Optimizing Door Assignment in LTL-Terminals by Evolutionary Multiobjective Algorithms

Thomas Bartz-Beielstein, Annette Chmielewski, Michael Janas, Boris Naujoks and Robert Scheffermann

Abstract— **In less-than-truckload terminals arriving trucks have to be allocated to a gate and to a time slot for unloading. The allocation to a specific gate results in different transportation volumes for the forklift trucks inside of the terminal, depending on the destinations of the truck's loads. While minimizing these transports the time for trucks waiting to be ordered to a gate also has to be minimized. For the first time this problem has been tackled as a 2-objective optimization problem and was solved by an (1+1)-evolution strategy. We developed a model which is derived from real freight forwarder's data and represents a small company's terminal on an average workday.**

I. Introduction

In logistical terminals it is to be decided at which gate and at what time a truck should be unloaded. The goods have different destinations inside of the terminal and the distance from the gates to these destinations is different for each gate as illustrated in Fig. 1. It is important to minimize the waiting time for trucks and keep the transportation volume inside of the terminal low. Goods with a total weight under three tonnes, which are often placed on a pallet for further transport activities, are called *less-than-truckload* (LTL) consignments.

Bermudez & Cole (2001) were one of the first tackling this kind of problem. They used a genetic algorithm to min In their model they assume that a single gate does serve only a single truck, which means just the allocation of trucks to gates is considered and no time constraints exist. Another approach by Stickel & Furmans (2005) on crossdocking terminals concentrates on the time-scheduling aspect and also takes the vehicle routing for inbound and outbound routes into account. The associated mathematical model is very complex. It was possible to solve the *mixed integer linear program* (MILP) with CPlex for very small problem instances. The processes inside of terminals were solved by Li & Rodrigues (2004) using an hybrid evolutionary algorithm.

Chmielewski & Clausen (2005, 2006) developed an enhanced mathematical model for optimizing less-than-truckload terminals that is based on a time discrete multicommodity flow and supplemented by necessary side constraints. The resulting MILP was programmed with the optimization solver CPlex 4.1 and different test scenarios were applied to the Branch-and-Cut algorithm implemented in CPlex. On the one hand the test scenarios show that for small and middle sized problem instances good solutions can still be found. But on the other hand—especially for the case of medium sized problems—finding optimal solutions takes up to 30 minutes or more. Obviously, this time span is prohibitive for on-line optimization problems. So, exact solution methods are not relevant for the dynamic allocation of trucks to gates in logistical terminals. Also the two objectives have so far only been considered by including a penalty for late docking into the monocriterial objective function.

In our paper we present an approach that allows to allocate multiple trucks to the same gate on different timeslots, so extending the model Bermudez & Cole (2001) used for their genetic algorithm. Compared to Stickel & Furmans (2005) we were able to find good solutions for much larger problem instances, but the underlying model of crossdocking-terminals is different in many aspects from LTL-terminals. Therefore, the results are not directly comparable. The underlying model is similar to the one used by Chmielewski & Clausen (2005), but in our new approach the problem was tackled as a multiobjective problem for the first time.

We solved the two criteria decision problem of minimizing the transportation volume inside of LTL terminals and the waiting time for trucks between arrival at the terminal and being assigned to a gate. This problem will be referred to as the *LTL-problem* in the remainder of this article. The next section will give a more detailed definition of the corresponding model, which is very similar to the model used in Chmielewski & Clausen (2005). Section 3 will introduce the algorithm we used to solve the problem: a 1+1 evolution strategy. The experiments are then described in Sect. 4. Next the experiment's results are discussed in Sect. 5 and different variants are being compared. Finally we give a summary in Sect. 6.

II. Problem

The transport of LTL goods within a country or a region is organized via a transportation network. The transportation request of one customer (normally between 1 and 10 pallets) usually does not suffice to fill the load area of a whole truck (up to 33 pallets). The network structure allows companies to use bundling effects by consolidating all consignments with the same long distance destination on one truck. Therefore, a transportation network consists of several logistical sites, which will be referred to as freight forwarding terminals.

