

# Integrated Solutions for Delivery Planning and Scheduling in Distribution Centres

Galina Merkuryeva and Vitaly Bolshakov

**Abstract** This chapter presents integrated solutions for product delivery planning and scheduling in distribution centres. Integration of cluster analysis, forecasting, computer simulation, metaheuristic optimisation and fitness landscape analysis allows improving planning decisions at both tactical and operational levels. Scheme for integrated solutions is provided. Integrated solutions are described and illustrated by a demonstration case for a regional distribution centre and a large network of retail stores. Cluster analysis and classification methods are applied to determine typical demand patterns and corresponding tactical product delivery plans. A multi-objective optimisation approach is introduced for grouping stores by their geographical location and demand data. Different simulation optimisation scenarios to define the optimal delivery routes and schedules at the operational planning level are considered and compared. Potential applications of a fitness landscape analysis for adjusting an optimisation algorithm are described and applied for product delivery scheduling. Two-stage vehicle routing supplement with vehicle scheduling is described and shown in application experiments. Optimisation techniques described in this chapter are applicable to solve specific routing and scheduling tasks in logistics.

## 1 Introduction

To ensure business competitiveness, modern management practices require application of different methods in the fields of information technology and operations research. To find the best solution to the problem, these methods must be integrated to complement each other for mutual benefit. Cluster analysis, computer simulation and metaheuristic optimisation techniques may be applied to provide an integrated planning and scheduling of deliveries from a distribution centre (DC) to a net of regional stores.

---

G. Merkuryeva (✉) · V. Bolshakov  
Riga Technical University, Riga, Latvia  
e-mail: Galina.Merkurjeva@rtu.lv

V. Bolshakov  
e-mail: Vitalijs.Bolsakovs@rtu.lv

## ***1.1 What is Problem Complexity?***

Product delivery planning and scheduling is a high commercial priority task in transport logistics. In real-life applications the problem has different stochastic performance criteria and conditions. Optimisation of transportation schedules itself is computationally time-consuming task which is based on the data from tactical planning of weekly deliveries. This chapter also focuses on the methodology that allows reducing the effect of the demand variation on the product delivery planning and avoid numerous time-consuming planning adjustments and high computational costs.

In distribution centres, this problem is related to deliveries of various types of goods to a net of stores in predefined time windows, taking into account transportation costs and product demand variability. The problem has a high number of decision variables, which complicates the problem solution process. In practice, product demand from stores is variable and non-deterministic. As a result, the product delivery tactical plan that is further used for vehicle routing and scheduling has to be adjusted to real demand data, and product delivery re-planning supervised by a planner is often required. This task is very time consuming and requires specific knowledge and experience of planning staff in this domain.

## ***1.2 What is Motivation for Integrated Solutions?***

An acceptable vehicle schedule can be created with the help of commercial scheduling software usually based on heuristic optimisation. However, in practice a schedule created with heuristic algorithms cannot satisfy all constraints of the real-life problem. In this case, an analyst needs to modify or correct a schedule generated by a standard software in order to satisfy the problem specific constraints and to adapt this schedule to a new information received by a planner. Simulation modelling can be applied in this case to numerically evaluate the efficiency of a new schedule candidate.

Integration of different techniques such as cluster analysis, forecasting, computer simulation and optimisation to product delivery planning and scheduling allows improving tactical and operational decisions in logistics distribution centres. A cluster analysis of product demand data of stores allows identifying typical dynamic demand patterns and associated product delivery tactical plans. Customer regional clustering based on multiple criteria is performed through multi-objective optimisation. Meta-heuristic optimisation provides effective techniques to define optimal parameters of product transportation and delivery schedules. Vehicle scheduling may be performed also for the routed solution. Fitness landscape analysis allows enhancing the optimisation process and tuning of parameters of optimisation algorithms. An attention should be given to simulation optimisation of the vehicle schedule, which allows incorporating stochastic data. Integration of these technologies advances traditional optimisation techniques known in operations management and lead to coordinated decisions in the area of delivery planning and scheduling at different management (strategic, tactical and operational) levels.

### ***1.3 What are the Main Objective and the Problem Solution?***

*The main objective* is to prepare an effective tactical plan for product deliveries from DC to a net of stores for the upcoming week. In practice, the real demand can be very different from the expected or average demand. Hence, significant changes should be made in the delivery plan for each new week. The reasons for the demand variance can be product demand seasonal effects or marketing events. In this case, it is very important to reduce the effect of demand variation on the delivery planning process and avoid numerous time-consuming adjustments of the base delivery plan.

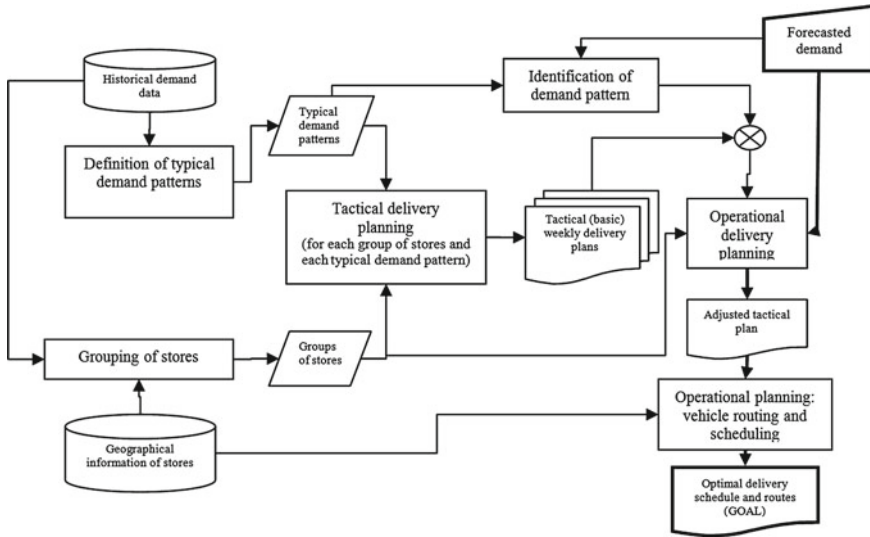
*The problem solution* is a detailed delivery plan, in which schedules, routes and amounts of goods to be delivered are defined as the best ones for the input data defined. Input data contain information on the historical demand and location of the stores, available vehicles for the product transportation and existing rural delivery routes. *Additional constraints* such as time windows for product deliveries to specific stores need to be taken into account. An optimal delivery plan should satisfy the following criteria:

1. An amount of goods delivered to the stores should be equal to the demand of these stores for a particular day.
2. Product delivery costs have to be minimised. This implies sub-criteria such as the number of vehicles used to deliver all goods should be decreased, and transportation costs should be minimised by optimising delivery routes and schedules.

## **2 Integrated Solution Scheme**

This solution scheme provides selecting an appropriate product delivery tactical plan for the upcoming week and optimising product transportation routes and delivery schedules. This is achieved by integration of a cluster analysis to define typical product dynamic demand patterns and identify an appropriate demand cluster and tactical weekly delivery plan, and using simulation and optimisation techniques to model and optimise vehicle routes and delivery schedules for product deliveries.

Vehicle routing and schedule optimisation is based on the data from tactical planning for a week delivery. At the same time, a weekly delivery plan is dependent on the data about a number of goods to be delivered to stores in a particular day of a specific week and geographical allocation of stores. In practice, historical data of store demands can be very different from expected or average one, which is determined in a predefined or base plan. Thus, significant changes should be made in the base delivery plan to be adjusted for each new week. So, it is reasonable to specify typical patterns of dynamic daily demand for different planning weeks and introduce several base plans each representing an appropriate product delivery timetable for a specific demand pattern. This will reduce the work of adjusting a typical or base delivery plan to the current situation. Since there are now more typical delivery plans that are based on typical demand patterns, the work will be reduced to making a decision, which delivery plan should be used for the next week and small adjustments of it still



**Fig. 1** Scheme of integrated solution

may be required. In addition, selecting the most suitable delivery plan may ensure better scheduling solutions and reduce their computational costs.

The integration scheme for the problem solution (Fig. 1) includes the following main tasks [20]:

1. Definition of typical dynamic demand patterns by clustering historical daily demand data available for different planning weeks.
2. Grouping of stores based on their geographical locations to leverage the total product demand over regions.
3. Tactical weekly delivery planning performed for each group of stores and each demand pattern.
4. Identification of a specific demand pattern based on the classification model created for typical dynamic demand patterns and selection of an appropriate tactical delivery plan for the new week.
5. Adjustment of a selected tactical weekly delivery plan to a new or forecasted demand.
6. Vehicle routing and scheduling, e.g. by using metaheuristic optimisation methods [21, 22].

### 3 Cluster Analysis of Dynamic Demand Data

This section describes methods for determination and recognition of weekly patterns of dynamic demand data. Determination of patterns of similar dynamic demand data allows introducing base delivery plans each representing an appropriate product

delivery timetable for a specific demand pattern. Described methods allow determining a number of typical demand patterns based on historical demand data, as well as defining characteristic features of these patterns and identifying which weekly delivery plan would be the most suitable for the forthcoming week. An example of cluster analysis of dynamic demand data is provided.

### 3.1 Motivation

It is assumed that it is possible to specify *typical patterns* of dynamic daily demand for different planning weeks and introduce several *base plans* each representing an appropriate product delivery timetable for a specific demand pattern [17].

The required weekly *tactical delivery plan* is found based on information on the weekly demands during one year. The objective is to find certain dynamic demand patterns, which combines weeks into groups in the way that demand data are similar for all weeks within a specific group but different from those weeks that belong to other groups.

In this case, a cluster analysis of historical demand data [27] provides an opportunity to divide a variety of planning weeks into clusters and to find a number of clusters that represent weeks with a specific demand pattern. It also gives information for the construction of the classification model to identify which weekly delivery plan would be the most suitable for the forthcoming week [17].

Thus a cluster analysis of dynamic demand data is used to:

1. Find a number of typical dynamic demand patterns and corresponding clusters of planning weeks;
2. Construct a classification model that for any week allows determining an appropriate demand pattern, allocating a specific week to one of previously defined clusters and determining correspondent product delivery plan.

### 3.2 Determination of Typical Dynamic Demand Patterns

To identify typical dynamic demand patterns based on historical demand data, or its observations, the k-means clustering algorithm [15] is applied. It divides  $n$  observations into a user-specified number  $k$  of clusters, in which each observation belongs to a cluster with the nearest mean value representing a cluster centroid. The result is a set of  $k$  clusters that are as compact and well-separated as possible.

