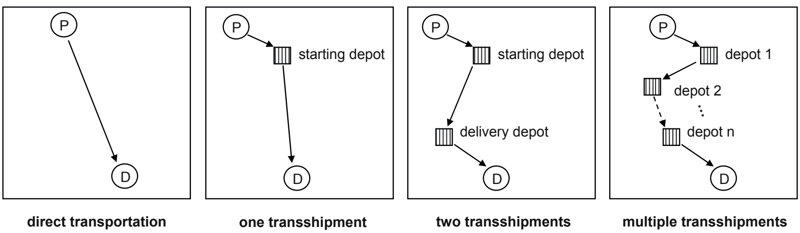

Figure 3.6: Exemplary illustration of available shipment scenarios (cp. Bock, 2010)

The framework also allows for a kind of *en route diversion*. A directly scheduled transportation task, which is already loaded onto a vehicle, can be dynamically exchanged at the next transshipment point, thus enabling the assignment of another task to this vehicle. Construction and improvement procedures are based on *Best Insertion*. In the improvement part, a specific number of requests is extracted and re-inserted, investigating all possible transshipment scenarios. If no improvement can be found, the number of extracted requests is successively increased (VNS). As soon as an improvement has been found, it is accepted and the procedure starts with the first step (=just extracting one request).

For testing purposes, the author generates data sets whose structure is "derived from practical applications". He compares a *time-based* (continuous improvement with fixed anticipation horizon) and an *event-based* simulation technique. Better performance is achieved with the time-based approach. In addition, a rule-based benchmark procedure is applied to the test data. As expected, this procedure is clearly outperformed by the VNS-like procedure. Finally, it is proven that the availability of complex transshipment structures results in more transportation options and therefore yields better overall results.

Ant Colony

The concept of Ant Colony optimization was proposed by Dorigo (1992), who was inspired by the behavior of ants seeking a path between their colony and a source of food. Each ant lays down a *pheromone trail* on the paths it travels. If other ants find such a path, they are likely to follow the existing trail, thus *reinforcing* it. The concept is transferred to Vehicle Routing in order to find "optimal" paths.

For this metaheuristic, two dynamic publications have been chosen that consider dynamically occurring customers for a VRP (Montemanni et al., 2005), and dynamic changes in customers locations for a TSP (Guntsch and Middendorf, 2002).

Montemanni et al. (2005) propose an *Ant Colony approach* for solving a capacitated VRP with dynamically occurring requests. The objective is the minimization of total travel time. In order to handle the dynamic requests, the working day is divided into time slices of equal length, wherein new orders are batched. These new orders are incorporated during the planning run of the subsequent time slice. Over the period of each time slice, the ant colony heuristic is run and the best solution that is found is realized at the beginning of the next time slice.

The ant colony heuristic works as follows. Every ant produces a feasible VRP solution by choosing customers successively according to given arc probabilities. The probability of visiting customer j after customer i depends on two factors: the general attractiveness of the arc (depending on travel time) and the *pheromone level* (indicating how proficient it has been in the past to visit j after i). When an arc has been chosen within one generation of ants, its pheromone level may be locally reduced to favor exploration. Once all ants of the colony have completed their computation, the best known solution is used to globally modify the pheromone trail. In this way, a "preferred route" is memorized in the pheromone trail matrix and future ants will use this information to generate new solutions in a neighborhood of this preferred route. To reduce the impact of "older" solutions, some pheromone information is *evaporated* at the beginning of each time slice.

On the basis of dynamic test data from Kilby et al. (1998), the authors compare their approach with an easy heuristic (Nearest Neighbor construction and Best Re-Insertion improvement). Better performance is achieved by the new Ant Colony approach with an average decrease in total travel time of 3.2%, compared to the easy benchmark heuristic.

Guntsch and Middendorf (2002) deal with a TSP with dynamically changing cities. While the total number of cities is kept constant, a fixed number of random cities is exchanged with other cities (from a pool of cities) every t time units. The objective is the minimization of total traveled distance. The authors propose a *population based Ant Colony approach*, which connects Ant Colony with aspects of a *Genetic Algorithm*. Instead of transferring pheromone information, a set of solutions is transferred from one iteration of the algorithm to the next. This set of solutions is then used to compute the pheromone information for the ants of the next iteration. A specified number of ants generates TSP solutions, in which each routing decision depends on the probability of the optional links. With a probability of 0.9, the arc with the highest probability is chosen. With a probability of 0.1, one of the other arcs is chosen according to their individual probability.

In order to update the pheromones, the authors investigate several strategies for replacing an old solution by the best new generated solution. It turns out that the best strategy is either to simply replace the oldest solution or to randomly choose a solution for exchange (with higher probability for an inferior one). As a consequence of dynamic changes in cities, the solutions in the population are altered infeasible. To overcome this problem, the authors discuss a *Complete Restart* or a *Repair by Best Insertion*. Repair performs better when only minor changes in data have to be included, while Restart is preferable in situations with higher dynamism.

Second Objective Function

Finally, a last metaheuristic concept is considered (cp. Helay and Moll, 1995), which differs slightly from the others. In order to escape from local optima, a *Second Objective Function* is introduced with the goal of temporarily deteriorating the *primary objective function*. This secondary objective function should be rather different from the main one in order to drive search far enough, but should also be partially dependent on the primary one, in order to avoid worsening its value too strongly.

Xiang et al. (2008) consider a dynamic Dial-A-Ride problem with soft time windows. Nearly *all possible sources of dynamism are regarded:* arrival of new requests, fluctuations in travel speed and service time, no-shows of customers and cancelation of requests, traffic jams and vehicle breakdown. An algorithmic solution concept based on *Best Insertion*, supplemented by improvement with basic versions of *intra- and intertour exchanges*, is proposed.

The *primary objective function* minimizes a weighted cost function, including vehicle fixed costs, distance, travel time, waiting, service time, violation of maximum travel time, overtime and delay. The *secondary objective function* is chosen in a similar way, focusing specifically on the costs of empty distance, empty travel time and empty waiting time. For testing purposes, a data set is self-generated. Interestingly, not all possible sources of dynamism are tested at once. Instead, the impact of each source is investigated separately with the following findings: while long term traffic jam and vehicle breakdown cause severe modifications in the schedule, cancelation of customers, travel time fluctuations and service time variances induce only minor changes.

After the treatment of different metaheuristics, we will continue with the next "main algorithmic solution concept", which is based on exact procedures.

3.3.1.3 Heuristic Application of Exact Procedures

The third group of dynamic myopic approaches is based on exact procedures, which are applied to the static subproblems of a dynamic instance (including all available information up to a specific point in time). Since the dimension of those static subproblems is usually quite big and due to limited computation time to solve a static subproblem, the regular application of exact solution procedures is less suitable. Instead, the exact procedures are applied only to a relaxed subproblem combined with some subsequent repair mechanism. Another option actually applies the original exact procedure, but interrupts it after some time, using the best solution found so far.

Those approaches are called *heuristic application of exact procedures*. Due to the character of dynamic problems, "not finding the exact solution" of a static subproblem is not dramatic. As explained in Section 2.1, a "better" solution of a subproblem may not necessarily result in a better overall solution of the dynamic problem.

In the following, a *Column Generation* based approach and a procedure using *Lagrange relaxation* are reviewed. Afterwards, three publications that try to solve the static subproblems with *CPLEX solver* are presented. Finally, the idea of using a *linear assignment procedure* is explained, which is actually solved exactly for each static subproblem (but with the input information for the assignment matrix including some heuristical calculations). Table 3.4 summarizes the properties of the selected publications.

<u>Column Generation</u>

Chen and Xu (2006) consider a capacitated VRP with hard time windows, where up to 75% of customer requests occur dynamically and have to be completely served (no rejection). To fulfill this task, an infinite number of vehicles is available. The objective is the minimization of total distance traveled.