Thomas Bartz-Beielstein, Michael Janas, and Boris Naujoks are with the Chair of Algorithm Engineering and Systems Analysis, University of Dortmund, 44221 Dortmund, Germany (email: {thomas.bartz-beielstein, michael.janas, boris.naujoks}@udo.edu)

Annette Chmielewski and Robert Scheffermann are with the Chair for Transportation Science, University of Dortmund, 44221 Dortmund, Germany (email: {scheffermann, chmielewski}@uni-dortmund.de)

These terminals are spread uniformly over the region or country and are interlinked by line haul traffic.

The core element of a terminal is the transfer station which is a building with several gates (depending on the size of the terminal between 20 and 100 gates). The gates can be separated in inbound and outbound gates. Inbound gates are used for discharging trucks and outbound gates for charging. Some gates, called multi-functional gates, can be used for both logistical functions (inbound and outbound).

A local area is assigned to each terminal of a transportation network. The daily transportation requests of all customers located within this area are collected in local tours during the day by small trucks. In the afternoon, these trucks arrive according to a certain timetable with earliest arrival time at the freight forwarding terminal. Some trucks will be needed for further tours or transport services and therefore have to leave the terminal at a certain point of time or at least as soon as possible. Within its attendance time, a truck is allocated to an inbound gate to be discharged. The different goods from that truck are consolidated according to their long distance destinations. Afterwards, they are transported by forklift trucks within the building to those outbound gates where the trucks for the different long distances are loaded.

The long distance trucks leave the site in the late evening according to a certain timetable that guarantees their over night arrival on time at a partner terminal. Recapitulating, a terminal has two main operating periods: the inbound of local collection tours with subsequent outbound of long distance (12am - 8pm) and the inbound of long distance trucks with the outbound of small trucks for delivering goods in the local area (12pm - 9am).

The assignment of trucks to gates and time slots is also known as yard management. It is the interface between tours that are conducted on the road network and the processes and operations for the transshipment of goods within the terminal building. It effects the amount of the resulting distances for the transshipment of all load units between the inbound gates and the outbound gates. Therefore, one objective of the planning is to find an optimal allocation that leads to minimal total distances and a minimal number of resources needed in operations. A second objective is the minimization of waiting times. Trucks should be allocated to a gate as soon as possible after their arrival at the terminal. Each truck has an individual time table indicating the earliest arrival time and the latest departure time from the terminal.

The planner has to reserve a time slot within this period of time that is long enough for discharging and charging the booked number of load units. If a truck is not allocated right after its arrival, the driver has to wait in a parking zone until he gets further information. Therefore, minimizing waiting times leads to less crowded yards. In addition, trucks should be charged and discharged as soon as possible to reserve dock gates for time critical or very late trucks.

Figure 1 shows how the transport volume depends on both the assigned gate of the inbound tour and the number of palettes to each outbound tour/gate. In this example the

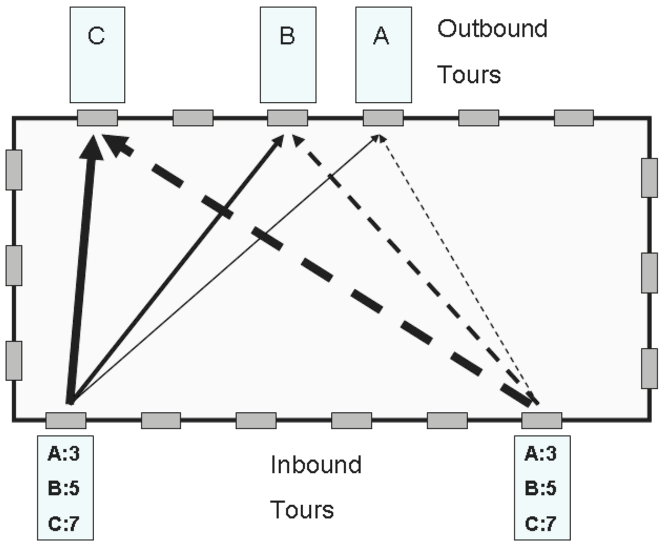

Fig. 1. Visualization of the decision to make: The transportation volume depends on the assigned gate and number of palettes to be transported to the outbound Tours. In this example the inbound tour contains 3 pallets for outbound tour A, 5 for tour B and 7 for tour C. If the inbound truck is assigned to the left gate then the average transportation volume is low as most of the pallets have to be transported to a close destination. If the truck is allocated on the right gate then the destination C, to which most pallets have to be transported, is far away so the resulting transportation volume is higher. One of the objectives of the fitness function described in section III-D is the total transportation volume for all trucks

left gate results in less transportation volume than the right gate. As the time available on each gate is limited not every inbound tour may be assigned to its optimal gate. Also when the inbound tour arrives at the terminal its optimal gate may be blocked by another inbound tour - the decision to make is to either wait until the optimal gate is freed or pick a sub-optimal inbound gate. Depending on these decisions the two conflicting objectives of minimizing total transport volume inside of the terminal and minimizing the total waiting time for inbound tours have to tackled. The algorithms may also decide to which gates the outbound tours have to be located to, so the inbound tours can be assigned best.