K-means clustering is a partitioning method that operates on actual observations and creates a single level of clusters. Thus, for large amounts of data, k-means clustering is often more suitable than hierarchical clustering.

K-means clustering uses an iterative algorithm that minimises the sum of distances from each object to its cluster centroid, over all clusters. This algorithm moves objects

between clusters until this sum cannot be further decreased. The result is a set of clusters that are as compact and well separated as possible. Like many other types of numerical minimisations, the solution that k-means clustering reaches often depends on the starting points. It is possible for an algorithm to reach a local minimum, where reassigning any one point to a new cluster would increase the total sum of point-to-centroid distances, but where a better solution does exist. However, it is possible to overcome that problem by performing a cluster analysis multiple times and selecting the best result.

While implementing k-means clustering algorithm, its parameter  $k$  that defines a number of the resulted clusters needs to be specified. In case of typical demand patterns' determination, this number corresponds to an approximate number of such patterns and can be calculated using the following methods.

### 3.3 Definition of an Appropriate Number of Demand Patterns

There exist a number of approaches to find the best number of clusters by completing cluster validity or measuring goodness of the clustering results compared with ones created by other clustering algorithms. Here, an appropriate number of  $k$  clusters, or typical demand patterns is defined by using silhouette plots [14]. In this method, a numerical measure of how close each point is to other points in its own cluster compared to points in the neighbouring cluster is defined as follows:

$$s_i = \frac{b_i - a_i}{\min(a_i, b_i)}, \quad (1)$$

where  $s_i$  is a silhouette value for point  $i$ ,  $a_i$  is an average dissimilarity of point  $i$  with the other points in its cluster, and  $b_i$  is the lowest average dissimilarity between point  $i$  and other points in another cluster. Higher mean values of silhouettes show better clustering results that determine better clusters giving the best choice for a number of clusters.

Another method uses the Davies–Bouldin (DB) index [9], which defines the average similarity between each cluster and its most similar one and is calculated by a formula:

$$DB_n = \frac{1}{n} \sum_{i=1}^n R_i, \quad (2)$$

where  $n$  is a number of clusters and  $R_i$  is defined as follows:

$$R_i = \max_{i=1, \dots, n, i \neq j} R_{ij}, \text{ and } R_{ij} = (s_i + s_j) / d_{ij}, \quad (3)$$

where index  $R_{ij}$  is a similarity measure between two clusters and is based on a measure of dispersion  $s$  of a cluster  $i$  and a dissimilarity measure  $d_{ij}$  between two clusters.

### 3.4 NBTree for Dynamic Demand Pattern Recognition

A classification model that assigns an appropriate demand cluster is presented by an NBTree, which induces a hybrid of decision tree and Naive Bayes classifiers. This algorithm is similar to classical recursive partitioning schemes, except that leaf nodes created are Naive Bayes categorizers instead of nodes predicting a single class [27].

For a specific week and demand time series, a cluster is identified by determining a proper leaf number  $C$  according to the decision tree. When the leaf number is known, a cluster is estimated by a formula:

$$C = \arg \max_{c_j=C} P(c_j) \prod_{i=1}^m P(a_i|c_j), \quad (4)$$

where  $P(c_j)$  defines the probability that weekly demand belongs to cluster  $c_j$ , and  $P(a_i/c_j)$  defines a conditional probability that demand on day  $a_i$  belongs to cluster  $c_j$ . Probabilities  $P(c_j)$  are calculated from clustering results, while  $P(a_i/c_j)$  are defined from the classifier according to the above-determined leaf number.

For a specific week, an NBTree allows identifying an appropriate cluster and then choosing weekly tactical delivery base plan corresponding to this cluster. The selected weekly delivery plan is then used for the optimisation of parameters of vehicle schedules.

### 3.5 Example Applications

Let us assume that historical demand data for 52 weeks are available and specified by weekly demand time series each representing a sequence of points—daily numbers of the product deliveries for a specific week (see Fig. 2).

K-means clustering experiments are performed on the historical demand data for a number of clusters from 2 to 8. Then for each clustering experiment, silhouette plots are built and mean values of silhouettes per cluster are calculated (Fig. 3).

Analysis of silhouettes mean values leads to the conclusion that the best cluster separation could be done at  $k = 4$  that gives the highest silhouette mean value equal to 0.558. Clusters 1–3 seem to be appropriately clustered (see Fig. 4).

However, silhouettes values for the cluster 4 are negative. Theoretically, weekly dynamic demands assigned to this cluster could be better allocated to another cluster. These weeks are different due to demand dynamics and specific days, where demand peaks are observed.

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
2885	3390	3891	4115	4612	4687	3371
2831	3553	3859	3785	4432	4899	3527
2763	3548	4067	4631	4838	5057	3511
2951	3820	3787	4366	5075	4694	3345
2595	2731	3101	2988	3385	3524	2643
3150	3459	4339	4377	5187	4956	3545
2394	3229	3643	3693	4018	4411	3583

Fig. 2 Sample demand data

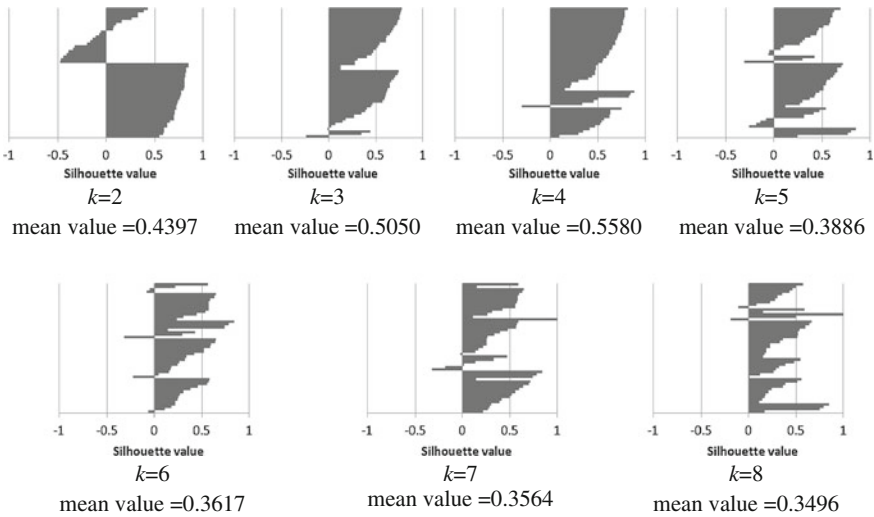


Fig. 3 Silhouette plots for the number of clusters  $k = 2$  to  $k = 8$

Reallocation of ‘unlike’ weeks avoids receiving negative silhouette values (see Fig. 5). However, this does not provide an increase of the silhouette mean value as might be expected. In this case, ‘unlike’ weekly demands behave as a ‘noise’ in their ‘native’ clusters, decreasing silhouette values. Then, clustering experiments have been performed with 49 weeks, where three ‘unlike’ weeks have been excluded from a cluster analysis. This has allowed increasing the silhouette mean value up to 0.5822, while getting the same groups of data clusters 1–3.

The DB index is calculated based on the output data from k-means clustering experiments for  $k$  in the range 2–8 clusters (Fig. 6). The lower value of DB index which is equal to 0.899 is received for the number of clusters equal to 4. Thus, the best number of clusters in this case confirms the results of the silhouette plot analysis.

As a result, a number of clusters is set to  $k = 4$ . It is worth noting that a tactical weekly delivery base plan is defined for a cluster with a silhouette mean value greater than 0.5. In this case, a tactical product delivery base plan is selected, adjusted or



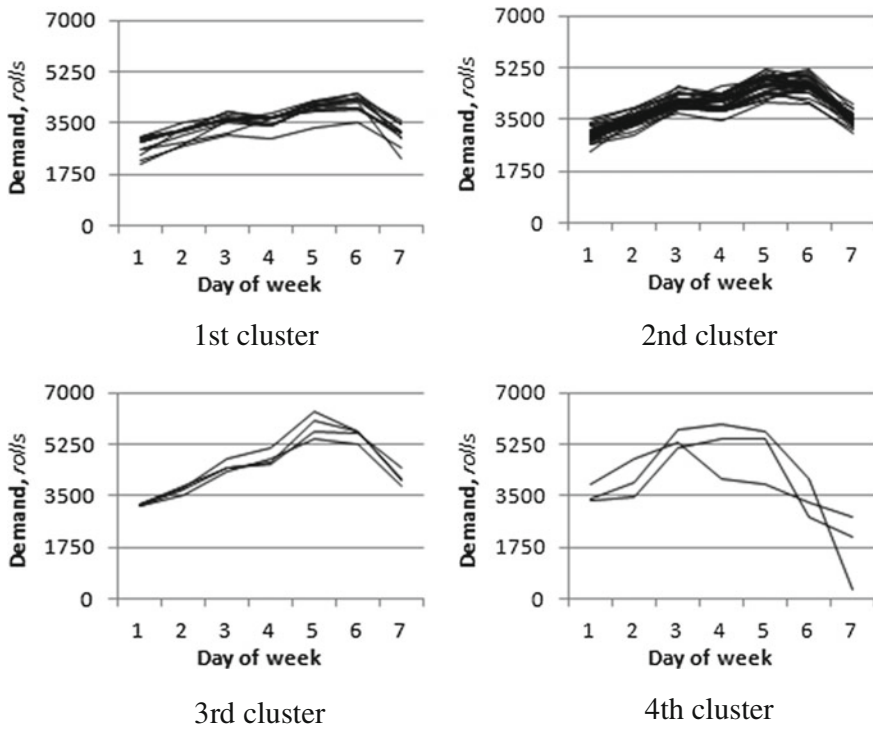


Fig. 4 Sample demand patterns found by k-means analysis for a number of clusters  $k = 4$

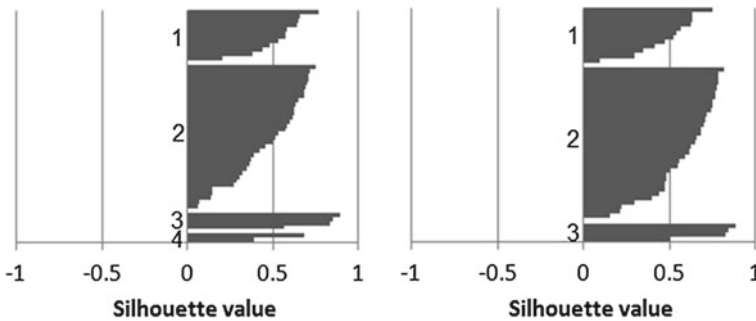


Fig. 5 Silhouette plots for the number of clusters  $k = 4$  with reallocation of 'unlike' weeks and for the number of clusters  $k = 3$  and 49 sample weeks

built for the first three clusters and not analysed for the last one. Dynamic patterns received for clusters from 1 to 3 are presented in Fig. 4.