The planning horizon is equally divided into decision epochs (with a length of one or two minutes, according to the scenario). Solutions are successively fixed on a rolling horizon basis up to a prespecified point in time (anticipation horizon). The authors apply a heuristic solution approach, based on *Column Generation*, where a column corresponds

	dynamic aspects orders	dynamic aspects other	capacitated	problem	TW constr.	area	en route diversion	dynamic test data
Column Generation								
Chen and Xu, 2006	X (75%)		cap.	VRPTW	hard	local	X	Solomon, 1987
Lagrange Relaxation								
Li et al., 2009		vehicle breakdown	cap.	VRPTW	hard	local		Solomon, 1987
Application of CPLEX solver								
Yang et al., 1999	X		cap.	SLPDPTW	hard	wide	X	self-generated
Mahmassani et al., 2000	X		cap.	SLPDPTW	hard	wide	X	self-generated
Kim et al., 2002	X (100%)		cap.	SLPDPTW	soft	wide	X	self-generated
Linear Assignment								
Fleischmann et al., 2004	X (49%)	travel time	cap.	SLPDPTW	soft	local		real-life
Powell et al., 2000a	X (70%)	travel time	cap.	SLPDPTW	soft	wide		real-life
Powell et al., 2002	X		cap.	SLPDPTW	soft	wide		real-life

Table 3.4: Heuristic application of exact procedures

to a single vehicle trip. The algorithm consists of two levels. At the first level, a *heuristic is used to generate new columns and to update old columns* (e.g. insertion of new orders). At the second level, a *set-partitioning-type formulation is exactly solved with CPLEX*. Information about dual values of orders is exchanged between both levels in order to guide the local search heuristics. En route diversion is explicitly permitted.

For testing purposes, dynamic extensions to the VRPTW instances of Solomon (1987) are generated. The authors first compare their new approach with the best known solutions for the static Solomon instances. Results are inferior, by on average 3.97% for data sets R1, C1, RC1 and 0.54% for data sets R2, C2, RC2. Afterwards, the new dynamic solution approach is benchmarked with a Local Search Approach (Best Insertion plus improvement with 2-Exchange and OR-opt, with unlimited time) and an "unlimited time version" of the new dynamic Column Generation approach. The results show a 5% better performance of the new approach, when compared to Local Descent. Interestingly, when the new approach, having unlimited time available is compared to its version with limited time, it produces better results only for 70% of the instances.

Lagrange Relaxation

Li et al. (2009) consider a capacitated VRP with hard time windows and dynamism induced by **vehicle breakdowns**. It is observed that a VRP with Pickup tasks has to be treated different from a VRP with Delivery tasks in the case of vehicle breakdown. In the Pickup case, other vehicles can just change their routes to collect the packages from the broken down vehicle's customers. In the Delivery case, however, other vehicles have to change their routes to first Pickup the packages loaded on the broken down vehicle and then deliver them to the corresponding customers (transshipment is only allowed from the broken down vehicle, not between the other vehicles). For both cases, Pickup or Delivery, the authors additionally model the option of having a backup car ready at the depot. If available, this vehicle may be used to fulfill parts of the broken down car's Pickup or Delivery tasks.

The authors develop a *Lagrange relaxation based heuristic*, which is supplemented by a *Best Insertion* algorithm to ensure feasible results. The approach is compared to a "naive manual strategy" and to a pure "Best Insertion" heuristic. The "naive manual strategy" just cancels the services if no extra truck is available at the depot. If such a backup car is available, it is simply sent to continue the tasks of the broken down car. In the case of Delivery, it has to drive first to the breakdown point to collect the loaded packages.

For testing purposes, the static Solomon (1987) instances are taken. The best-known solutions are used as initial routes of the vehicle re-routing problem. Then, one vehicle breakdown is introduced early in the schedule. Some instances are equipped with a backup vehicle at the depot. Solutions show best results for the Lagrangian heuristic: total costs are reduced by 8.53% compared to the "naive manual strategy" and by 4.46% compared to "Best Insertion". In addition, the authors observe generally more service cancelations for Delivery services than for Pickup services if the same algorithm and settings are used.

Application of CPLEX Solver

Yang et al. (1999) consider an SLPDP with hard time windows, where new requests occur dynamically. The objective is the minimization of a weighted cost function consisting of empty distance traveled, delay (deviation from preferred time within hard time windows) and lost revenue (for rejected orders). A mathematical problem formulation, explicitly allowing en route diversion, is given and several strategies to find good insertion positions of new orders are distinguished:

- the load is simply placed at the end of each truck's current job queue,
- the load is placed at the best position in the queue,
- the load is placed at the best insertion position, considering re-sequencing and re-assignment.

The third strategy is implemented with the help of a branch-and-cut procedure in CPLEX and produces optimal solutions for the static subproblems. However, the computational burden is quite high, so that the number of demands which can be reoptimized at any given time has to be limited. As test data, the authors use some self-generated instances of relatively small size. The best results are achieved by applying the "optimal" strategy on a limited number of ten variable requests.

Mahmassani et al. (2000) consider the same problem, primarily discussing strategies for how to reduce the number of variables for the "optimal" insertion strategy. They suggest disregarding the vehicle's next order and the shift of a "cut-off time" from the present to the future, successively removing requests from the pool of potential re-assignment orders as long as the number of remaining orders after the “cut-off time” equals the predefined maximum number of orders. In addition, the merging of spatially close orders is discussed.

Kim et al. (2002) publish a related paper, which is also based on the mixed integer model of Yang et al. (1999), where time windows of the SLPDPTW are now treated as soft constraints. The authors investigate the task of maximizing revenue in an oversaturated system, where half of the dynamic demands have to be rejected.

Several concepts for the acceptance/rejection decision are discussed:

- In the first concept, simply all requests are accepted, until a maximal holding capacity of 360 orders is reached.
- A second concept limits the number of accepted demands that have not been picked up yet to a prespecified threshold of 270 (75% of holding capacity), in order to leave more room for improvement operations to the existing demands.
- In the third concept, the minimal additional empty mileage to reach a new order's Pickup location is calculated with Best Insertion over all vehicles. If the empty mileage is below 18 miles, the new order is accepted.

Improvement is based on the “optimal” CPLEX approach, specified in Yang et al. (1999). To comply with a maximum computation time of 10 seconds, the problem size is reduced. Initially, the vehicle to which the new demand was assigned is chosen, then the spatial proximity of other vehicles' orders is calculated (including a check for time window feasibility). Finally, a subset of promising vehicles plus some random vehicles is chosen for improvement. Every time optimization is restarted, a snapshot anticipating the planning situation after 10 seconds is generated, which serves as starting point for the improvement procedure.

For tests with self-generated test data, the authors introduce the following cost parameters: revenue per loaded mile (\$1.2), variable cost per mile, including both empty and loaded movements (\$0.57), and daily fixed costs for driver and trucks (\$45 + \$45). The results show that the acceptance strategy based on additional empty mileage produces the highest revenue. The other strategies are inferior by 4.5% (threshold of 270 orders) and 10.6% (maximum holding limit 360 orders). It is indicated that keeping the number of waiting jobs in the queue below the holding capacity (at about 75%) is more beneficial than accepting and holding as many demands as possible. Response time may be improved significantly when the length of job queues is limited.

Linear Assignment Problems

Fleischmann et al. (2004) consider an SLPDP with soft time windows, where up to 49% of the customers occur dynamically. In addition, travel times are subject to a-priori known time-dependent fluctuations (in intervals of one hour) and dynamic disturbances occurring in 5-minute intervals. The objective is the minimization of a weighted cost function including travel time, delay and overtime.

The authors propose a solution approach, which is based on the optimal solution of a *Linear Assignment Problem*, where all vehicles and all open orders are considered simultaneously. The associated assignment matrix additionally includes some dummy vehicles and some dummy orders, which are introduced to enable postponement of orders (when an open order is assigned to a dummy vehicle) and waiting of idle vehicles (when a dummy order is assigned to a real vehicle). Each time a new event (e.g. new order, completion of an order) occurs, the matrix is updated and the assignment problem is re-solved. An order-to-vehicle assignment first becomes effective when the vehicle's preceding order is completed.

For tests, the authors use real-life data from a local area express service and travel time data from a traffic management system in the city of Berlin, Germany. For comparison purposes, (i) simple assignment rules and (ii) a Best Insertion procedure with improvement (OR-opt and vehicle-to-tour re-assignment) are used. Procedures are run on test data with varying degrees of dynamism, ranging from 0% to 49%. For the completely static case, the Best Insertion procedure shows best results, while for test instances with 49% dynamic customers, the new assignment procedure outperforms all the other approaches. The authors determine an increasing advantage of the new assignment procedure for increasing levels of dynamism and attribute these findings to the preservation of high flexibility that is achieved by fixing the order-to-vehicle assignments at the latest possible time.