In our model we consider a small terminal of rectangular shape with 25 gates. 50 trucks have to be assigned to these gates, 10 of the trucks being long distance trucks (outbound). The position of outbound gates, inbound gates and multi-functional gates are part of the solution the algorithms create and just the number of gates is defined beforehand. Each truck has to be allocated to a gate in an individual time window between 3pm and 6pm and each truck carries 10 palettes in average for up to 5 different outbound gates. The model used for our algorithms is derived from real freight forwarder's data and represents a small company's terminal on an average workday.

III. Algorithm

A. Representation of solutions

A candidate solution I implements an array of lists. Consider the array of lists for the ith solution $I_i =$

$[G_{i1}, G_{i2}, \ldots, G_{im}]$, where m denotes the number of gates. Each list $G_{ij} = [T_{ij1}, \ldots, T_{ijk}]$ represents one gate. These list entries represent tours. Each tour T_{ijl} consists of an array with four integer values:

$$T_{ijl} = [\text{tourNumber}, \text{gateNumber}, \text{startTime}, \text{endTime}].$$

To accelerate the function evaluation, two additional arrays to store times and gates were implemented. Within the framework of evolutionary algorithms, candidate solutions are also called individuals.

B. *Description of the Simple Heuristic*

The existing heuristical approach mimics the decisions of a human planner and follows some simple rules and classification figures.

First the tours are sorted by a combination of priority and expected difficulty of being assigned to a gate. The tour's priority depends on an assesment based on the user's experience and mix/type of loads on the truck, while the difficulty depends on the size of the time window and time of arrival. The weight of each figure can be defined by the user to find a sorted list of tours matching the individual requirements.

Following this sorted lists each tour is then assigned to a gate at the earliest possible time. This is done by calculating the expected waiting time for the truck at each gate and the resulting transportation volume for the loads on the truck from that gate to all the load's destinations inside of the terminal. These two criterias are used to assign the tour to a gate and the user may define the weights for combining the two objectives to find a solution.

By changing a tour's priority or varying the heuristic's weights different solutions can be found focusing either on optimizing the waiting times for trucks and/or transportation volume inside of the terminal.

It has to be pointed out, that the heuristic is unable to optimize the allocation of outbound tours—they have to be assigned to gates manually beforehand. This is similar to the real world planning task when in LTL-terminals the allocation of outbound tours usually is historically grown and not adjusted regulary. Another drawback is, that the heuristic is not built to find a set of Pareto-optimal solutions. It has to be decided beforehand, if waiting times or transportation volume has to be considered or how these objectives should be weighted. So a single solution can be found but there is no further knowledge about the possible solution space available.

For these reasons, we decided to develop an algorithm that can tackle the problem in a better way. A multiobjective approach not only gives a range of solutions so that the human planner is able to pick one that satisfies his needs—we also gather knowledge about the solutions possible to create.

A much more important issue is to develop an algorithm which is able to estimate the potential benefit when outbound tours can be allocated freely. As a result, the current allocation scheme and processes in LTL-terminals can be optimized.

C. *Description of the Evolution Strategy*

Evolutionary algorithms are well suited to satisfy our needs, as they can easily be implemented to find a set of Pareto-optimal solutions and work quite well even on very complex combinatorial optimization problems.

Therefore we have chosen a simple EA, the $(1+1)$-ES. Schwefel (1995) described this algorithm as "the minimal concept for an imitation of organic evolution." Let f denote a multi-objective function to be minimized. The rules of an $(1+1)$-ES for MCO can be described as shown in Algorithm 1.

As already mentioned before, there are two objectives arising from the problem: minimization of distances within the minimization of waiting times. Due to this multi-objective nature of the problem, it was reasonable to apply multi-objective optimization techniques. The decision to invoke evolutionary multi-objective optimization techniques is based on the needs of decision makers (planners) to have a set of alternative solutions at hand to derive a final decision. Here, the concept of Pareto dominance comes into play.