To identify an appropriate demand cluster for a specific planning week, NBTree classification model is built (see Fig. 7). This will allow choosing a weekly delivery plan which corresponds to this demand cluster.

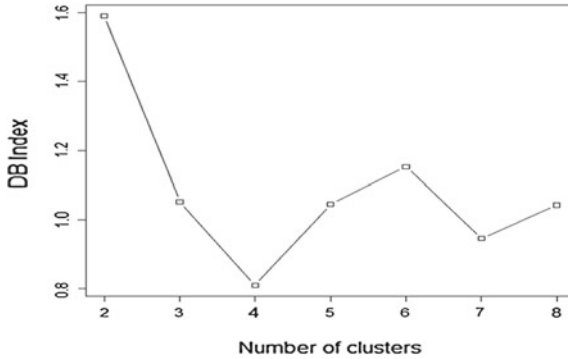


Fig. 6 Davies–Bouldin index for different numbers of clusters

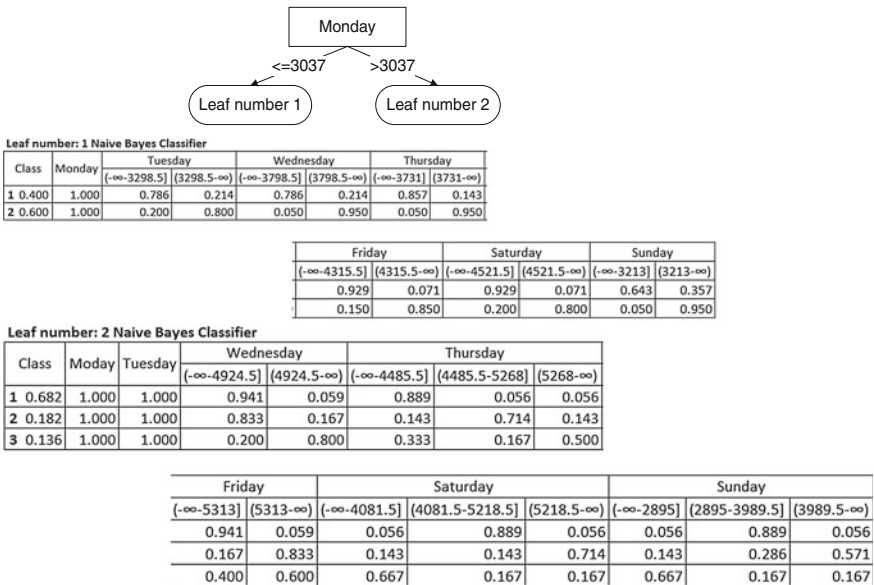


Fig. 7 NBTree-based classification model

To improve the performance of the classification model, demand data sample size has been increased up to 156 weeks. Two demand time series were generated for each planning week by daily demand uniform change within  $\pm 5\%$ . In a similar way, input data for another 52 weeks have been generated to validate a classification model itself. Built on this data the NBTree-based classification model with an example of the leaf Naive Bayes classifier is given in Fig. 8. In this case, tenfold cross-validation showed that only eight weeks have not been classified correctly, which produced an error value of about 5%.

```

FRI <= 4294.5
| TUE <= 3380.5: 1 cluster
| TUE > 3380.5: Leaf Number 1
FRI > 4294.5
| TUE <= 3700.5
| | MON <= 3062: 2 cluster
| | MON > 3062
| | | THU <= 4368.5: Leaf Number 2
| | | THU > 4368.5: Leaf Number 3
| TUE > 3700.5: Leaf Number 4
    
```

**Leaf Number: 4 Naive Bayes Classifier**

Class	Monday	Tuesday	Wednesday			Thursday		
			(-∞-4406]	(4406-5152.5]	(5152.5-∞)	(-∞-5467.5]	(5467.5-∞)	
2	0.348	1.000	1.000	0.727	0.182	0.091	0.900	0.100
3	0.522	1.000	1.000	0.067	0.867	0.067	0.929	0.071
4	0.130	1.000	1.000	0.167	0.167	0.667	0.200	0.800

Friday			Saturday			Sunday		
(-∞-5204]	(5204-∞)	(-∞-4299.5]	(4299.5-4903]	(4903-∞)	(-∞-1745.5]	(1745.5-3770]	(3770-∞)	
0.900	0.100	0.091	0.818	0.091	0.091	0.727	0.182	
0.357	0.643	0.067	0.067	0.867	0.067	0.067	0.867	
0.200	0.800	0.667	0.167	0.167	0.667	0.167	0.167	

Fig. 8 Detailed NBTree classification model

Table 1 Weekly demand sample data (in roll containers)

Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
3231	3462	3842	4437	5021	4868	3391

Let us assume that demand for Friday and Tuesday is 5021 and 3462 roll containers, correspondingly (see Table 1). From the classification model in Fig. 8, the leaf number 4 is selected.

Then a likelihood for cluster 2 is calculated by multiplication of probabilities received from the Leaf Number 4 Classifier for specific week days, i.e. 0.348, 0.727, 0.9, 0.9, 0.818 and 0.727. This probability is equal to 0.122. Likelihoods for cluster 3 and 4 are calculated and are equal to  $5.2 \cdot 10^{-5}$  and  $2.4 \cdot 10^{-5}$ , correspondingly.

The probability that weekly dynamic demand belongs to cluster 2 is defined as follows:

$$p(C_2) = \frac{0.122}{0.122 + 5.2 \cdot 10^{-5} + 2.4 \cdot 10^{-5}} \approx 0,9994. \tag{5}$$

Similar, probabilities that weekly dynamic demand belongs to cluster 3 or 4 are calculated:  $p(C_3) = 0.0004$ ;  $p(C_4) = 0.0002$ . Finally, a cluster with the highest probability is selected. As a result, the considered week belongs to cluster 2 with the probability which is very close to 100 %.

For a specific week, an NBTree allows identifying an appropriate demand cluster and then choosing weekly tactical delivery base plan corresponding to this demand cluster. Then, selected weekly delivery plan is used for optimisation of parameters of vehicle schedules.

## 4 Grouping of Stores Based on Geographical Locations

In case of a large number of stores in a delivery network, grouping of stores in relatively small groups based on their geographical locations allows to simplify the product delivery optimisation task by decreasing its dimension. In this section, the problem statement of store grouping that also provides the uniform distribution of the total demand over geographic regions is given, and problem-solving techniques are described and provided with illustrative examples.

### 4.1 Optimisation Problem Statement

In practice, weekly delivery planning is done based on the demand and data about store allocations to geographical regions. Grouping of stores based on their geographical locations allows leveraging the total product demand over regions. Let us suppose that all stores are grouped manually, and rearranging regions in case a new store is added may require. Also, it would be desirable to have separation of regions with a similar weekly total demand, or the total demand uniformly distributed over regions.

The use of a cluster analysis for dividing stores into regions according to their locations does not allow getting the total product demand equally distributed between these regions. But the region clustering task may be formulated as a multi-objective optimisation problem.

Input data contains the number of stores  $n$ , the number of regions  $k$ , two geographical coordinates  $x_i$  and  $y_i$  for each store  $i$ ,  $i = 1, \dots, n$  defined in the Cartesian coordinate system and the total weekly demand  $d_i$  for each store  $i$ .

Decision variables define a region (or cluster)  $a_i$  to which a store  $i$  is assigned:

$$a_i \in \{1, 2, \dots, k\}; \quad i = 1 \dots n. \quad (6)$$

Additional auxiliary variables are introduced as:

$$A_j = \{b | a_b = j\}, \quad (7)$$

where  $A_j$  is a set of stores assigned to each region, and

$$r(i, j) = \sqrt{(x_i - \dot{x}_j)^2 + (y_i - \dot{y}_j)^2}, \quad (8)$$

where  $r(i, j)$  defines an Euclidean distance from store  $i$  to the centroid of cluster  $j$ , and  $\dot{x}_j$  and  $\dot{y}_j$  are mean values of coordinates for all stores in cluster  $j$ :

$$\dot{x}_j = \frac{\sum_{i \in A_j} x_i}{|A_j|}, \quad \dot{y}_j = \frac{\sum_{i \in A_j} y_i}{|A_j|}. \quad (9)$$

Two objective functions are introduced in the problem. The first objective function determines how good regions generated from the geographical location point of view are, and the second one defines if the total demand is equally distributed among these regions. Both objective functions are minimised, i.e.:

$$f_1 = \sum_{j=1}^k \sum_{i \in A_j} r(i, j) \rightarrow \min, \quad (10)$$

$$f_2 = \sum_{j=1}^k \left| \sum_{i \in A_j} d_i - \frac{\sum_{i=1}^n d_i}{k} \right| \rightarrow \min, \quad (11)$$

where  $f_1$  defines the sum of distances between centroids of the regions and stores assigned to them, and  $f_2$  is the sum of differences of the total demand for each region and the average demand per region. No additional constraints are defined in the optimisation problem.

## 4.2 Multi-objective Optimisation Algorithm

As the problem has two objective functions, a multi-objective optimisation algorithm should be applied for grouping of stores into geographical regions. Thus for the grouping task, an application of widely used Nondominated Sorting Genetic Algorithm II (NSGA-II) [10] is proposed. Further discussion on application of NSGA-II implemented in the HeuristicLab optimisation framework [32] is given.

The optimisation problem itself is implemented as a multi-objective optimisation problem plug-in of HeuristicLab with integer encoding of solutions and their evaluation by two mathematical functions (10) and (11). Correspondingly, a chromosome representing a solution is defined as a string of  $n$  integer numbers and formalised as a vector  $(a_1, a_2, \dots, a_n)$  of decision variables, where  $a_i \in [1, k]$ .

In experiments with NSGA-II, the following GA operators are applied: (1) discrete crossover operator for integer vectors [13], (2) uniform One Position Manipulator [16] and (3) crowded tournament selector [10].

Parameters of the algorithm may differ for different sizes and complexity of the problem and should be tuned experimentally. For the case discussed in this chapter, with a number of stores equal to 88, the following parameters were determined in experiments: a termination criterion in a number of generations equal to 5000; the population size equal to 200; the crossover rate of 90%; the mutation rate of 5%; and 400 selected parents in a new generation.