Further publications considering the SLPDPTW with solution approaches based on the linear assignment problem are published by **Powell et al. (2000a)** and **Powell et al. (2002)**.

3.3.1.4 Rule-Based Decision Making

The fourth group of dynamic myopic approaches can be classified as "Rule-Based" publications, since *easy decision rules* are applied as a reaction to dynamically occurring information. Some of the following papers also include some more advanced procedures (e.g. Local Search) for comparison purposes. The concept of "Fuzzy Logic" is included here, because it basically reproduces human decision making, by transferring it into a form of rule-based computer decision making.

All of the selected papers consider dynamically occurring customers; in addition, one paper also includes dynamic service times. The SLPDP(TW) is regarded three times; the other two papers deal with a TRP and an MLPDPTW, respectively. Table 3.5 summarizes the properties.

Rule-Based Decision Making

Regan et al. (1995) investigate the potential benefits of *en route diversion* for an SLPDP under idealized conditions. The respective publication is also the first explicitly considering *en route diversion*. The objective is the minimization of empty distance traveled. The authors first consider the idealized *one vehicle and two requests case*. In a circular area with the depot in the center, a vehicle starts traveling to the first order's Pickup location. While the vehicle is on its way, the second order arrives (with Call-In time uniformly and randomly distributed within the vehicle's travel time to the Pickup location).

	dynamic aspects orders	other	capaci-tated	problem	TW constr.	area	en route diversion	dynamic test data
Rule-Based Decision Making								
Regan et al., 1995	X		cap.	SLPDP			X	self-generated
Regan et al., 1996	X		cap.	SLPDPTW	hard	wide	X	self-generated
Regan et al., 1998	X		cap.	SLPDPTW	hard	wide	X	self-generated
Larsen et al., 2002	X (100%)	service time		TRP		local		self-generated
Fuzzy Logic Approach								
Teodorovic and Radivoj., 2000	X (100%)		cap.	MLPDPTW	hard	local		self-generated

Table 3.5: Rule-based decision making

A first rule-based diversion strategy, "divert if the new Pickup location is closer to the vehicle's current position", yields savings in traveled distance of less than 1%. When, in addition, the potential empty distance, according to the sequence of both orders, is taken into account, the average reduction in traveled distance is more than 6%.

In a second scenario, *several dynamic orders* occur, with an arrival rate rapid enough so that more than one demand may arrive while the vehicle is en route to the Pickup location. The authors introduce a benchmark solution approach, where optimal re-sequencing is performed at the completion of each loaded movement. If en route diversion is allowed in this approach, the results are improved by about 1% to 2%. However, diversion creates a sort of "zig-zag" effect, where a vehicle is en route and then diverts and then diverts again. Thus, the authors recommend limiting the number of times that one diverts before some demand is serviced and not allowing diversion whenever it is locally better.

The exploration of idealized scenarios suggests that a reduction of traveled distance of between 5% and 10% by applying en route diversion is not unreasonable.

In **Regan et al. (1996)**, the authors extend their problem for profitability-based acceptance/rejection decisions, where a new load is only accepted if the *empty-to-loaded ratio* is smaller than a prespecified threshold value. The ratio is calculated by creating an optimal tour, including the candidate load and the already accepted loads. The *additional empty distance is set in relation to the new order's loaded distance.* Again, the advantage of diversion strategies is proved, which result in a 5% to 7% reduction in overall empty-to-loaded ratio.

Regan et al. (1998) describe a simulation framework to dynamic fleet management systems for the SLPDPTW. They discuss three *load acceptance* and eight *load-to-vehicle assignment* strategies.

The following *load acceptance strategies* are proposed:

(i) a new order is accepted if the number of loads waiting in the system is smaller than a prespecified number ("capacity-based strategy"),

(ii) a new load is accepted if it can be feasibly inserted into the current plan ("feasibility-based strategy"),

(iii) a load is accepted if the *empty-to-loaded ratio* is smaller than a specified threshold value ("profit-based strategy").

For the task of *assigning orders to vehicles*, the following strategies are compared:

(i) First Come First Served,

(ii) Nearest Origin,

(iii) Bipartite Assignment of open orders to available vehicles, triggered by time,

(iv) Bipartite Assignment, triggered by the number of open orders and idle vehicles,

(v) Best Insertion plus Intraroute Changes,

(vi) Best Insertion/Intraroute Changes allowing for en route diversion,

(vii) Best Insertion/Intraroute Changes plus re-assignment of loads between vehicles,

(viii) Best Insertion/Intraroute Changes, allowing for en route diversion and re-assignment of loads between vehicles.

The strategies are evaluated with self-generated test data, based on a circular geographic region with a radius of 417 km. A comparison is made on the basis of the performance indicators average *empty distance*, *waiting time* and *operating profit* for high, medium and low demand environments. Somewhat different assumptions are used: while assignment strategies (i) to (iv) are combined with the simple capacity-based acceptance strategy, strategies (v) to (viii) are combined with the profit-based load acceptance including a time window feasibility check.

When comparing assignment strategies (i) to (iv), the authors find the best results with *Nearest Origin* in high demand environments, while *Bipartite Assignment* performs best in moderate demand environments. The more flexible strategies (v) to (viii) produce much lower waiting times and therefore better customer service for all demand intensities. However, good profit values can only be achieved for moderate demand environments; the profit values for high demand environments in particular are not competitive. The authors explain this finding with the fact that a significant fraction of requests is turned away in strategies (v) to (viii) because of the time window feasibility check. In a final suggestion, a hybrid system that chooses an assignment strategy based on the current congestion level of the system is recommended.

Larsen et al. (2002) investigate a Traveling Repairman Problem (TRP) with up to 100% dynamic customers and completely dynamic service times. The objective is the minimization of total traveled distance.

The authors propose four *Rule-Based solution approaches*:

(i) First Come First Served (FCFS),

(ii) First Come First Served with relocation to the geographic median when the vehicle is idle,

(iii) Nearest Neighbor, and

(iv) First Come First Served within four regions of the geographic area.

For testing purposes, the authors use some self-generated data sets with 20, 30 and 40 customers, occurring in a 10km×10km unit square. The data are constructed with degrees of dynamism ranging from 0% to 100% and with effective degree of dynamism ranging from 0% to 60%. When comparing the rule-based strategies, best results are achieved with *Nearest Neighbor*. Strategy (iv) produced slightly higher route lengths and FCFS, as well as FCFS with relocation to the median the longest. For increasing degree of dynamism, the authors report a linear increase in route length across all policies. Interestingly, for increasing *edod* between 48% and 57%, the results show decreasing total travel times. Generally, the results of different rule-based strategies converge with higher levels of dynamism.

Fuzzy Logic Approach

Teodorovic and Radivojevic (2000) investigate a capacitated MLPDP with soft time windows, where all orders occur dynamically. The associated decision problem is split into the subproblems "assignment" and "routing/scheduling" with two different objective functions. For the assignment decision, the objective is to minimize the sum of total distance traveled and waiting time; for the routing/scheduling decision, the goal is to minimize distance and time of detours for new customers.

The authors propose a *"Fuzzy Logic" method that tries to replicate a human dispatcher's decision-making process, based on previous decisions taken by a skilled dispatcher.* For the assignment decision, in a first step a *membership function* is derived. This function transforms (the explicitly calculated) additional vehicle distance and additional waiting time into the categories "big", "medium" or "small".

Afterwards, an *approximate reasoning algorithm* translates the findings into a dispatcher's preference strength. When, for example, both the additional distance and the additional waiting time are "small", the preference for assigning the new order to the associated vehicle is "very strong". Within the pool of vehicles, those currently traveling are favored over idle vehicles. A similar procedure is also applied for the routing/scheduling decision. The authors test their approach with some self-generated instances and report "very promising results".

3.3.1.5 Multi-Agent Systems

Since Multi-Agent Systems differ substantially from the previous approaches, they are treated separately. In contrast to other procedures, there is no global view. Instead, multiple interacting agents with specific objectives decide about subproblems. Solving dynamic myopic SLPDPTW's with a Multi-Agent system was considered by a publication of Mes et al. (2007), whose specifications are given in Table 3.6.