A solution one is said to dominate a solution two, iff all components of the fitness function f of solution one are not greater that the corresponding components of solution two and really smaller in at least one component. The set of non-dominated solutions is called the Pareto set of solutions while the corresponding pictures under function f are called the Pareto front.

The appreciated set of alternative solutions, a Pareto set, to allow an a posterior decision in multi-objective optimisation problems is offered by evolutionary optimisation techniques (Deb, 2001; Coello Coello et al., 2002). Several other techniques need an a priori choice of a ranking of objectives or the definition of weights to start the optimization.

Algorithm 1 $(1+1)$-ES

```
1:  $t = 1$                                   /* Initialize iteration counter */
2:  $I^{(t)} \leftarrow init()$                 /* Initialize candidate solution */
3:  $\mathcal{A}^{(t)} \leftarrow \emptyset$     /* Initialize archive */
4:  repeat
5:    $I^{(t)}_{New} \leftarrow$ mutate$(I^{(t)})$    /* Generate offspring */
6:    if $(\nexists I^{(t)} \in \mathcal{A}^{(t)} : f(I^{(t)}) \prec f(I^{(t)}_{New}))$ then
7:      $I^{(t+1)} \leftarrow I^{(t)}_{New}$
8:      $\mathcal{A}^{(t+1)} \leftarrow \{I^{(t)}_{New}\} \cup \mathcal{A}^{(t)}$    /* Update archive */
9:    else
10:     $I^{(t+1)} \leftarrow I^{(t)}$
11:   end if
12:   $t \leftarrow t + 1$
13: until stopping criterium fulfilled
```

D. *Fitness Function*

As already mentioned while describing the problem, two objective functions f_1, f_2 are considered for minimization.

Ignoring the resources needed for the operations inside the transfer stations, the first one describes the way of each pallet:

$$f_1(I_i) := \sum_{j=1}^{m} \sum_{l=1}^{k} \sum_{r=1}^{s} d(G_{ij}(P_{ijlr}), G_{id}(P_{ijlr})),$$

with P_{ijlr} being r-th pallet of tour T_{ijl} at gate G_{ij} with destination gate G_{id} $(d \in \{1, \ldots, m\})$. The function d describes the distance inside the transfer station from one gate to another. It could also invoke different kinds of resources for the operations, but we limited ourselves to distances here.

The second objective function displays the waiting time for each truck:

$$f_2(I_i) := \sum_{j=1}^{m} \sum_{l=1}^{k} t_w(T_{ijl}),$$

with function $t_w(T_{ijl})$ being the difference between the point of time the unloading of truck of the corresponding tour is started and the arrival time at the transfer station. This time is normally spent in some parking area. For reasons of simplicity, we neglected a detailed description of all constraints that can be derived from the problem description. Of course, all constraints are represented in our algorithm for the task.

E. Problem Specific Operators for the 1+1-ES

Search points are initialized as follows: *Long distance tours* ($\mathcal{T}_{\text{long}}$) are randomly assigned to *long distance gates* ($\mathcal{G}_{\text{long}}$). If all long distance gates are occupied, the remaining long distance tours are assigned to multifunctional gates. Short distance tours ($\mathcal{T}_{\text{short}}$) are assigned to the first available gate from the set of short distance and multifunctional gates. The initialization is restarted if a tour cannot be assigned to any gate.

The mutation operator chooses randomly a tour $T \in \{\mathcal{T}_{\text{long}} \cup \mathcal{T}_{\text{short}}\}$, which will be reassigned. Next, a gate $G' \in \mathcal{G}$ with feasible arrival time is selected randomly. Two mutation opererators have been implemented:

1) Random Mutation: redistribute tours T' that have been previously assigned to G' randomly to available gates.
2) Quick Mutation: assign T' to the first available gate.

In both cases, the mutation is repeated if a tour cannot be assigned.

F. Selection

As mentioned above, we utilized a simple $(1+1)$-ES selection scheme, but it has to deal with multiple objectives and therefore differs from the single-objective case, of course. The selection scheme implemented accepts the offspring individual to become the parent in the next generation, iff it is non-dominated by all individuals generated by the algorithms until now. The set of individuals generated within the optimization run and non-dominated by each other is called the *current Pareto front* PF$_{\text{cur}}$.