### 4.3 Examples of Optimisation Experiments

The results of optimisation experiments for grouping of stores into geographical regions are performed for different numbers of generations with a population size of 200 solutions (see Fig. 9). Here, *demand quality* is equal to the sum of variances of the total demand for each region, and *geographic quality* corresponds to the sum of distances between stores in groups and centroids of these groups. An increase in the number of generations improves the Pareto front of non-dominating solutions by minimising both objective functions. However, when the number of generations exceeds 2000, these improvements are small. Finally, at 5000th generation the grouping of stores into geographical is reached that lead to a uniform distribution of demand between regions. Further improvements become minor for the discussed case.

However, graphical representation of the obtained results shows that solutions with high demand quality give worse results for the second objective function and vice versa. Thus, a solution in the middle of the Pareto front (see Fig. 10, where different regions with the stores assigned are shown) is selected which provides compact clusters of stores in regions. A large number of regions in the central part

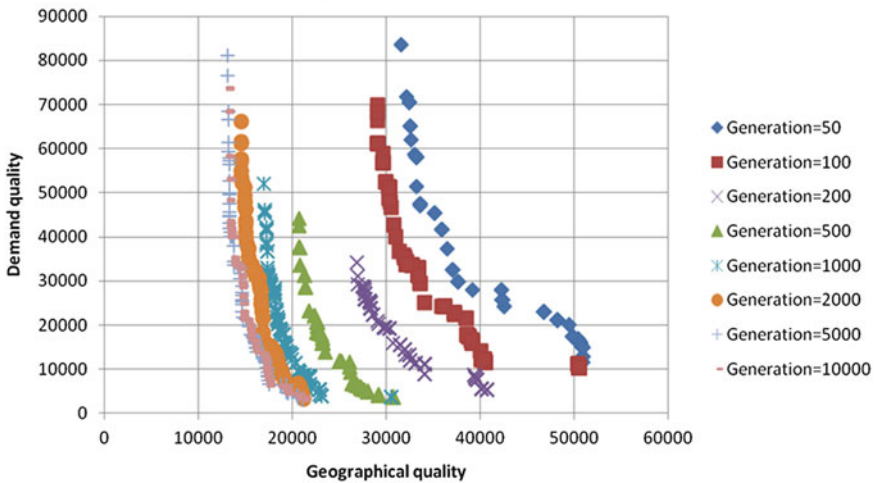
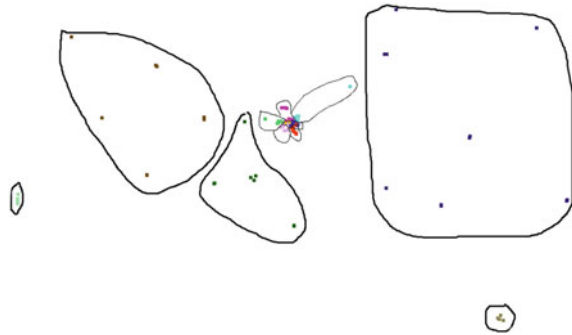


Fig. 9 Pareto fronts for different numbers of generations

**Fig. 10** Solution in the middle of Pareto front



of Fig. 10 may be explained by high density of stores with high product demands in this geographical part, which corresponds to a large city or urban area. Moreover, there are only two regions with demands that are lower than others. Further gradual leverage of the regional demand may be experimentally tested. In the example that is considered, this may worsen the geographical location of high-priority regions (see Table 2).

Further on, for each group of stores and each demand pattern a *weekly delivery base plan* that defines an amount of products to be delivered to stores on specific days of the calendar week (Table 3) is built. It is based on the average demand for each store from a specific group in the planning weeks that belong to a certain demand cluster. Here, a weekly delivery base plan is developed once using knowledge-based heuristics and updated relatively rare.

**Table 2** The distribution of the total weekly demand

Region	Total demand	Region	Total demand
1	28,443	7	25,328
2	21,429	8	21,552
3	23,440	9	21,787
4	21,687	10	12,152
5	23,583	11	14,722
6	23,101	12	21,860

**Table 3** Sample delivery plan for one week

Day	MON	TUE	WED	THU	FRI	SAT	SUN
Store A	30	35	30	25	45	50	30
Store B	–	25	–	–	35	–	–
Store C	20	–	15	–	10	–	25
Store D	–	25	–	40	–	35	20
Σ	50	85	45	65	90	85	75

Variation in daily deliveries for specific days of the week may produce extra costs on peak delivery rates. Thus, variation in daily deliveries needs to be minimised for each store and each group of stores. Additionally, customers with low demand would not be served each day as it will produce extra transportation costs to deliver small order quantities too often. This could be achieved by simple division of the total weekly demand into a number of delivery dates. For stores with low demand, a number of delivery days is reduced to minimise transportation costs. Afterwards, the weekly delivery base plan still needs to be further customised to correspond with demand dynamics during the week (see Table 3).

To ensure maximal operational effectiveness, the base plan selected is adjusted in order to match the actual customer demand for the forthcoming week. Then, for each day of the week, vehicle routes and/or delivery schedules are defined to minimise their transportation costs.

## 5 Simulation Optimisation of Vehicle Schedules

When the base delivery plan for the upcoming week is selected and adjusted, product delivery (vehicle) routes and schedules need to be optimised in order to reduce the transportation costs. Two types of a vehicle routing and scheduling problem are discussed. In this section, simulation-based vehicle scheduling when delivery routes are fixed and known is considered. Later on in this chapter a more complex case is introduced, when first the vehicle routes have to be determined and then these routes have to be scheduled.

Here, a vehicle scheduling problem with time windows is discussed, with a description of a vehicle schedule simulation model and scheduling optimisation scenarios.

### 5.1 Problem Express Analysis

A *vehicle schedule* defines a schedule of deliveries of various types of goods from DC to a network of stores. Distribution *routes or trips for vehicles* are fixed. For each route, the following parameters are defined: a sequence of stores (route points), average time intervals for vehicle moving between these points, loading and unloading average times and types of goods to be carried on this route. Goods are delivered to stores in the predefined time windows. For each store, an average demand of goods of each type is defined. Vehicle capacities are limited and known.

Vehicles are assigned to routes and schedules for routes are generated that minimise the total costs of a schedule. Samples of predefined vehicle routes and schedules that define at which time each route has to be started and which vehicle will perform it are shown in Fig. 11. The vehicle idle time is defined as a sum of time periods, when a vehicle is waiting for the next trip in the DC depot.



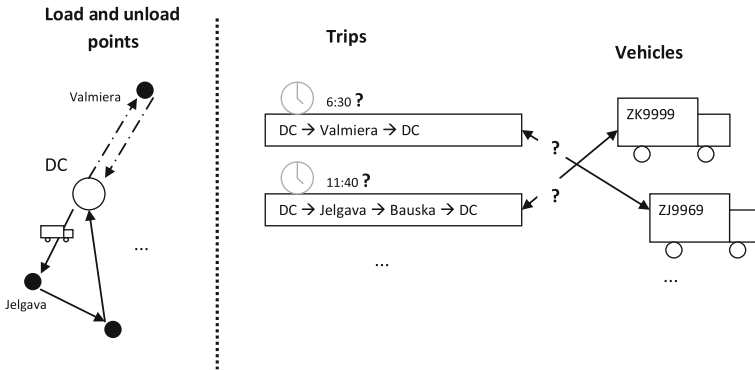


Fig. 11 Examples of vehicle routes

*Vehicle Scheduling Problems (VSP)* present a class of optimisation problems that are aimed at assigning a set of scheduled trips to a set of vehicles, in such a way, that each trip is associated with one vehicle, and a cost function for all trips is minimised [11, 24]. This problem is often modified with additional constraints, like time windows, different vehicle capacity, etc. Correspondingly, a VSP with Time Windows is denoted as VSPTW. A number of methods to solve VSP problems are proposed in the literature, e.g. integer programming, combinatorial methods, heuristics [11]. Such problems do not have efficient traditional optimisation methods and can be solved by application of evolutionary algorithms. Furthermore, an analysis of a fitness landscape can be used to evaluate the optimisation problem complexity and to select the most appropriate algorithm.

In practice, the VSP also can be complicated by stochastic processes existing in the problem, e.g. when the duration of a trip is a random variable. In this case, evaluation of potential solutions can be made through simulation, and simulation optimisation could be used to solve such problems. Simulation technology provides a flexible tool to determine the optimality of each solution. Therefore, the simulation-based fitness landscape analysis that supposes fitness evaluation of the solution with use of simulation becomes an important task.

### 5.2 Problem Statement

*Decision variables* are introduced to assign vehicles  $v_i$  to routes and define a start time  $t_i$  for each route [21, 22], where  $i$  is a route number,  $v_i$  is a vehicle assigned to trip  $i$  and  $t_i$  is start time of the trip  $i$ . The *objective function*  $f$  is aimed to minimise the total idle time for all vehicles:

$$f = \sum_{i=1}^N T_{idle}^i \rightarrow \min, \tag{12}$$

where  $T_{idle}^i$  is the total idle time for vehicle  $i$ ; and  $N$  is a number of vehicles.

The *problem constraints* are divided into three groups presenting vehicle capacity constraints, delivery time constraints, and gate capacity constraints, correspondingly. In the last case, a number of vehicles that can be loaded in a warehouse simultaneously cannot exceed a number of gates.

Express analysis shows that the problem could have many solutions that are not feasible within defined constraints. This makes a solution search process non-efficient in terms of computational time. To increase optimisation efficiency, all constraints are converted into soft constraints [21], and the objective function  $f$  in (12) is modified by introducing penalties taking into account the total number of times when the constraints were not satisfied by a potential solution:

$$f^* = \sum T_{idle} + k_1 T_c + k_2 T_m + k_3 T_o + k_4 N_{ol} + k_5 N_{ot}, \quad (13)$$

where  $f^*$  is the modified objective function;  $T_c$  defines the total duration of overlapping trips for one vehicle;  $T_m$  defines the total time of window mismatches;  $T_o$  and  $N_{ol}$  determine the total time and a number of vehicles that exceeded the total working time;  $N_{ot}$  is a number of vehicles overloaded. In (13), all indexes for unsatisfied constraints are multiplied with penalty coefficients  $k_i > 1$ ;  $i = 1..5$  that artificially increase the value of the objective function and make the fitness of a potential solution worse.