Mes et al. (2007) consider an SLPDP with soft time windows, which is in particular

	dynamic aspects		capaci-	problem	TW	area	en route	dynamic
	orders	other	tated		constr.		diversion	test data
Mes et al., 2007	X	several	cap.	SLPDPTW	soft	local		real-life and self-generated

Table 3.6: Multi-agent systems

applicable for local area scheduling of Automated Guided Vehicles (AGV). Dynamism is primarily induced by requests; in addition, a single random variable is modeled, specifying the total time from arrival at the Pickup location until completion of service at the Delivery location. Since a vehicle moves empty immediately to the assigned Pickup location and waits over there, the random time interval includes: waiting for Pickup, loading at the Pickup location, driving from the Pickup to the Delivery location, waiting for unloading and unloading at the Delivery location.

The major contribution of the paper is the development of a *Multi-Agent Based Procedure*. Instead of a central planning instance, the authors model several agents (for every vehicle and every request) that interact with the help of a market mechanism. The *job agent's* objective is to arrange transportation of the corresponding load before due time at minimal costs, while *vehicle agents* try to maximize their profit by deploying capacity. Both meet at the "marketplace", where job agents request prices for their specific transportation task. Each vehicle agent submits a quote, based on its current scheduling. Afterwards, in a *Vickrey auction manner*, "the best (lowest) price vehicle agent" wins the bid (getting a payment for transportation equal to the second lowest offer). If all quotes are above a certain threshold (calculated with respect to the request urgency), the assignment of the request may be postponed by the job agent.

Improvement is achieved by specific agents: A *fleet agent* is responsible for a subset of vehicles and tries to re-assign jobs between these vehicles. A *shipper agent* is responsible for a set of orders, he may re-allocate orders within the already acquired transportation capacity of its job agents.

The procedure is tested with a data set derived from an AGV system at the Amsterdam Airport, Schiphol. For benchmarking, the authors use hierarchical scheduling methods (cp. Ebben et al., 2005), which distribute vehicles amongst nodes at the top level, while actual load-to-vehicle assignment is performed at the node level. These simple approaches are significantly outperformed by the proposed multi-agent procedure, especially in terms of empty travel time and total costs.

The group of *dynamic myopic approaches without knowledge of the future* has now been considered in detail. The approaches which are presented in the following section also have not available any knowledge of the future. However, there is a decisive difference to the previous ones: They try to identify and to apply ways of anticipating the future.

3.3.2 Strategies Anticipating the Future without Knowledge of the Future

We now give attention to publications which propose strategies for how to construct myopic solutions in order to leave open space (route slack) for the viability of future yet unknown events. These strategies try to reduce the probability that an urgent request (with tight time windows) arrives and the only vehicles that can serve it are already committed, so that servicing this new request may have to be delayed or the request may even be rejected. These strategies also try to postpone final fixation of decisions for as long as possible, in order to have more options to incorporate further new information.

Basically, three concepts have been proposed: *vehicle waiting strategies*, *different objective functions for short and long-term decisions* and *request postponement of non-urgent requests.* In addition, an extension to the Multi-Agent-based procedure by Mes et al. (2007) is presented, which leaves open time slack by applying opportunity-based bid-pricing. In a strict sense, the acceptance-rejection strategies by Regan et al. (1996, 1998) are also directed to “leave open some slack” for profitable future requests.

The **advantageousness of different waiting strategies**, in order to efficiently distribute waiting time along a dynamically constructed route, was compared by **Mitrovic-Minic and Laporte (2004)**. The authors consider an MLPDP with hard time windows for a local area courier service.

Waiting after service allows the accumulation of requests by the planner, which may result in better routing and scheduling decisions. However, waiting may also result in some wasted time that could have been used to serve additional requests. Four waiting strategies are included in the comparison:

(i) *“Drive First”* - A vehicle leaves its current location at the earliest possible departure time. This may result in waiting time at the next location if the vehicle arrives before the time window opens.

(ii) *“Wait First”* - A vehicle leaves its current location at the latest possible departure time, that means it arrives at the next location at the end of this location’s time window. An advantage is that more requests are known at the time the vehicle leaves, resulting in more potential for optimization. On the other hand, more vehicles are required, because the strategy tends to concentrate long waiting time in the first part of the route, leaving too little waiting time in the remainder.

(iii) *“Dynamic Waiting”* - The requests in a route are clustered to “service zones” (related to spatial and time distance). Within each service zone, vehicles drive according to “Drive First”, between service zones according to the “Wait First” strategy. The strategy is illustrated in Figure 3.7 (cp. Mitrovic-Minic and Laporte, 2004).

(iv) *“Advanced Dynamic Waiting”* - Identical to “Dynamic Waiting” with the extension that total waiting time between service zones is spread proportionally.

The authors’ algorithmic approach uses Best Insertion to generate an initial solution, supplemented by Tabu Search improvement, based on an ejection chain neighborhood. Test data, with completely dynamic requests, were derived from two courier companies, operating in Vancouver, Canada. Best results are achieved for the “Advanced Dynamic Waiting” strategy, which produces solutions with up to 8% shorter route lengths compared to “Drive First” and close to or shorter route lengths compared to “Wait First”. The number of vehicles is either close to the number of vehicles used in “Drive First” or better. “Advanced Dynamic Waiting” also outperforms “Dynamic Waiting” in all aspects.

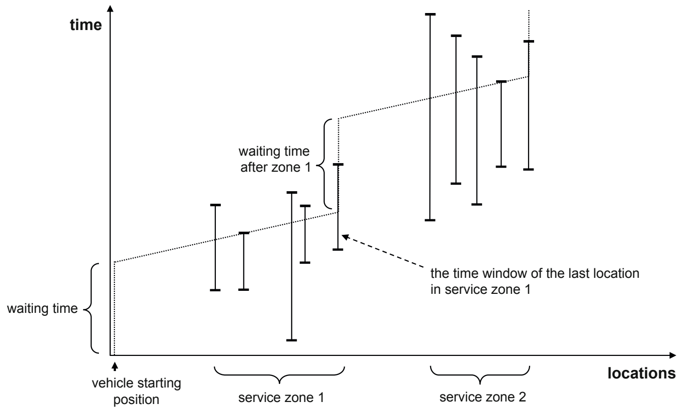

Figure 3.7: Waiting times generated by a Dynamic Waiting strategy

Another paper, considering the **advantageousness of dynamic waiting strategies** was published by **Branke et al. (2005)**. In contrast to Mitrovic-Minic and Laporte (2004), the authors regard a VRP where only one dynamic customer occurs. The objective is to find a waiting strategy in order to maximize the probability that this customer can be serviced within a feasible plan.

Several simple waiting strategies and an Evolutionary Algorithm are presented. The strategy "variable" ("serve all known customers in a tour and finally wait at the last customer's location") and the Evolutionary Algorithm perform best in simulations on a modified test data set based on Beasly (1990). It is shown, compared to the reference strategy "not to wait", that the best waiting strategies are able to reduce the probability of not being able to serve a customer by 10%, while the average length of the detour for a new customer was reduced by approximately 35%.

In a second work, **Mitrovic-Minic et al. (2004)** consider the identical problem to Mitrovic-Minic and Laporte (2004), but now investigate the application of a double horizon based heuristic. By using **different objective functions for short and long-term decisions**, the authors try to achieve better flexibility to incorporate future events. The short-term goal is to reduce traveled distance, while the long-term goal in addition considers maintaining the routes in a state with plenty of options for future requests. Decisions which restrict future planning options are penalized.

The authors compare the *double horizon approach* with a *standard rolling horizon approach*, applying the Tabu Search heuristic introduced in their preceding paper. Again, two test data sets from Vancouver courier services are used, which both range over a 10-hour service period. The short-term horizon is empirically chosen as one hour and two hours, respectively. The authors report superior behavior of the new double horizon based approach, leading to improvements in total route length ranging from 3.6% up to 7.6%.