In contrast to the simple evolutionary multi-objective optimizer SEMO, our approach keeps the parent individual, if the offspring individual is not selected. SEMO chooses a new parent in each generation uniformly from PF$_{\text{cur}}$ (Laumanns, 2003). The current Pareto front is updated after each generation, individuals dominated by the new parent are removed.

IV. Experiments

An experimental design has to be specified before the experimental analysis can be started. Our experiments are based on the experimental methodology from Bartz-Beielstein (2006). A hypervolume can be used to judge the performance of algorithms for multiobjective optimization problems. To calculate the hypervolume value $\mathcal{S}(\text{PF}_{\text{cur}})$ the objective function values of each individual of the Pareto-front are considered. The hypervolume is the space covered by the solutions of the Pareto front calculated with respect to a chosen reference point x^{ref}:

$$\mathcal{S}(\text{PF}) = \Delta \left(\bigcup_{I \ inPF} \{x \in \mathbb{R}^n | f(I) \prec x \prec x^{ref}\} \right),$$

with Δ being the Lebesque measure of the hypercube spanned by the solutions from the Pareto-front and the reference point. For the two-dimensional case studied here, this can be simplified to:

$$\mathcal{S}(\text{PF}) = \Delta \left(\bigcup_{I \ in\text{PF}} [x_1^{ref} - f_1(I)] \times [x_2^{ref} - f_2(I)] \right).$$

As the door-assignment problem was introduced as a new problem class, no representative results (as for TSP instances) are available. To overcome this difficulty, we proceeded as follows: The problem was solved with several algorithms that used a similar budget, i.e., number of function evaluations. The upper 10% quantile of the function values from all results was chosen to characterize "good" algorithms. Run-length distributions (Fig. 2) as proposed by Hoos (1998) were used to determine an adequate number of function evaluations for the final comparisons.

They are reliable tools to avoid floor- and ceiling effects. These effects occur if problem instances that are chosen, which are too hard, or too easy, respectively, for the algorithms under consideration.

As can be seen from Fig. 2, 300,000 function evaluations are a good compromise to detect differences between algorithms and to enable a fair comparison.

One main research topic in evolutionary computation is the design of problem-specific evolutionary algorithms (Beielstein et al., 2003). The aim is to systematize the design of evolutionary algorithms for problems with nonstandard representations. Especially nonstandard, problem-specific representations and variation operators are of great importance. Therefore, it is an important step to develop and analyze mutation operators for the LTL-problem. Two mutation (variation) operators, which were introduced in Sect. III-E, are subject of our experimental analysis. Our comparison is

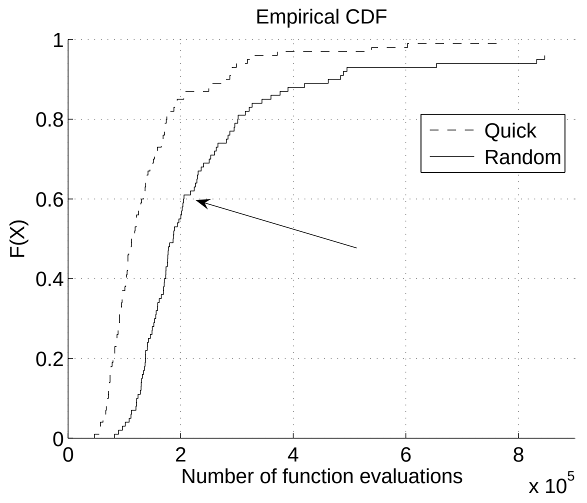

Fig. 2. Run-length distribution to determine the computational budget, i.e., the number of function evaluations for the comparisons. The graphs illustrate the RLD of the $(1+1)$-ES with quick mutation and with random mutation, respectively. Based on these distributions, 300,000 function evaluations were chosen

TABLE I

RESULT TABLE OF THE FUNCTION VALUES $(\times 1.0e+06)$ FROM $n = 50$ RUNS FOR THE LTL-PROBLEM. BETTER VALUES ARE PRINTED *boldface*

Design	Min	Mean	Median	Max	SD
Quick	7.6587	8.6733	8.7182	9.3124	0.2967
Random	**8.1642**	**8.8457**	**8.8562**	**9.4976**	**0.2380**

based on the LTL-problem with 300,000 function evaluations. Each run was repeated 50 times. Results from these runs are shown in Table I. Histograms (Fig. 3) visualize the numerical values from Table I. Outliers and variation of the function values can easily be compared.