### 5.3 Simulation of Vehicle Schedules

To estimate fitness of potential schedule solutions, the vehicle schedule simulation model is introduced. It is built as a discrete-event simulation model, for example using AnyLogic simulation software [3]. It is based on the object-oriented conception and presents a simulation model as a set of active objects that are functioning simultaneously and interact with each other.

In the vehicle schedule model [22], each vehicle is modelled as an active object, and its behaviour is described by a state chart that defines vehicle states (e.g. parking, loading, moving and unloading) and transitions between them (see Fig. 12). The objects of each vehicle are aggregated by the model main object. Three classes are defined for store, trip and job objects to specify input data. During the model initialisation it is connected to a database of input data and variable collections of the main active object are set up with data from the database. Processes related to DC operations are simulated. During simulation the constraint violations are monitored and a number of violations are stored in the model.

In the model animation screenshot (Fig. 13), utilisation graphs (timelines) of all vehicles are combined in a Gantt chart for a vehicle schedule where different states of the vehicle are shown with different colours, e.g. the grey colour for parking in DC, green one for loading and blue for moving between route points times.

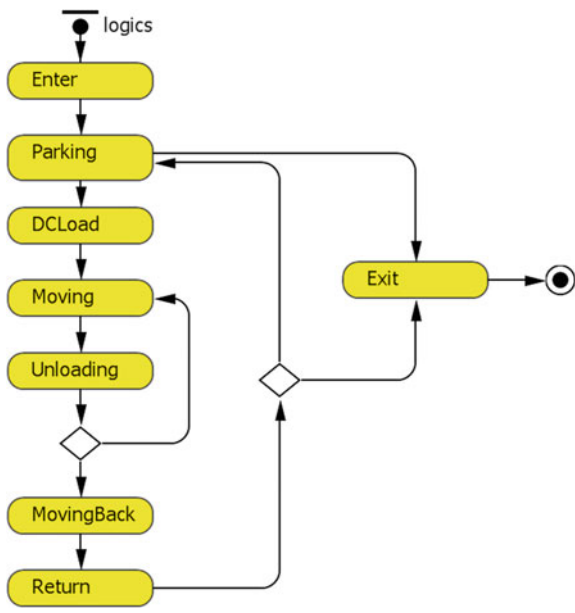


Fig. 12 State chart of active object “Vehicle”

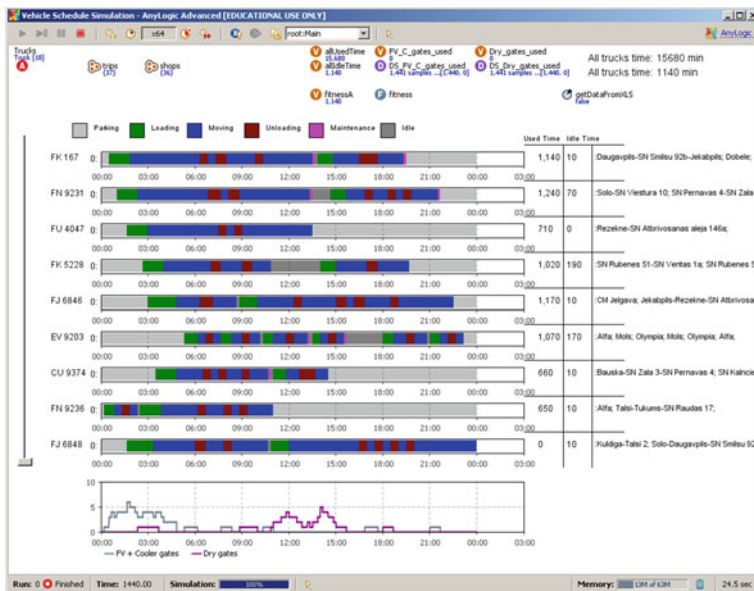


Fig. 13 Screenshot of the simulation model

The utilisation chart for DC gates during the daytime is displayed below the vehicle utilisation timelines. It shows how many loading gates are busy each time. Online statistics about the idle time and completed jobs is provided along the timeline for the corresponding vehicle. A list of performed trips including visited shops is generated for each vehicle.

## 5.4 Vehicle Schedule Optimisation Scenarios

Example input data of the VSP contains 37 routes, 17 vehicles and 36 stores. Specific parameters of vehicles, stores and routes are defined in the simulation model. The input data for vehicle moving times are interpreted as deterministic and then as stochastic (depends on the optimisation scenario). As simulation output the total idle time for all vehicles is calculated. The number of decision variables that define a vehicle schedule is equal to 74. Thus, exploration of the problem search space requires evaluation of a huge number of possible solutions [23]. Finally, function (13) is used for fitness evaluation of the solutions simulated. Two optimisation scenarios based on using genetic algorithms and a fitness landscape analysis are discussed in this section, and the third scenario based on scheduling of routed solutions is considered in Sects. 6 and 7.

### 5.4.1 Simulation-Based Optimisation with GA

In this scenario, a genetic algorithm (GA) is used to search for the best combination of the schedule parameters, while simulation model is applied to estimate quality of a schedule generated. The optimisation tool is implemented as a Java class, which interacts with the simulation model via ‘Parameter variation’ experiment in Any-Logic [18].

In GA, solution candidates of the scheduling problem are encoded as integer vector chromosomes, which length is twice a number of trips (routes). In chromosome, genes with even sequence numbers represent start times of corresponding trips, and ones with odd sequence numbers define a vehicle assigned for this trip. For example trip 1 will be performed by vehicle 2 starting at 12:20 a.m. (Fig. 14).

A mutation operator is introduced that changes one randomly selected trip in the solution candidate. For a selected trip, a new randomly chosen vehicle is assigned, and the start time is shifted by a certain constant value. One-point crossover with rate

2	20	1	600	2	690	1	120	3	490
Trip 1		Trip 2		Trip 3		Trip 4		Trip 5	

**Fig. 14** A sample chromosome of the vehicle schedule

of 75 %, above-mentioned mutation operator with rate of 1 %, one elite individual and tournament selection with tournament size of two individuals are involved in algorithm. Termination condition of GA is set to occur when there is no significant improvement of the best solution in population after a large number of generations.

Optimisation results show that a feasible schedule which satisfies all defined constraints can be found. Acceptable results are obtained with a population size of 1000 chromosomes, but larger populations significantly increase computational time. The solution allows decreasing the total idle time from 1140 to 700 min.

#### 5.4.2 Fitness Landscape Analysis and Optimisation Performance

While a genetic algorithm provides very good solutions in VSP optimisation, nonetheless it needs tuning of parameters and operator selection by skilled personal. To make tuning and adjusting of an optimisation algorithm easier *Fitness Landscape Analysis* (FLA) is proposed in the literature [26, 29]. FLA provides methods and techniques for a mathematical analysis of a search space of optimisation problems, and can be applied as a support tool to enhance optimisation of complex systems. It was proposed that the structures of a fitness landscape affect the way, in which a search space is examined by a metaheuristic optimisation algorithm. The fitness landscape analysis would allow getting more information on the problem's properties dependent on a specific optimisation method, which will guide the optimisation process.

FLA techniques apply different strategies for data collection based on simple moves, which generate a trajectory through the landscape (e.g. *random walk*). The *information analysis* interprets a fitness landscape as an ensemble of objects, which are characterised by their form, size and distribution and is based on the information theory. Four information measures are proposed by [30], e.g. the *information content* is a measure of entropy in the system and *partial information content* characterises the modality of the performed walk. Higher information content and partial information content values indicate higher hardness of the analysed problem. The *statistical analysis* proposed in [33], calculates the autocorrelation function in the random walk to measure the ruggedness of the landscape. In case of a high correlation between fitness values the landscape is considered less rugged and thus the problem should be easier. In this section, to analyse the problem fitness landscape the autocorrelation function is successfully used.

To apply FLA in the vehicle schedule optimisation the methodology described in [5] is applied. The optimisation problem is experimentally analysed in a comprehensive way to find how parameters of the problem and selection of optimisation operators and representations of solutions influence both FLA measures and optimisation performance. If the relationships between the fitness landscape measures and optimisation performance are found, afterwards, for the same class of the problem an optimisation algorithm can be adjusted based on the FLA results to a new problem instance.

To perform comprehensive optimisation and analysis of the VSPTW, the above described simulation model [22] was re-implemented as a plug-in of HeuristicLab framework [32] maintaining all model's logic and functionality.

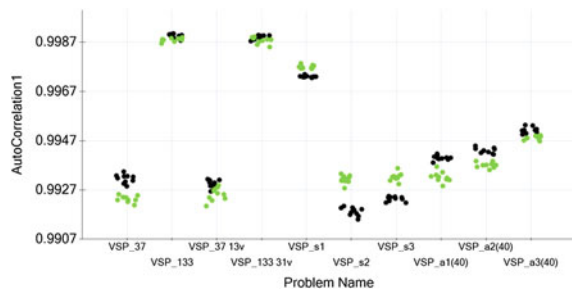
Hereinafter two types of mutation operators are defined for the representation described in this chapter. The single position replacement manipulator (*VSPManipulator*) changes the start time of the trip to a new uniformly distributed random number, but the single position shift manipulator (*VSPShiftManipulator*) shifts the start time with a uniformly distributed random number.

To enhance the quality of optimisation results, permutation encoding for the VSP solutions is introduced. The encoding is based on the Alba encoding [2] for a vehicle routing problem. A chromosome is of the permutation type and contains  $m+n$  genes, where  $n$  is a number of vehicles and  $m$  is equal to a number of trips in the problem. Genes that have values less or equal to  $m$  encode a trip number and values greater than  $m$  encode delimiters or vehicle designators and define a vehicle number for the next sequence of trips.

The logic of the simulation model is that if no time window constraints are defined for the first trip, it starts at midnight; otherwise it starts at time to match the first customer's window. The next trip starts immediately after the previous one, unless its start time should be delayed to satisfy time windows of customers in the route of this trip. No times are encoded, and hence, the potential solution has no directly encoded idle time, and its trips cannot overlap. Due to the high universality of this encoding, different permutation manipulation operators can be applied in the search of the optimal solution.

A grid of landscape analysis experiments is created to compare values between different landscapes. First, comparison of different mutation operators is performed. Second, comparison between existing and proposed encodings is done. Results of comprehensive analysis experiments are described in [6]. Particularly found, that in a random walk, values of autocorrelation function are slightly lower for the replacement operator. In the up-down walk the situation is the opposite: replacement mutation has higher correlation than shift mutation, but the three artificial problems are different to the others (see Fig. 15; black dots are for *replacement* and green for *shift* mutator). Moreover, the value of the autocorrelation function in random and up-down walks is lower for the permutation encoding, which means that landscapes of this encoding should be more rugged.