Pureza and Laporte (2008) investigate a **combined vehicle waiting and request buffering strategy** for a dynamic MLPDP with hard time windows, with the objective of minimizing a weighted cost function (including: number of lost requests, number of used vehicles and total traveled distance). Customers occur dynamically, in addition time-dependent travel times are considered. The authors first prove the advantageousness of a *basic waiting strategy (WE)* compared to "Drive First". A waiting time at the present location is scheduled in order to avoid early arrival (and waiting) at the subsequent location. The waiting time is determined to be just as long as to ensure punctual arrival at the subsequent location's EPT. This enables the consideration of new incoming events in the short-term routing, which may change the decision of the next planned locations, or leave it unchanged without delaying the beginning of service. This basic waiting strategy was also successfully employed in Fleischmann et al. (2004).

Afterwards, *WE* is extended with a version *making use of time-dependent travel times, WE_FP*. It is evaluated whether indirect paths to reach a location yield shorter travel times, thus allowing for a further increase in waiting time at the present location. For a dynamized Li and Lim (2003) PDPTW data set, the authors report a reduction in lost requests of 2% to 8.3% and a reduction of used vehicles by 1.9% to 4.3% when applying WE_FP (compared to the basic version of WE). In a further step, a *request buffering strategy (WE_RB)* is added, which postpones the assignment of some non-urgent new requests to the next route planning cycle. This add-on achieves a further increase in solution quality, especially the traveled distance can be systematically reduced.

Finally, the authors report best impacts of their combined waiting and buffering strategy for a degree of dynamism between 0.4 and 0.6. If the *dod* is too low (<0.2), there is actually no positive effect at all. Only minor positive effects are reported for high degrees of dynamism. Due to dispersion of incoming requests over time, postponement activities in the beginning do not result in a sufficient mass of decision options, thus diminishing the strategy's ability to produce improved results.

In a further *Multi-Agent based publication*, **Mes et al. (2010)** propose a concept of how to improve the pricing technique of an individual *vehicle agent*. They introduce a *time slack measure*, which indicates the maximum amount of time that a job can be postponed by, without causing an increase in delay (for itself or one of the succeeding jobs). When calculating a *bid price* for a possible new job, the *decrease in time slack* between already accepted jobs is considered. With the help of historical demand data, the authors translate the required time slack into an expected profit of future moves, which has to be at least compensated by the reached bid-price of the possible new order. The authors test their **opportunity-based bid-pricing approach** with a small self-generated data set with up to nine nodes and ten vehicles and report a 10% reduction of system-wide logistical costs if the new concept is used by all vehicle agents.

3.3.3 Dynamic Stochastic Approaches with Explicit Knowledge of the Future

Information about the future is usually given in the form of probability distributions, e.g. covering the spatial and temporal occurrence of new orders. A popular approach for handling future information, which is applied in five of the subsequently twelve presented

	dynamic aspects		stochastic information	capacitated	problem	TW constr.	area	solution approach	test data
	orders	other							
Bent and van Hentenryck, 2004	X (80%)		expected total number of requests, probability distribution of temporal request arrival per customer location	cap.	VRPTW	hard	local	sampling "multiple scenario approach"	Solomon 1987
van Hemert and La Poutre, 2004	X		probabilities about "fruitful regions", where new loads are likely to occur	cap.	VRPTW	hard	local	sampling "probable orders"	self-generated
Ichoua et al., 2006	X (75%)		orders occur according to Poisson process with arrival rate depending on geographic region and time period	uncap.	VRPTW	soft	local	waiting opportunity	self-generated
Hvattum et al., 2006	X (50%)		orders occur according to Poisson process with arrival rate depending on geographic region	cap.	VRP	hard	local	(i) recourse function, (ii) sampling	real-life
Hvattum et al., 2007	X (50%)	amount of demand	orders occur according to Poisson process with arrival rate depending on geographic region	cap.	VRP	hard	local	sampling with improved selection	real-life
Ghiani et al., 2009	X		requests arrive according to known stochastic process	cap.	MLPDPTW	soft	local	sampling (short-term horizon)	self-generated
Kim et al., 2004	X (100%)		spatial and temporal distributions of priority demands	cap.	SLPDPTW	hard	wide	feasibility index (to serve future priority demands)	self-generated
Yang et al., 2004	X (100%)		uniformly distributed customer locations	cap.	SLPDPTW	soft	wide	opportunity costs of serving new jobs	self-generated
Powell, 1996	X		load distribution by origin, destination and Call-In time	cap.	SLPDPTW	hard	wide	multistage stochastic network with approximate recourse function	real-life
Spivey and Powell, 2004	X		the complete future, included in "gradients"	cap.	SLPDP		wide	opportunity cost arcs	self-generated
Larsen et al., 2004	X (23%)	service time	orders occur according to Poisson process with arrival rate depending on geographic region, probability distribution for service times	uncap.	TSPTW	soft	local	probability based reallocation of idle vehicles	self-generated
Liao, 2004		travel time	probability of link travel times, depending on the time of day	uncap.	VRP		local	avoidance of probably congested links	real-life

Table 3.7: Publications with dynamic stochastic solution approaches

dynamic stochastic publications, is *sampling*: the algorithm generates a sufficient number of future scenarios (by drawing from the given probability distribution) and uses the scenarios to approximate a decision's impact on the future.

Another option is the *direct usage* of probability distributions, for example to decide whether a vehicle should wait at its current location or whether it should be relocated to another promising location. The given probabilities may also be used for calculation of *recourse functions*, which include the costs that an assumed scenario does not occur.

Table 3.7 summarizes the main features of the selected *dynamic stochastic publications*. As in the dynamic myopic case, the sources of dynamism are given in the first two columns. Afterwards, the available stochastic information is specified (column 3). In columns 4-7, the associated problem is defined, as well as the geographical extension. The subsequent eighth column includes the solution approach, which is used to integrate the given stochastic information. Finally, information about the used test data sets is provided (column 9).

Bent and van Hentenryck (2004) consider a VRP with hard time windows and with up to 80% dynamic customers. As stochastic information, the algorithm knows the expected total number of customers, customer locations, and the probability distribution of temporal request arrival per customer location. The objective is to maximize the number of serviced customers.

As solution procedure, the authors propose a *Multiple Scenario Approach*: a pool of feasible plans **(sample scenarios)** is maintained, each plan including known and unknown future requests. Whenever a new plan needs to be generated, future requests are randomly drawn out of the known probability distributions. So the resulting plan leaves room for accommodating future requests if they materialize.

A dynamically occurring request is accepted if it can be feasibly inserted into at least one plan in the pool. All plans are continuously kept up-to-date. A plan is deleted, if new information makes it unrealizable. The plan for execution is chosen with the help of a *consensus function*, which selects the plan most similar to the current pool of routings (for every routing, it is calculated how often identical routing decisions, e.g. from location A to B, can be found in other routings). So a *preferably robust plan* is selected, in order to accommodate many dynamic customers in the future.

The authors use modified 100-customer Solomon (1987) test instances and compare the cases of having information about the future available, or not. In the second case, the average number of unserved customers for four different classes of test instances is 1.5, 3.5, 2.3, and 6 on average. In the first case, the average number of unserved customers decreases to 0.75, 1.2, 1, and 2. In general, dramatic improvements by exploiting stochastic information are reported, observing more benefits in environments with a higher degree of dynamism.

In a similar approach, **van Hemert and La Poutre (2004)** deal with a dynamic stochastic VRP, where dynamic loads have to be picked up and transported to a depot within hard time windows. As stochastic knowledge, probabilities are available about "fruitful regions", where dynamic loads are likely to occur. The objective is the maximization of transported loads.

The authors propose an Evolutionary Algorithm, with an initial population of 30 individuals. In addition to known orders, *probable orders*, based on the probabilities for future requests, are generated **(sample scenarios)**. The fitness of the individuals is evaluated by the number of real and *probable* orders which could be feasibly inserted, where the weight of *probable orders* is decreased by a factor α.

When an event occurs, the *best individual* out of the population is chosen as plan in execution. This plan may include anticipated moves, that means vehicles may drive to nodes that have not requested service. The performance of the approach is tested with some self-generated test data. The authors report encouraging results and benefits by the use of information about the future.

Ichoua et al. (2006) examine an uncapacitated VRP with soft time windows, representing an application of Express Mail Service, where parcels are picked up from customers (75% of customer orders occur dynamically) and brought to a central office for further processing. The operational area is partitioned into geographical zones. As information about the future, it is known to the algorithm that orders occur according to a Poisson process, with specified arrival rates depending on geographical region and on time period.