V. ANALYSIS

Experiments were performed to tackle the following tasks:

- *Validation.* Is the simulation model correct?
- *Comparison.* Is an evolutionary approach better than a practical heuristic?
- *Operators.* Is it beneficial to implement problem-specific operators for the ES?

A. Validation

Our simulation model is more than just an abstraction developed by an analyst or theoretician working in isolation. It is based on representative data. Comparisons with similar studies and with expected results from real world data support the assumption that our model is valid. Results have been reported to practitioners who confirmed their validity. However, this is the first step in a very complex validation procedure. A detailed statistical comparison of output data from our model and real-world data (which may take a year or more) was not performed yet. Hence, we can state that the validation performed so far gave no evidence that the model is wrong.

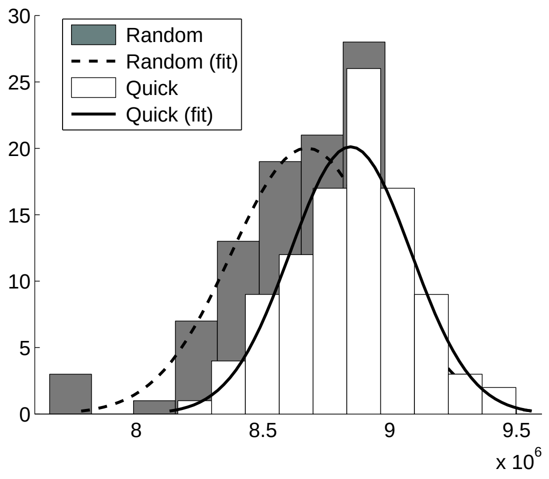

Fig. 3. Histograms comparing distributions of function values from runs with random mutation and quick mutation schemes as introduced in Sect. III-E. Larger values are better. Quick mutation outperforms random mutation

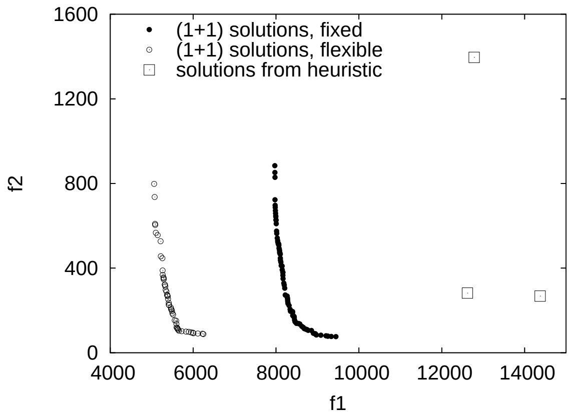

Fig. 4. Comparison of results from the heuristic described in the text and from two simple (1+1)-selection scheme. The presented Pareto fronts were received from two runs with the (1+1)-Algorithm while the three points were received with the heuristic featuring different weights. The left Pareto front was created with flexible outbound tours while the right Pareto front has the same fixed allocation of outbound tours to gates as the heuristic. It can be seen that allowing to reallocate outbound tours has a major impact on the solution and the size of the gap shows the potential of reallocating outbound tours to gates. The three solutions of the heuristic have been created with three different settings of how to weight the two objectives. It can be seen that the (1+1)-Algorithm is better than all of the three single results and that also more alternative solutions along the found Pareto front exist

B. Comparison

Results from the heuristic introduced in Sect. III-B are compared to results from the multi-objective $(1+1)$-ES. Figure 4 compares Pareto fronts after 300,000 function evaluations of the $(1+1)$-ES with three solutions from the heuristic.