**Fig. 15** Autocorrelation in up-down walks



### 5.4.3 Optimisation with HeuristicLab

A number of VSP optimisation experiments with the algorithms implemented in the HeuristicLab framework are performed. For the integer encoding both Evolution Strategy (ES) and Simulated Annealing (SA) algorithms are experimentally found to be fast and highly successful, but the ES is able to find solutions with better quality [6]. A genetic algorithm is able to find even better solutions, but with larger numbers of evaluations than evolution strategy. Permutation encoding is more effective in optimisation of the VSP, than integer encoding, as algorithms are able to find good solutions in less time. Moreover, almost all idle times are eliminated, and trip overlapping is also avoided. The evolution strategy is more effective than a genetic algorithm also for the permutation encoding. Even though the search space for this type of encoding is more complex and rugged, nevertheless, due to its smaller size the search of the globally optimal solution becomes more effective.

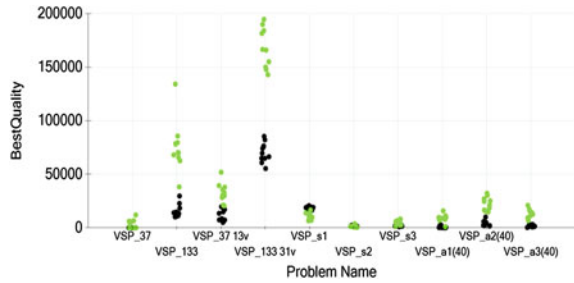
The best results found by different algorithms and for different encodings are presented in Table 4. Each algorithm for a specific problem has been run 10 times and the mean fitness of the best solution in each run has been calculated. A GA involved the population size of 100 chromosomes, tournament selection, 5% mutation rate and termination criterion defined by 500 generations. ES (20 + 100) strategy has been applied with maximum 1000 generations. Simulated annealing has been run with 3000 iterations and 10 evaluations in each iteration. Each run of GA or ES on a computer with 4-core CPU required from 8 to 10s, and about 3 s for simulated annealing.

Additional detailed optimisation experiments with similar instances and parameters show that there are relations between values of fitness landscape analysis measures and an ability of the optimisation algorithm to find the best solution. The genetic algorithm with population size of 100 individuals and termination condition of 500 generations is applied in one series, and the evolution strategy (20 + 100) with

**Table 4** Mean fitness values of the best solutions found by different algorithms

Problem	Genetic algorithm		Evolution strategy		Simulated annealing
	Integer	Permutation	Integer	Perm.	Integer
VSP_37	13 144.6	2 481	39.6	0	10 414.4
VSP_133	111 196.1	132 475	18 279.2	0	112 539.1
VSP_37(13)	47 313.1	26 367	15 860.1	180	67 904.6
VSP_133(31)	199 750.8	313 750	75 233.6	2 887	230 752.4
VSP_s1	33 759.3	3 600	10 346.1	0	12 741.2
VSP_s2	4 082.1	0	757.5	0	2708.5
VSP_s3	5 666.2	5 382	984.6	0	14 274.8
VSP_a1	12 446.2	6 715	347.7	0	26 530.7
VSP_a2	30 575.8	31 880	6377	0	52 755.6
VSP_a3	10 927.2	11 156	1 257.6	0	36 462.7

**Fig. 16** Quality of best found solutions with ES



crossover and 1000 generations in the second series of experiments. For the problem instances, which are better solved with the shift operator by GA, the autocorrelation for this operator also is higher (Figs. 15 and 16; black dots are for *replacement* and green for *shift* mutator). The same dependency is found for the ES algorithm.

Finally, it is concluded, that optimisation using the ES is the best choice for the solution of a vehicle scheduling problem with time windows. In case of the integer vector encoding is applied, selection of an appropriate mutation operator is based on the measures of the FLA, i.e. an operator, which has the highest autocorrelation value in the up-down walk should be selected.

## 6 Vehicle Routing

Often the delivery planning at the operational level also requires optimisation of vehicle routes. The section describes optimisation approach for the vehicle routing with time windows. First, the problem statement is given, and then an optimisation algorithm and its experimental adjustment are discussed.

### 6.1 Problem Statement

The classical statement of the vehicle routing problem with time windows (VRPTW) [7] is used further. Input data contains a set  $V$  of vehicles, a set  $C$  of customers and data about their geographical locations in the form of a directed graph  $G$ . The graph consists of  $|C| + 2$  vertices, whereby the customers are denoted as  $1, 2, \dots, n$  and the depot (in this case DC) is represented by vertices 0 and  $n + 1$ . A set of vertices of  $G$  is denoted as  $N$ , while a set  $A$  of arcs represents connections between customers and between the depot and customers. For each arc  $(i, j)$ , where  $i \neq j$ , a distance  $c_{ij}$  and a travel time  $t_{ij}$  are defined. Each customer  $i$  has demand  $d_i$  and should be served by one vehicle  $k$  with capacity  $q_k$  and only once within a planning horizon. For each customer, time window  $[a_i, b_i]$  when it has to be served is defined. Vehicles routes start and end in DC.



The VRPTW model contains two sets of *decision variables*, namely:  $x$  and  $s$ , which are defined as flow and time variables, correspondingly. Variable  $x_{ijk}$  for each arc  $(i, j)$ , where  $i \neq j$ ,  $i \neq n + 1$ ,  $j \neq 0$ , and each vehicle  $k$  is defined as follows:  $x_{ijk} = 1$  if vehicle  $k$  drives from vertex  $i$  to vertex  $j$ , and  $x_{ijk} = 0$ , otherwise. The decision variable  $s_{ik}$  for each vertex  $i$  and each vehicle  $k$  denotes the time, when vehicle  $k$  starts to service customer  $i$ .

Shortest routes for a fleet of homogenous vehicles with a limited capacity have to be found, i.e.:

$$\sum_{k \in V} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ijk} \rightarrow \min. \quad (14)$$

The main problem constraints are defined as follows [7]:

1. Each customer is visited only once;
2. No vehicle is overloaded:

$$\sum_{k \in V} d_i \sum_{j \in N} x_{ijk} \leq q; \quad \forall k \in V; \quad (15)$$

3. Each vehicle leaves depot 0, leaves a customer after its serving and finally arrives at the depot  $n + 1$ ;
4. Vehicle  $k$  cannot arrive at customer  $j$  before time  $s_{ik} + t_{ij}$  if it is travelling from customer  $i$  to customer  $j$ ; and
5. Time windows are defined by:

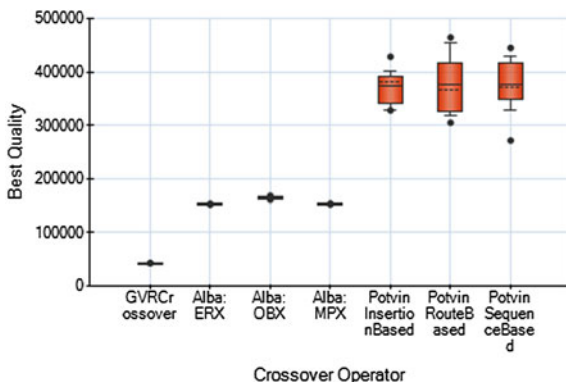
$$a_i \leq s_{ik} \leq b_i; \quad \forall i \in N, \forall k \in V. \quad (16)$$

## 6.2 Optimisation Algorithm

For vehicle routing, an *island genetic algorithm with offspring selection* (IOSGA) described in [31] is used. It presents a coarse-grained parallel genetic algorithm where population is divided into several islands in which GA works independently. Periodically, after a certain number of generations best solutions migrate between islands. The IOSGA is enhanced with an *offspring selection* to prevent a premature GA convergence. Offspring selection forces the algorithm to produce offspring solutions with better fitness than their parents [1].

For the considered problem instances, operators and parameters of the IOSGA are determined experimentally as follows: a proportional selector; 5 islands; 200 individuals in population; ring migration each 20 generations with 15 % rate: random individuals are replaced with the best ones from the neighbour island. The maximal selection pressure was set equal to 200 and the mutation rate equal to 5 %. Mutation operators provided in HeuristicLab framework were involved, and a GVR crossover [25] is selected in an experimental analysis below.

**Fig. 17** Performance of crossover operators in VRP



It is worth mentioning that GA is not considered as the strongest optimisation method for the VRP [4, 12], and more often the Tabu Search (TS) algorithm with constraint relaxation [8] is recommended as more efficient approach. However, genetic algorithms show good performance for routing problems and are highly robust and adjustable. Also, for the considered example instance, the IOSGA shows better results than TS as described below.

### 6.3 Route Optimisation Experiments

After an optimisation algorithm is selected, its components such as the crossover operator need to be justified. To select a crossover operator for the IOSGA a set of *route optimisation experiments* are performed (see Fig. 17). Here, the GVR crossover [25], edge recombination (ERX) and maximal preservative (MPX) crossovers for solutions using Alba encoding [2] are analysed. As the VRP is minimisation problem, better operators have lower best found quality values in Fig. 17. ERX application provided better results in terms of the total length of vehicles' routes, while preserving an available number of vehicles. However, the results obtained for Alba encoded solutions lost in terms of capacity constraints. In turn, application of GVR crossover provided solutions with an overflow of available vehicles, nevertheless the capacity constraints were satisfied in most cases. Finally, the GVR crossover operator which works with an unlimited number of vehicles, but provided the best feasible results in terms of keeping routes not overloaded was selected.

To determine, if selected optimisation algorithm has the best performance for the reviewed problem, additional route optimisations experiments are performed with the tabu search algorithm [8]. In these experiments TS was set up with following parameters: for the move generation and evaluation, the Potvin Shift Exhaustive Move operator; and the tabu tenure equal to 15 for problems with 109 customers and 13 for problems with 53 and 56 customers. The initial solution in each run was created using Push Forward Insertion heuristics with  $\alpha = 0.7$ ,  $\beta = 0.1$  and  $\gamma = 0.2$ . The termination criterion was defined by 1000 generations.

**Table 5** Routing results obtained with different algorithms

Problem	Stores	IOSGA		Tabu search	
		Distance	Vehicles	Distance	Vehicles
1	109	6331	44	6396	43
2	53	2561	18	2599	18
3	56	4346	26	4410	25

Results of IOSGA and TS optimisation experiments for the same problem instances are compared in Table 5. For each problem instance, the number of stores is defined. For each algorithm and problem instance, the shortest distance in km and a minimum required number of vehicles received from 10 optimisation experiments are given. A sample VRP instance of problem 2 solved by the IOSGA can be seen in Fig. 20.