The authors expand the parallelized Tabu Search algorithm with Adaptive Memory, introduced by Gendreau et al. (1999), for a **waiting opportunity**: (i) if the vehicle's next destination is far enough, (ii) if there are not too many other vehicles in the current zone, and (iii) if a new customer is likely to unfold in the vehicle's proximity within a specified time period δ_k, a vehicle is required to wait at its current location for the specified time period.

The approach is tested with self-generated customer data, with associated locations in a 5km×5km unit square. The advantageousness of the new waiting strategy is proved by comparison with the original algorithm: total travel time and lateness can be reduced by 2.3% on average. It is also noted that the new strategy is more effective when it is applied on harder problems (i.e., smaller fleet size or higher request arrival rates.).

Hvattum et al. (2006) observe a dynamic stochastic planning problem at "Linjegoods AS", a distribution company in Norway. The problem is traced back to a capacitated VRP (Pickup), with 50% of the orders arriving dynamically. The geographical area is divided into n×n sectors, where probabilities that a customer shows up in a specific sector are described as a Poisson process. In contrast to Ichoua et al. (2006), the arrival rate depends only on the geographic sector, not on the time of day.

To capture the stochastic elements, the authors first analyze the application of a *two-stage stochastic model with recourse function*, where all unknown information is assumed to be revealed at time t. However, computing the expected recourse cost for a particular solution turns out, even for this simplified case, to be exceedingly difficult.

So the authors proceed with a *rolling horizon based heuristic*, which solves a set of **sample scenarios** (Best Insertion is applied to known customers and to randomly drawn future requests) and then uses "common features" from the resulting solutions to build a preferably "good plan". The decision to serve a customer in the interval under consideration becomes more attractive if such a decision has also been taken in various sample scenario solutions. The objective is to minimize the number of vehicles and total travel time, with greater weight on the first factor.

For comparison purposes, the new approach is benchmarked with a myopic heuristic, which ignores all probabilistic information. While routes produced by the new approach are about 15% shorter, the average number of used vehicles increases slightly. Nevertheless, significant savings are yielded by the new heuristic using stochastic information.

In a subsequent publication, **Hvattum et al. (2007)** modify the selection process from the pool with the **sample scenario** solutions. Instead of only counting the frequency with which a decision can be found in the pool, the authors also include an evaluation step, which tries to avoid overall poor effects. With respect to the objective function, this step discloses the possibility to exclude a few customers with high (solution pool) presence from service and to include a few customers with low (solution pool) presence for service.

In addition, the authors extend their procedure for the case of customers with *stochastic demand* which is revealed first at customer location arrival. If the vehicle discovers that the demand of the customer is higher than the available vehicle capacity, it has to skip the customer completely (split transportation of a load is not allowed) and follow the remaining parts of its tour. The customer has to be serviced later by another vehicle.
Finally, the authors perform some tests for the new program version and report quite good performance, even on problem instances that have radically different properties as compared to the instances for which it was intended.

A similar procedure is proposed by **Ghiani et al. (2009)** for the dynamic stochastic MLPDP with soft time windows. Stochastic information about customer arrivals is used to generate a specified number of **sample scenarios**, which in contrast to Hvattum et al. (2007), cover only a *short-term horizon*. After Best Insertion of randomly drawn requests, the resulting sample scheduling is used for *approximating the future impact of a new real request's insertion*.

The authors benchmark their "anticipatory procedure" with a purely reactive algorithm (not taking into account knowledge of the future) and achieve "dramatic benefits" with the new procedure in objective function value. Since the objective function only covers the minimization of user inconvenience (delay), it should also be mentioned that other important aspects, such as average vehicle utilization, show significantly worse behavior (a decrease of 19%).

Kim et al. (2004) investigate a truck dispatching problem (SLPDPTW) with two types of orders: low price normal orders with wide time windows and high price priority orders with narrow time windows. In an oversaturated system with more than enough requests (100% dynamic), an acceptance/rejection decision has to be made in order to maximize profit. Knowledge of the future is available in the form of spatial and temporal distributions for priority demands, as well as average haul length and required empty distance for this type of order.

Whenever a new request occurs, it is tested whether it can be feasibly inserted into the current plan (allowing for en route diversion). If the test is positive, a priority order is immediately accepted. For a normal order, a **feasibility index** is calculated, based on the current state of the system and the potential inclusion of the normal order. *The index approximates the expected number of vehicles that would be able to serve future priority demands.* The normal order is only accepted if the feasibility index exceeds a prespecified threshold.

Test data sets are self-generated, so that approx. 30% of the demands have to be rejected with an efficient dispatching algorithm. The fraction of priority demands is between 6.25% and 25%. The authors compare their new approach with the rule-based benchmark policies "accept if feasible" and "accept if current number of orders in the system is below a threshold." While the total number of accepted demands is quite similar with all three policies, the number of accepted priority demands can actually be increased by the new solution approach. This results in a significant improvement in total profit.

Yang et al. (2004) deal with an SLPDPTW, where all customer orders arrive dynamically. As information about the future, customer locations are known to be uniformly distributed in a unit square. The objective is the minimization of a weighted cost function for empty movement, for delay, and for lost revenue from job rejections.

The authors propose five planning approaches to support the acceptance/rejection decision. All policies are used to calculate a new order's marginal insertion cost. If marginal insertion cost is smaller than prospective revenue, an order is accepted. In policy (i), marginal cost of serving a new request is calculated over all vehicles by inserting the new request at the end of each vehicle's queue. In policy (ii), all possible insertion positions in each vehicle's queue are considered. Policy (iii) considers the possibility of re-ordering waiting requests within each vehicle's queue. Policy (iv) optimally solves the acceptance and allocation decisions for all open orders with ILOG. Policy (v) in addition incorporates knowledge of the future, by introducing the *opportunity costs of serving new jobs.* Based on the uniform distribution, for a request with central Delivery location it is more likely to find a subsequent order with little empty movement. Hence, **central locations are favoured and remote locations are penalized.**

The paper reports on results obtained with self-generated test data. Optimization policies, simultaneously considering all open orders, appear to outperform the more limited local policies by a significant margin. The worst performance is achieved by policy (i), the best performance by taking into account the future job distribution in policy (v). However, the size of instances which could be optimally solved with ILOG was limited to twenty open orders.

Powell (1996) investigates a dynamic SLPDPTW, where information about the future is available in the form of load distributions by origin, destination and Call-In time. In addition to truck-to-load assignment, it has to be decided whether a driver should be *held in a region or* whether he should be *repositioned empty to a neighboring region* (both, in anticipation of future loads). The goal is the minimization of a weighted cost function, including cost for empty movement, cost for waiting, and cost for rejection of a load.

The author presents a solution approach, which is based on a *stochastic network* with two components: an *assignment network*, including known loads, and a *forecast network*, including forecasted loads as well as known loads lying in the future. Network arcs represent driver-to-task assignments (in the assignment network) and loaded moves, empty moves or waiting times (in the forecast network). An **approximate recourse function** is represented by a cluster of *recourse links*, which capture the expected marginal contribution of each unit of flow into a region in a time period. An approximation of this recourse function value is added to the arc cost and the resulting problem is then solved with a network simplex algorithm on a rolling horizon basis.

The new approach is benchmarked with a completely myopic version of the algorithm, using test data derived from a major truckload motor carrier. Results indicate that the dynamic stochastic approach outperforms the dynamic myopic one by 15%. In addition the author investigates the impact of the total number of trucks (density) on overall profitability and reports substantial improvements ($0.05 per mile) when a larger fleet of vehicles is used.

Spivey and Powell (2004) present a strategy to incorporate advance information into a simple linear assignment model for the dynamic SLPDP. The assignment model explicitly allows "not assigning" a resource/order with the help of arcs connecting each resource/order to a corresponding super-sink.

Information about the future is assumed to be completely available and is made known to the solution approach in the form of different types of *gradients*. That means the assignment problem's arcs are manipulated in order to produce solutions anticipating future information. To realize this task, three different types of gradients are defined: **resource gradients**, **task gradients**, and **arc gradients**. A *resource gradient*, for example, can be viewed as the contribution for not assigning the corresponding resource (vehicle). It is added to the regular cost-value of each arc, which is connected to the specific resource, thus decreasing the vehicle's current attractiveness in anticipation of the future. *Task gradients* work equally for open orders. An *arc gradient* is more specific and captures the impact on the future for each specific arc.