It can be clearly seen, that the results from the heuristic are outperformed by the solutions from the $(1+1)$-ES by far. Nearly all points from the Pareto front dominate at least one of the solutions from the heuristic. Many points even dominate all three such solutions. Therefore, it can clearly be stated, that the $(1+1)$-ES works much better than

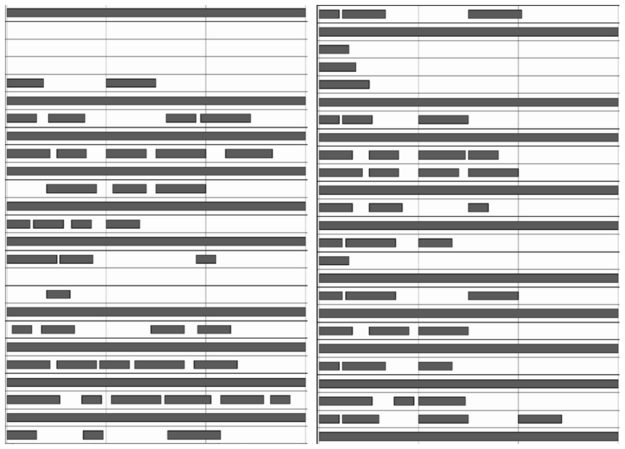

Fig. 5. Two individuals of the left Pareto front from Fig. 4. It can be seen that if waiting time for trucks is the main objective, then at least one truck has been assigned to each gate and trucks are docking to gates very early in general (right chart). On the other extreme if mainly the transportation volume inside of the terminal is considered then some of the gates are not used at all while some - probably generally good gates - are very crowded and some trucks are assigned to gates on a very late time slot (left chart)

the heuristic designed by experts on this special problem. Furthermore, the generation of the Pareto front with the $(1+1)$-ES required only a few seconds CPU time. Compare this value to the time required by the deterministic algorithm used in Chmielewski & Clausen (2005, 2006). Therefore we can state that the $(1+1)$-ES significantly outperforms state-of-the-art approaches with respect to solution quality and time to obtain this solution.

C. Operators

The quick mutation operator assigns tours to gates and considers the time slots, whereas the random mutation operator does not consider arrival times. Results presented in Table I and Fig. 3 clearly demonstrate the usefulness of problem specific knowledge for evolutionary algorithms. Furthermore, experiments with a multimembered evolution strategy which uses standard mutation and recombination operators were performed. The $(1+1)$-ES outperformed the multimembered-ES, too.

D. Interpreting solutions

A more detailed look at the solutions generated by the $(1+1)$-ES might provide some insight that is helpful for further improvements and can guide the development of enhanced variation operators or representations. From two solutions of the Pareto front, Gantt charts of the distribution of tours to gates are presented additionally in Fig. 5.

These Gantt charts represent one gate in each row and the blocks give a tour, that is assigned to the corresponding gate and point of time. Therefore, the Gantt charts also display the distribution of tours in time. Within all presented Gantt charts, a block allocating a whole row (a gate for the whole time) means the corresponding gate is allocated by a long distance tour. This implies, that the gate is an outbound or a multi-functional gate. In Fig. 5, two Gantt charts illustrating solutions from the Pareto front are detailed:

1) A Gantt chart for the solution from the upper left flank of the Pareto front, where the distances inside the transfer station (f_1) are minimized without caring too much about waiting times.
2) A Gantt chart for the solutions from the lower right flank of the Pareto front. Here, the focus lies on the the minimization of the waiting time (f_2) instead of the distances.

The presented Gantt charts show the expected appearance. In the one assigned to the solution minimizing f_1, some gates are not attended by any truck. This is the major difference to the Gantt chart assigned to the solution minimizing f_2 in the lower right flank of the figure. Here, a tour is assigned to every gate right in the beginning. This clearly links to the minimization of waiting time, that is focused on in this area of the Pareto front. In the other solution, some gates are unattended, meaning these are far away from the long distance trucks and the resulting distances for the pallets would be too long. Here, a certain time is accepted to be assigned to a gate nearer to the corresponding long distance trucks. This clearly indicates the focus on distance minimization in this area of the Pareto front.

Considering multi-functional gates, the different Gantt charts emphasize another advantage of the $(1+1)$-ES. In contrast to the heuristic, this approach is able to assign different gates to long distance trucks. This can be seen from the two Gantt charts, where different gates are occupied by these trucks. The heuristic needs an a priori decision which gates are assigned to long distance trucks.

VI. Summary and Outlook

We introduced a simulation model derived from real freight forwarder's data. It models a small company's terminal on an average workday. This model–which is used for LTL-terminals–differs in several aspects from breakbulk-terminals and crossdocking terminals. Hence, results are difficult to be compared. However, the proposed model is superior to existing models in the following sense: Bermudez & Cole (2001) did not take into account that multiple trucks may share the same gate. Stickel & Furmans (2005) did not include the waiting time for trucks—also the mix of loads on a truck and the number of possible destinations is much more complex in LTL-terminals. The model proposed by Chmielewski & Clausen (2006) is very similar, but the method is unable to solve larger problems efficiently and the problem is optimized for a single objective only.