The IOSGA provides better results in terms of distances and found solutions with less number of evaluations. The Tabu Search outperformed in minimisation of a number of required vehicles, which is supposed to be reduced in the next step of integrated methodology. Thus, the Tabu Search does not provide the significant improvement of the routing solution in the considered case.

## 7 Vehicle Scheduling for Routed Solution

The vehicle scheduling for routed solution is discussed. The approach is similar to one described in Sect. 5, but the difference is in input data and that scheduling in this case also aims at minimising a number of required vehicles for the same number of routes. The section presents the problem statement for routed solution is given and an optimisation algorithm to solve the problem. Then sequential application of vehicle routing and route scheduling algorithms is described.

### 7.1 Problem Statement

Obtained in Sect. 6 the vehicle routing solution provides minimal costs on vehicle driving distances. At the same time this solution can be improved further with a proper scheduling of routes between available vehicles to minimise also a number of required vehicles and related costs. In the classical VRPTW statement vehicle may perform only one route in the planning horizon. In practice, all routes often are shortened due to a limited capacity of vehicles. This may lead to ineffective solutions when a vehicle performs only one short route of a few hours long, while most of the day it may be idle. Thus, the *vehicle scheduling problem* is formulated *for the routed solution*. Here, routes are assumed to be independent from vehicles, while vehicles may perform a fair number of routes during the day. To solve the problem, methods developed in [19] are considered.

As far as the VRPTW solution is feasible for capacity and time window constraints, it can be further improved by combining and compacting routes. As a result, each vehicle can perform a sequence of predefined routes during the day and as a result, utilisation of vehicles may be significantly increased. A vehicle capacity is not involved in the statement as in the capacitated VRP all vehicles have the same capacity and no route of a feasible solution may exceed this value. Finally, *input data* is defined as follows: (1) ready time  $a_i$  for customer  $i$ ; (2) due time  $b_i$  for customer  $i$ ; (3) service time  $z_i$  for customer  $i$ ; (4) a list of routes  $R$  obtained in VRP solution, where each route defines a sequence of visited customers; (5) a set of transportation times (or vehicle moving times  $t_{ij}$  between route sequential points  $i$  and  $j$ ); (6) an estimated number of vehicles  $|V|$ .

Decision variables are similar to ones introduced in the VRPTW model, except that  $x_{ijk} = 1$  means that route  $j$  is the next route after  $i$  for vehicle  $k$ . Two soft constraints are introduced for each vehicle:

1. a number of times  $N_{ad}$  when a vehicle leaves a customer after due time:

$$N_{ad} = |\{i | s_{ik} + z_i > b_i; \forall i \in N; \forall k \in V\}|; \quad (17)$$

2. a number of times when vehicle busy time may exceed 24 h in a day:

$$N_{ot} = |\{k | (s_{(n+1)k} - s_{0k}) > 24 \cdot 60; \forall k \in V\}|. \quad (18)$$

Additional constraints are introduced to assure the model integrity and provisions of schedule simulations. Finally, a fitness function  $f$  defines a sum of vehicles idle times due to fitting deliveries to the time windows and a number of constraint violations multiplied by penalty values:

$$f = \sum_{k \in V} l_k + p_{ad} N_{ad} + p_{ot} N_{ot} \rightarrow \min, \quad (19)$$

$$l_k = \sum_{i \in R} \sum_{j \in R} \max(a_i - s_{ik}, 0) x_{ijk}; \forall k \in V, \quad (20)$$

where  $l_k$  is the total idle time of vehicle  $k$ ;  $V$  defines a set of available vehicles;  $p_{ad}$  and  $p_{ot}$  are penalty values or coefficients for late deliveries and vehicle overtimes, correspondingly, and  $p_{ad}$ ,  $p_{ot}$  are assumed to be significantly greater than 1.

## 7.2 Optimisation Algorithm

For vehicle scheduling, a problem plug-in in HeuristicLab optimisation framework is implemented by maintaining the logic of the above described vehicle schedule simulation model. Here, a fitness evaluator simulates a vehicle schedule candidate

6	4	10	3	8	7	12	1	11	5	2	9
---	---	----	---	---	---	----	---	----	---	---	---

**Fig. 18** A sample chromosome of permutation encoding for vehicle scheduling

and identifies time windows mismatches as well as evaluates vehicle idle and busy times. As a result, plug-in is used to calculate a fitness of a candidate solution by formula (19).

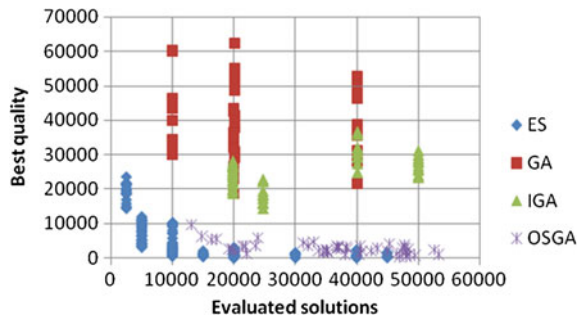
The chromosome which represents the schedule candidate is encoded as a permutation of integer numbers. Integers larger than the number of routes encode gaps in the chromosome when a new sequence of vehicle routes starts. Other integers define corresponding routes in sequences. The encoding used is similar to that described in Sect. 5.4. A sample chromosome for an instance with a number of vehicles  $n = 3$  and a number of routes  $m = 9$  is shown in Fig. 18. Here, the first vehicle marked as ‘10’ performs the sequence of routes: 3, 8 and 7; the second one the sequence of five routes: 5-2-9-6-4 and the third vehicle will perform only one route, marked as ‘1’.

The application of permutation-based encoding allows easy use of different recombination and mutation operators. For schedule optimisation, an Evolution Strategies (ES) algorithm implemented in HeuristicLab [32] is selected as it has shown high efficiency in vehicle schedule optimisation [6].

### 7.3 Schedule Optimisation Experiments

Various series of *schedule optimisation experiments* are performed to compare scheduling results obtained by different metaheuristic optimisation algorithms. The following algorithms are examined: ES ( $\lambda, \mu$ ) and ( $\lambda + \mu$ ) algorithms, GA, island genetic algorithm with 5 islands (IGA), and offspring selection genetic algorithm (OSGA) [1]. Maximal preservative crossover and insertion manipulator are applied in all algorithms. Numbers of solution evaluations performed to obtain candidates with equal fitness values are compared on hard instances, with a low number of vehicles and short time windows. The results of optimisation experiments for a single instance are shown in Fig. 19. As considered problem is a minimisation problem, best solutions have lower fitness and are in the bottom of the chart.

**Fig. 19** Productivity of optimisation algorithms for VSPTW



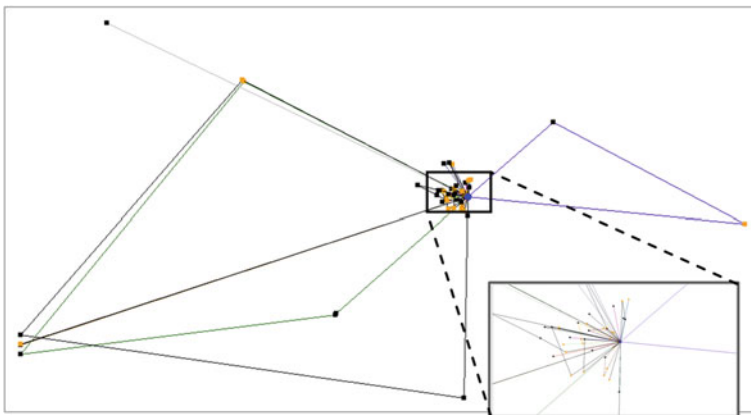
The experimental results show that the same instance is solved with ES and OSGA algorithms in shorter time, while GA without modifications demonstrated the worst result. The ES algorithm is selected as the most suitable having an ability to search for global solutions with fewer evaluations.

## 7.4 Example

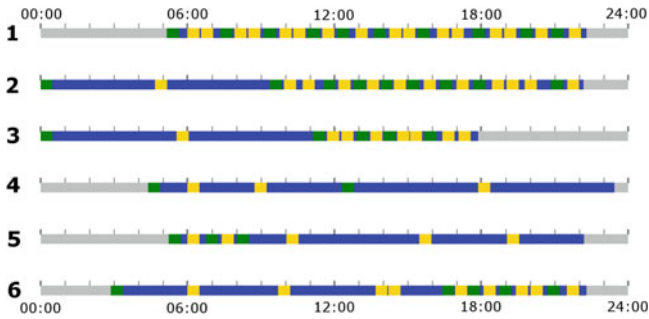
The following is an example of vehicle routes and schedule optimisation for a daily plan and specific demand data defined for 53 stores. Time windows for most stores are fixed from 6:00 a.m. to 9:00 p.m. But some stores can be served in any time. Application of the IOSGA in the VRP has given 34 routes (see Fig. 20) in the best solution. Due to a limited capacity of vehicles, most routes in the solution are very short, including one or two customers. As most of stores are located close to the DC (see inset in Fig. 20), due to small vehicle moving times, the trip length for many routes is relatively short. But, in case of large time windows and a long planning horizon, these routes can be combined.

Finally, evolution strategies (20 + 100) are applied for the routed solution. In schedule optimisation experiments, the maximal preservative crossover and insertion mutation operators were applied, and the termination condition was defined by 1000 generations. In experiments it is obtained that the problem instance that required 34 vehicles for deliveries (Fig. 20) has globally optimal solutions with all constraints satisfied when a number of vehicles is equal to 6.

The corresponding schedule Gantt chart for a planning horizon of 24 h is demonstrated in Fig. 21. Here, green lines define loading times in DC and the beginning of routes from the VRP solution, while blue and yellow lines depict transportation times and unloading times at stores, correspondingly. For example, the 4th vehicle in



**Fig. 20** VRP solution of case instance



**Fig. 21** VSP solution of case instance

Fig. 21 has to perform two very long trips that go to the stores in bottom left corner of Fig. 20. At same time the vehicle on first timeline is scheduled to perform 10 very short trips to the stores located near DC and having large time windows.

Similar experiments performed for multiple problem instances of the business case shows that vehicle scheduling applied for the routing solution allows reducing the number of vehicles required for daily deliveries.