For small problems the gradients are calculated by enumeration of future resources and tasks, for bigger problems a hierarchical aggregation strategy is proposed. The authors compare the solutions achieved with the application of gradients with simple myopic solutions. As expected, all gradient solutions turn out superior to myopic solutions. The best results are achieved with specific *arc gradients*, where solution quality reaches "near optimal levels". However, the computational burden for the arc gradients is the highest, requiring a calculation for every arc, and not just for every node.

Based on a planning problem of a mail service provider in the US, **Larsen et al. (2004)** investigate a dynamic TSPTW, where between 11% and 23% of customer requests and all on-site service times are subject to dynamism. The geographical area is divided into several sub-regions, in which orders occur according to a Poisson process with region specific arrival rates. This information and the probability distribution of on-site service time is a-priori known to the planning approach. Decisions about routing and scheduling for a single vehicle have to be made, with the objective of minimizing the weighted sum of travel time and lateness.

A rolling horizon based solution approach is presented, which uses Best Insertion and subsequent improvement with 3-opt. *A-priori information is only utilized for potential re-allocation of vehicles during idle time* (in anticipation of a better location to serve future loads). Three strategies are compared with a reference strategy of just waiting at the current location:

(i) re-allocation to the nearest prespecified idle point,

(ii) re-allocation to the idle point with the highest arrival rate, and

(iii) re-allocation to the idle point with the highest expected number of immediate requests, depending on vehicle idle time and chosen idle point.

The proposed **re-allocation strategies** are only executed if the probability of receiving at least one new request within the vehicle's idle time is sufficiently high in the chosen idle point's subregion.

The authors perform tests with two data sets, one self-generated, the other based on real-world data. Interestingly, best results in terms of distance and lateness for the first data set are achieved by the reference strategy of just waiting at the current location. For real-world test data, however, strategy (ii) performs best, while the reference strategy proves competitive as well. Results show that using information about the future in the suggested way may not (!) lead to significant improvements.

Liao (2004) reports on a dynamic VRP in the Taichung network in Taiwan and focuses on dynamic travel times. As information about the future, the planning system has available probable link travel times, depending on the time of the day. The objective is the minimization of total travel time.

The author's solution procedure consists of an initial route generation by a Nearest Neighbor heuristic, which is followed by Tabu Search improvement with neighborhood 2-opt. Each time, when changes in link travel times emerge, the travel time matrix is updated with a shortest path algorithm. As result, a vehicle may be re-routed. Probabilistic information about the future is used to generate a **temporal tabu list**, in order to avoid possibly congested links in the traffic network.

In tests based on real-life data, the impact of having available dynamic travel time information is compared to just knowing static travel times. Results show that the objective function value can be significantly improved by real time routing when considering information about the future: decreases in travel time range from 19.74% up to 24.48%.

3.4 Dynamic Test Data

This section reviews the most frequently applied dynamic test instances. Interestingly there are not so many publicly available dynamic data sets: many authors use "self-generated" or "real-life" data sets, but do not explicitly provide these data sets for other authors. The same is true for the results: in many cases, only selected criteria of the objective function are reported. In addition, the reported criteria are often highly aggregated average values. Hence, these data are only partially applicable for comparison purposes.

Figure 3.8 summarizes the selected dynamic test sets. The first column includes the basic problems, the second column lists the associated sources, and the third column shows the underlying static sources, if those have been used to derive the dynamic instances.

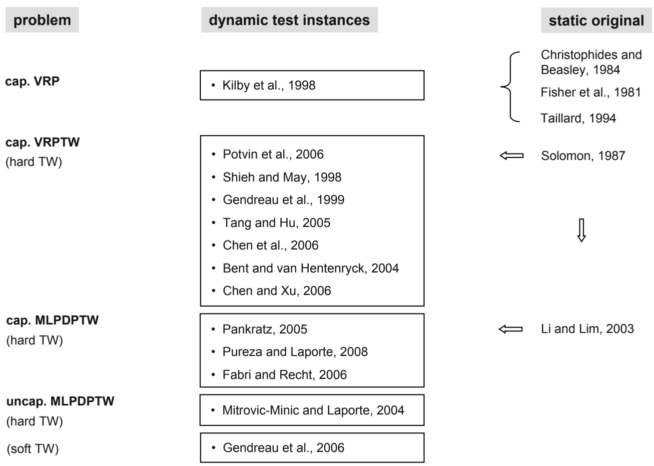

Figure 3.8: Dynamic test instances

Dynamic test data for the **capacitated VRP** were generated by Kilby et al. (1998): the authors take the static instances, published by Christophides and Beasley (1984), Fisher et al. (1981) and Taillard (1994) as a basis and extend them by Call-In times and service times for each task. The *Call-In times* are chosen according to a uniform random distribution throughout the whole planning horizon. The resulting dynamic data sets, however, were used only by a few authors, e.g. by Montemanni et al. (2005).[8]

For testing the **capacitated VRP with hard time windows**, many authors propose dynamic extensions to the test data published in Solomon (1987). Since the original data set evokes so much interest, a detailed summary of the original static data set is given, followed by seven approaches of bringing dynamism into the instances.

In the original 56 data sets, 100 customers are spread in a 100×100 unit square with varying geographical distributions: uniformly and randomly distributed (problem classes R1, R2), clustered (C1, C2), and semi-clustered (RC1, RC2), the last representing a mixture of uniformly and randomly distributed and clustered customers. The problem classes of

[8] The Kilby et al. (1998) data sets are no longer available at the author's homepage. Hence, we refer to http://www.fernuni-hagen.de/WINF/menuefrm/publik.htm. Here the data sets can be found in the folder "Montemanni et al. 2005".

type 1 possess narrow time windows at the depot, so that only a few customers can be serviced in each route; in contrast, problem classes of type 2 possess wider time windows at the depot. Travel times between the customers are taken to equal the corresponding distances, which are calculated as Euclidean distances. Each customer requires an individual service time (10 minutes for R1, R2, RC1, RC2 and 90 minutes for C1, C2). For servicing the customers, an unlimited number of homogeneous capacity constrained vehicles is available at a central depot. The vehicles have to return to the depot within the specified opening time (hard constraint).

The Solomon instances have been employed for tests of many new *static* algorithms, which have produced high quality and even optimal solutions. The availability of these solutions is an advantage when solution quality of "dynamized" Solomon instances has to be evaluated with *optimal static solutions.* For a discussion of advantages and disadvantages of this kind of "performance analysis", see Section 2.1.

Comparison of solution quality is also possible *by relative comparison of two dynamic approaches, applied to the same test data set.* Unfortunately, this is complicated here by the fact that every author generates his "own dynamic extension" to the Solomon instances. Thus, the *"same test data set"* assumption is no longer fulfilled. Subsequently, seven different ways of calculating the *Call-In time* for a specific request i are reported. The "U" denotes a uniform random distribution within a given interval.

- Call-In(i) = EPT(i) · U(0,1). *(Potvin et al., 2006)*
- Call-In(i) = MAX(0, LPT(i) - constant - U(0, LPT(i))). *(Shieh and May, 1998)*
- Call-In(i) = U(0, MIN(EPT(i), t_{i-1})), where t_{i-1} is the departure time from i's predecessor in the best known solution for the static problem. *(Gendreau et al., 1999)*
- Call-In(i) = U($c_1 \cdot$ EPT(i), $c_2 \cdot$ LPT(i)), where c_1 and c_2 ($0 \leq c_1 \leq c_2 \leq 1$) are two parameters. *(Tang and Hu, 2005)*
- Call-In(i) = MAX(0, EPT(i) - 1.5 · t_{Oi} - r), where t_{Oi} denotes the travel time between depot and node i, where r = U(0, EPT(i) - 1.5 · t_{Oi}). *(Chen et al., 2006)*
- Call-In(i) = U((k-1) · H/3, MIN(λ_i, k $\cdot (H/3) - 1$), where k denotes an interval of the planning horizon H, where λ_i denotes the latest time a vehicle can depart from depot, service i and return to the depot. *(Bent and van Hentenryck, 2004)*
- Call-In(i) = U(0.5 · MIN(EPT(i), LPT - $d_i - \Delta - \tau$), MIN(EPT(i), LPT-$d_i - \Delta - \tau$)), with d_i denoting the travel time from depot to customer i, with Δ denoting the time between two consecutive decision epochs, with τ denoting the computational time. *(Chen and Xu, 2006)*

Obviously, there are many different ways of calculating a Call-In time.