We demonstrated that a $(1+1)$-ES can solve the two-objective problem. The $(1+1)$-ES outperformed an existing heuristic. Problem specific operators improve the performance of the $(1+1)$-ES. An evolution strategy, which did not incorporate domain knowledge, failed completely on this problem.

After the superiority of the $(1+1)$-ES over the simple heuristic could have been shown, the approach will be further investigated and compared to other techniques. The $(1+1)$-ES shares some properties of the SEMO algorithm. It suggests itself, that these approaches are to be compared on the current test problem as well as on other ones.

Moreover, the $(1+1)$-ES will be compared to the multi-membered $(\mu+\lambda)$-ES. For the selection in the current multi-objective test case, individuals will be ranked according to their dominance-rank in comparison to all other $\mu+\lambda-1$ individuals. If it has to be decided between different solutions with the same dominance-rank, this is done uniformly distributed at random. And, we did not apply the very efficient and effective *sequential parameter optimization* technique to improve the performance of the $(\mu+\lambda)$-ES, see Bartz-Beielstein (2006). This tuning procedure will be integrated into further analyses.

Furthermore, new approaches to solve smaller instances of the current problem mathematically are under development and will be investigated and compared to the approach at hand in the near future. A detailed comparison will be done to deduce in what strategy should be preferred in which cases. This is the most important conclusion for operators. But it leads to more restrictions concerning gates if handled within the mathematical approach. For a reliable comparison, this needs to be treated in the evolutionary algorithm as well. On the other hand side, the evolutionary approach offering more flexible solutions without the need to fix gates to special tours is highly appreciated by the operators.

Acknowledgment

This work was supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the Collaborative Research Center “Computational Intelligence (531).”

References

Bartz-Beielstein, T. (2006). *Experimental Research in Evolutionary Computation—The New Experimentalism*. Berlin, Heidelberg, New York: Springer.

Beielstein, T., Mehnen, J., Schönemann, L., Schwefel, H.-P., Surmann, T., Weinert, K., & Wiesmann, D. (2003). Design of evolutionary algorithms and applications in surface reconstruction. In H.-P. Schwefel, I. Wegener, & K. Weinert (Eds.), *Advances in Computational Intelligence—Theory and Practice* (pp. 145–193). Berlin, Heidelberg, New York: Springer.

Bermudez, R. & Cole, M. H. (2001). *A genetic algorithm approach to door assignments in breakbulk terminals*. Technical Report MBTC-1102, Mack-Blackwell Transportation Center, University of Arkansas, Fayetteville, Arkansas.

Chmielewski, A. & Clausen, U. (2005). Entwicklung eines Dispositionsleitstandes zur Bestimmung optimaler Torbelegungen in Stückgutspeditionsanlagen. *WGTL Logictics Journal*.

Chmielewski, A. & Clausen, U. (2006). Entwicklung optimaler Torbelegungspläne in Stückgutspeditionsanlagen. In *DSOR Contributions to Information Systems, MKWI 2006, Information Systems in Transport and Traffic*.

Coello Coello, C. A., Van Veldhuizen, D. A., & Lamont, G. B. (2002). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer, New York.

Deb, K. (2001). *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley-Interscience Series in Systems and Optimization. New York NY: Wiley.

Hoos, H. H. (1998). *Stochastic Local Search—Methods, Models, Applications*. PhD thesis, Technische Universität Darmstadt, Germany.

Laumanns, M. (2003). *Analysis and Applications of Evolutionary Multiobjective Optimization Algorithms*. PhD thesis, Swiss Federal Institute of Technology, Zürich, Switzerland.

Li, Y. & Rodrigues, B. (2004). Crossdocking - jit scheduling with time windows. *Journal of Operational Research Society*, 10, 1–10.

Schwefel, H.-P. (1995). *Evolution and Optimum Seeking*. Sixth-Generation Computer Technology. New York NY: Wiley.

Stickel, M. & Furmans, K. (2005). An optimal control policy for crossdocking terminals. In *Proc. of the International Conference of Operations Research 2005*: Gesellschaft für Operations Research (GOR) Springer, Berlin. (accepted for publication).