To assess the quality of this approach the benchmark instances [28] are used. First, vehicle routes are found using the IOSGA, and then vehicles are scheduled with the ES. The experimental results show that these benchmark problems are designed to fit each route for each vehicle, so that subsequent scheduling does not provide any significant enhancement. When capacities of vehicles in benchmark instances are decreased twice to shorten potential vehicle routes and making them similar to those specified in the business case, the results of experiments on the modified benchmark problems show the benefit of the proposed approach for a number of instances. For example instance C102 requires only 16 vehicles instead of 21 to fulfil the product delivery plan from DC to a network of stores.

## 8 Conclusions

Modern delivery planning in large distribution networks with various constraining factors requires application of a number of methods to minimise delivery costs and cope with stochastic demand. Methods described in this chapter, such as cluster analysis, simulation, optimisation and fitness landscape analysis—are combined together into an integrated methodology to increase their application efficiencies and to reduce the computational requirements. Most of these methods are heuristic and metaheuristic and thus do not ensure obtaining globally optimal solutions, nonetheless they provide very good solutions, which are enough in most business cases, in less computational time comparing with traditional optimisation techniques.

The proposed integrated approach to product delivery tactical planning and scheduling allows identifying typical dynamic demand patterns and corresponding

product delivery tactical plans as well as finding the optimal parameters of product delivery schedules. Application of cluster analysis and classification algorithms to historic dynamic demand data allows identifying typical weekly demand patterns and developing base tactical delivery plans for groups of weeks with similar demand dynamics. Both k-means and NBTree methods are simple in implementation and implemented in a number of existing data mining tools. Store grouping allows determination of groups of stores, which have nearby location, considering the total demand of group. Application of multi-objective optimisation algorithm allows getting a set of non-dominating solutions and provides an opportunity to choose solution balancing between geographical and demand objectives. Both weekly demand pattern recognition and store grouping allow developing the base tactical delivery plan.

Two types of schedule optimisation solutions are provided. The first solution is designed, which has predefined vehicle routes and is simulation-based that allows dealing with stochastic factors of the deliveries, e.g. stochastic moving times. Fitness landscape analysis methods are presented and shortly described for tuning and adjustment of an optimisation algorithm. These methods if necessary could be also used for the second solution which allows optimising both vehicle routes and schedules when routes are not predefined. Application of advanced genetic algorithms or Tabu search methods allows obtaining in a fairly short time vehicle routes, which minimise transportation costs under constraints such as delivery time windows. Supplementing vehicle scheduling methods allows improving this routed solution by minimising a number of required vehicles and related costs. Both routing and scheduling are performed in same open source optimisation framework. Experimental results show that proposed two-stage routing and scheduling methodology provides good solutions in cases when vehicle routes are constrained with a small capacity. Joint sequential application of all presented methods allows obtaining cost-effective solutions in large store network delivery planning. In both solutions the parameters of the optimisation algorithms should be tuned for a specific case.

The proposed integrated approach to product delivery tactical planning and scheduling allows reducing the effect of product demand variation on the delivery planning process and avoids numerous time-consuming adjustments of the delivery tactical plans. Also, identifying demand pattern and an appropriate delivery plan ensure more qualitative solutions of the schedule optimisation task and cut down its computational costs.

## References

1. Affenzeller, M., Winkler, S., Wagner, S., Beham, A. (2009) *Genetic Algorithms and Genetic Programming: Modern Concepts and Practical Applications*. Chapman & Hall/CRC.
2. Alba, E. and Dorronsoro, B. (2004) 'Solving the vehicle routing problem by using cellular genetic algorithms' in *Proceedings of EvoCOP 2004*, April 5–7, Coimbra, Portugal, pp. 11–20.



3. AnyLogic (2014). Multimethod Simulation Software and Solutions. [ONLINE] Available at: <http://www.anylogic.com/>. (Accessed 01 August 2014).
4. Baker, B.M. and Ayechev, M. A. (2003) 'A genetic algorithm for the vehicle routing problem' in *Computers and Operations Research*, Vol. 30, pp. 787–800.
5. Bolshakov, V. (2013) *Simulation-based Fitness Landscape Analysis and Optimisation of Complex Systems*. PhD Thesis, Riga Technical University, Latvia.
6. Bolshakov, V., Pitzer, E., and Affenzeller, M. (2011) 'Fitness Landscape Analysis of Simulation Optimisation Problems with HeuristicLab' in *Proc. of 2011 UKSim 5th European Symposium on Computer Modeling and Simulation*, November 16–18, 2011, Madrid, Spain, pp. 107–112.
7. Cordeau, F., Desaulniers, G., Desrosiers, J., Solomon, M.M. and Soumis, F. (2001a) 'VRP with Time Windows' in Toth P. and Vigo D. (Eds.), *The vehicle routing problem*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pp. 157–193.
8. Cordeau, J.F., Laporte, G., Mercier, A. (2001b) 'A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows' in *The Journal of the Operational Research Society*, Vol. 52, No. 8, pp. 928–936.
9. Davies, D.L., Bouldin, D.W. (1979) 'A cluster separation measure' in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2), pp. 224–227.
10. Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002) 'A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II', *IEEE Transactions on Evolutionary Computation*, 6(2), pp. 182–197.
11. Eliiyi D.T., Ornek A., Karakutuk S.S. (2008) 'A vehicle scheduling problem with fixed trips and time limitations' in *International Journal of Production Economics*, Vol. 117, No.1., pp. 150–161.
12. Gendreau, M., Laporte, G., and Potvin, J.Y. (2001) 'Metaheuristics for the capacitated VRP' in Toth P. and Vigo D. (Eds.), *The vehicle routing problem*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, pp. 129–154.
13. Gwiazda, T.D. (2006) *Genetic algorithms reference Vol. I. Crossover for single-objective numerical optimization problems* [online]. TOMASZGWIAZDA E-BOOKS. [http://www.tomaszgwiazda.com/ebook\\_a.htm](http://www.tomaszgwiazda.com/ebook_a.htm). (Accessed 10 January 2013).
14. Kaufman, L. and Rousseeuw, P. J. (1990) *Finding Groups in Data: An Introduction to Cluster Analysis*. Hoboken, NJ: John Wiley & Sons, Inc.
15. MacQueen, J. B. (1967) 'Some Methods for Classification and Analysis of MultiVariate Observations' in *Proc. of the 5th Berkeley Symposium on Math. Statistics and Probability*, Vol. 1, pp. 281–297.
16. Michalewicz, Z. (1999) *Genetic Algorithms + Data Structures = Evolution Programs*. Third, Revised and Extended Edition. Springer-Verlag Berlin Heidelberg.
17. Merkurjeva, G. (2012) 'Integrated Delivery Planning and Scheduling Built on Cluster Analysis and Simulation Optimisation' in *Proceedings of the 26th European Conference on Modelling and Simulation*, pp. 164–168.
18. Merkurjeva, G. and Bolshakov, V. (2012a) 'Simulation-based Fitness Landscape Analysis and Optimisation for Vehicle Scheduling Problem' in Moreno D.R., et al., (Eds.), *Computer Aided Systems Theory – EUROCAST 2011, Part I, LNCS 6927*, pp. 280–286.
19. Merkurjeva, G. and Bolshakov, V. (2012b) 'Simulation optimisation and monitoring in tactical and operational planning of deliveries' in *Proceedings of the 24th European Modeling and Simulation Symposium, EMSS 2012*, pp. 226–231.
20. Merkurjeva, G., Bolshakov, V. and Kornevs, M. (2011) 'An Integrated Approach to Product Delivery Planning and Scheduling', *Scientific Journal of Riga Technical University, Computer Science, Information Technology and Management Science Ser. 5*, Vol. 49, pp. 97–103.
21. Merkurjeva G. and Bolshakovs V. (2010a) 'Simulation-based vehicle scheduling with time windows' in *Proc. of First International Conference on Computer Modelling and Simulation*, pp. 134–139.
22. Merkurjeva, G. and Bolshakovs, V. (2010b) 'Vehicle schedule simulation with AnyLogic' in *Proc. of 12th Intl. Conf. on Computer Modelling and Simulation*, pp. 169–174.

23. Merkurjeva, G., Merkurjev, Y. and Bolshakov, V. (2010) 'Simulation-based fitness landscape analysis for vehicle scheduling problem' in Proc. of the 7th EUROSIM Congress on Modelling and Simulation, September 6–10, 2010, Prague, Czech Republic.
24. Nagamochi, H., Ohinishi, T. (2008) 'Approximating a vehicle scheduling problem with time windows and handling times' in Theoretical Computer Science, Vol. 393, Is. 1–3, pp. 133–146.
25. Pereira, F.B., Tavares, J., Machado, P. and Costa E. (2002) 'GVR: A New Genetic Representation for the Vehicle Routing Problem' in Proceedings of the 13th Irish International Conference on Artificial Intelligence and Cognitive Science (AICS '02), September 12–13, Limerick, Ireland, pp. 95–102.
26. Pitzer, E., Affenzeller, M. (2012) 'A Comprehensive Survey on Fitness Landscape Analysis' in Recent Advances in Intelligent Engineering Systems: Springer, pp. 161–191.
27. Seber, G.A.F. (1984) Multivariate Observations, John Wiley & Sons, Inc., Hoboken, NJ.
28. Solomon, M.M. (2005) VRPTW Benchmark Problems. [online] <http://w.cba.neu.edu/~msolomon/problems.htm> (Accessed 26 September 2013).
29. Stadler, P.F. (2002) 'Fitness Landscapes' in Lassig M. and Valleriani A. (Eds.), Biological Evolution and Statistical Physics, Springer, pp. 183–204.
30. Vassilev, V.K., Fogarty, T.C., Miller, J.F. (2000). 'Information Characteristics and the Structure of Landscapes' in Evolutionary Computation, 8(1), pp. 31–60.
31. Vonolfen, S., Affenzeller, M., Beham, A., Wagner, S. (2011) 'Solving large-scale vehicle routing problem instances using an island-model offspring selection genetic algorithm', in Proc. of 3rd IEEE International Symposium on Logistics and Industrial Informatics (LINDI), 2011, August 25–27, 2011, Budapest, Hungary, pp. 27–31.
32. Wagner, S. (2009) Heuristic Optimization Software Systems - Modeling of Heuristic Optimization Algorithms in the HeuristicLab Software Environment. PhD Thesis, Institute for Formal Models and Verification, Johannes Kepler University Linz, Austria.
33. Weinberger, E. (1990) 'Correlated and Uncorrelated Fitness Landscapes and How to Tell the Difference' in Biological Cybernetics, 63(5), pp. 325–336.