In a next step, test data for the dynamic **capacitated MLPDPTW with hard time windows** are investigated. The available test sets are also based on the previous Solomon instances. A transformation of the static VRPTW data into static MLPDPTW data is accomplished by the following two authors: while Nanry and Barnes (2000) simply pair up the customers appearing in the routes of the best known Solomon VRPTW solutions *one* *by one* (regarding the "optimal" order), Li and Lim (2003) *randomly* pair up customer locations within routes of solutions obtained with their own heuristic solution approach. Both approaches render 100 VRPTW requests into 50 MLPDPTW requests.

Dynamic components have been added to the Li and Lim (2003) instances in three different ways:

- Pankratz (2005) first introduces the variable $t_r^{latest}(i)$, which is calculated as follows:

  $t_r^{latest}(i)$ = MIN(LPT(i), LDT(i) - $t_{PD} - t_{Service}) - t_{Depot,P}$,

  where t_{PD} denotes the direct travel time from Pickup to Delivery of request i, where $t_{Service}$ denotes the service time at the Pickup location, and where $t_{Depot,P}$ denotes the direct travel time from depot to the Pickup location. Afterwards, based on $t_r^{latest}(i)$, dynamic instances are generated with the formula

  Call-In(i) = a $\cdot\, t_r^{latest}(i)$,

  with a varying from 0.1 to 1.0 in steps of 0.1.

- A second approach is proposed by Pureza and Laporte (2008). They calculate

  Call-In(i) = MIN(EPT(i), MAX(U(1,5), LPT - $t_{Depot,P} - \beta$)),

  where $U(1,5)$ denotes an integer number uniformly randomly generated between 1 and 5, where $t_{Depot,P}$ denotes the direct travel time from depot to the Pickup location of request i at time t=0 and where β is chosen to take one of the values 0, 100, 200, 300. According to the authors, the formula does not guarantee service since the Delivery location restrictions are not taken into account and in the case of time dependency, the travel time used in the computation belongs to the specific first period.

- A third option was chosen by Fabri and Recht (2006). They generate dynamic arrival times with the formula:

  Call-In(i) = U(0, MIN(EPT(i), LPT(i) - $t_{Depot,P}$)).

In contrast to the previous dynamic test data, the following two publications for uncapacitated MLPDP test data do not rely on any available static instances.

Dynamic test data for the **uncapacitated MLPDP with hard time windows** were proposed by Mitrovic-Minic and Laporte (2004). The authors generate their own 40 test instances, based on real-life data from two courier companies operating in Vancouver (Canada). Up to 1000 requests (100% dynamic) occur in a 60km×60km geographical area with Call-In time being calculated according to a uniform random distribution over the whole planning horizon.

Finally, dynamic data sets for the **uncapacitated MLPDP with soft time windows** published by Gendreau et al. (2006) have to be considered. The data generation process as well as the achieved results stand out due to very detailed and convenient description.

As this data set is used for benchmarking purposes of the later proposed algorithmic procedures, it is refered to Section 4.4.2 for further analysis.

At the end of this section, it can be noted that there are not many publicly available dynamic test instances. Most instances concentrate on local area VRP(TW) applications, followed by some local area MLPDPTW instances. No dynamic instances at all are available in the category of wide area transportation problems, especially for the SLPDPTW.

3.5 Acceptance of Dynamic Planning Applications in Real-Life

Most of the surveyed dynamic publications report “dramatic benefits” and “high cost reductions”. However, most results are obtained in artificial test environments. In exceptional works, **Powell et al. (2000b)** and **Powell et al. (2002)** describe the challenging experience of transferring a dynamic planning algorithm for a wide area SLPDPTW into a running real-life application.

According to the authors, the real-life application of dynamic Decision Support Systems often (and especially in their case) does NOT result in dramatic benefits. The relatively small success of computer-based planning systems is particularly attributed to low “user compliance” (the rate with which a human dispatcher accepts the recommendation of a computer system), which is often below 60% in the truckload trucking area.

Reasons for this behavior can be discovered by comparing the different solution approaches of human dispatchers and of mathematical optimization systems. A human’s decision is *state-action based*, producing *locally greedy optimizations* mostly neglecting the effects downstream in space and time. A mathematical model’s decision is *cost minimization based*, producing a *global solution*.

Although producing a global solution (on a given data set), there are also some *drawbacks of the computer-based system*. The accurateness of the model’s real world description may be limited, in particular in dynamic situations, when subproblems are solved at a point in time where availability of data is rather limited. In addition, the data in the computer may be generally imperfect or incomplete. A human planner may possess information (acquired by phone, conversation or visual inspection) that has not yet been entered into the system. “Implicit information”, meaning general experience, for example based on historical events, may also be hardly available to a computer system.

Many decisions are fairly obvious, meaning that human and computer will coincide in these instances. Hence, a problem arises when the “higher reasoning” of the computer produces decisions that differ from the pattern-based reasoning of the dispatcher. Then the human has a dilemma: *is the discrepancy a result of “higher reasoning” or a simple data error?* Especially in *real time problems*, where fast decisions are elementary, it is not easy to find out if a computer’s decision is plausible. Often the dispatcher will go with his own intuition.

The authors suggest a *hybrid approach*, generating a computer solution between “global” and "greedy", in order to improve solution acceptance by human dispatchers. They introduce a random variable, which decides if the computer-generated "truck-to-load assignment" is accepted by the human dispatcher (this is more likely if the computer solution exhibits more similarities to a greedy solution). Furthermore, a factor α is introduced, which represents the degree of global optimization: for full global optimization its value is 1, for full greedy optimization its value is 0, intermediate strategies are represented by α-values in the interval (0,1).

In several simulations with dynamic orders and dynamic travel times, the authors evaluate correlation of α and user compliance: when varying α for given compliance probabilities (100%, 70%, 40%), best results are achieved for α-level 0.75, interestingly even in the 100% compliance case.

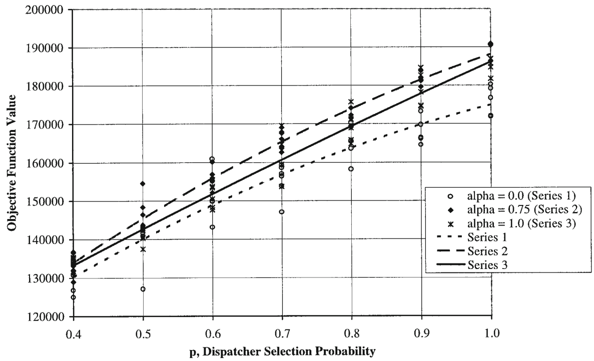

Figure 3.9: Impact of user-compliance and α on obj. function values (Powell et al., 2000b)

Figure 3.9 shows an original graph from Powell et al. (2000b), which includes the effects of varying user-compliance probabilities (x-axis) for given α's (1.0, 0.75, 0). The associated objective values are plotted on the y-axis. The authors discover that as the level of user compliance drops, the value of a globally optimal solution (α=1) over a greedy solution drops dramatically. On the other hand, if user compliance is high (over 90%), the value of global optimization (either $\alpha = 1.0$ or $\alpha = 0.75$) is quite high and demonstrates the usefulness of optimization models.

In a comparison, **Powell et al. (2002)** contrast the hypothetical case of perfect user compliance in global optimization (α-value = 1) with the case of perfect user compliance in completely "manual" planning (α-value = 0). They achieve *better results, in the range of 5% to 10%, when applying pure global optimization and assuming perfect user compliance.*

The problem of user acceptance of computer-generated solutions is also reported in Bell et al. (1983). The authors tackle the problem in a similar way as described in Powell et al. (2000b): they replicate human decision patterns. Neighboring customers are aggregated and treated as one single customer, which results in solutions similar to those the dispatcher is used to see. So the acceptance of computer solutions is increased. In addition, it is reported that once dispatchers felt comfortable with the system, most schedulers began to ask for size expansion of the neighborhoods to allow more options.