Modeling of capacitated transportation systems for integral scheduling

Mark Ebben*

Matthieu van der Heijden*

Johann Hurink†

Marco Schutten*

*University of Twente, School of Business, Public Administration and Technology
P.O. Box 217, 7500 AE Enschede, The Netherlands

†University of Twente, Fac. of Electrical Engineering, Mathematics and Computer Science
P.O. Box 217, 7500 AE Enschede, The Netherlands

October 14, 2003

Abstract

Motivated by a planned automated cargo transportation network, we consider transportation problems in which the finite capacity of resources (such as vehicles, docks, parking places) has to be taken into account. For such problems, it is often even difficult to construct a good feasible solution. We present a flexible modeling methodology which allows to construct, evaluate, and improve feasible solutions. This new modeling approach is evaluated on instances stemming from a simulation model of the planned cargo transportation system.

Keywords: Transportation, scheduling, modeling, heuristic

1 Introduction

In this paper, we consider transportation scheduling problems in which the finite capacity of resources (such as vehicle parking places and docks for loading and unloading) has to be taken into account. The motivation of our research is a planned automated cargo transportation network using automatic guided vehicles (AGVs) around Schiphol Airport, the

Netherlands (see van der Heijden et al. [12]). To run such a network, a wide range of coherent decisions has to be taken, such as prioritizing transportation jobs, load consolidation, assignment of loads to AGVs and docks (for loading), assignment of loaded AGVs to docks (for unloading), and redistribution of empty AGVs to cope with imbalanced transportation flows. A suitable logistic planning and control system should be fast and flexible, that is, it should be able to take decisions in real time, taking into account rapidly changing circumstances (such as the arrival of rush jobs and AGV failures), and the limited resource capacities. Van der Heijden et al. [12] developed a local control concept, in which they use relatively simple heuristics and algorithms for each decision. Although such an approach is fast and flexible indeed, efficient resource usage is not guaranteed. Integration of some key decisions may improve the system performance, see Van der Heijden et al. [11] and Ebben et al. [10]. Therefore, it is interesting to determine to which extend and under which circumstances integral optimization improves the system performance in terms of reaching the same service levels using less resources (or reaching higher service levels using the same resources). Obviously, such an integral approach should still be fast and flexible.

Real-life situations often incorporate a lot of side constraints, which are often not included in the available approaches. In our approach, we are able to include resource capacity constraints, such as limited loading/unloading capacity and limited parking space on the terminals. In this paper, we will present a generic approach to model integral network scheduling problems that can meet these requirements. The emphasis is on developing a flexible modeling methodology, facilitating construction, evaluation, and improvement of feasible solutions. Although our approach facilitates optimization, we do not consider such optimization algorithms with their performance in this paper. As an example, we will show how our approach can be implemented for the automated transportation network mentioned above. However, our approach is suitable to deal with other heavily automated transportation and transshipment systems as well, such as scheduling of container terminals, where loading and unloading operations have to be scheduled using cranes and vehicles.

The literature on transportation scheduling under multiple resource constraints is limited. Kozan [14] investigates the minimization of handling and traveling times of containers from the time the ship arrives until all containers from that ship leave the port. He performs a sensitivity analysis with respect to equipment type, trucks, highstackers and shore cranes. Interactions between vehicles and, for example, limited parking space for vehicles are not taken into account. Kozan and Preston [13] present a genetic algorithm for the related problem of optimizing container transfers at multimodal terminals.

Bostel and Dejax [4] address the problem of rail-rail transshipment shunting yards. They propose optimal and heuristic methods to minimize the number of container moves in a terminal. In their models they take intermediate container storage capacity into account.

Meersmans [16] considers the optimization of container handling systems, where he designs models for integrated scheduling of the handling equipment. To solve the models, he develops exact and heuristic algorithms and evaluates the performance of these algorithms.

In his models he does not consider finite parking capacity, i.e. the queuing of AGVs in front of quay cranes and stacking lanes.

Alicke [2] investigated an intermodal terminal concept called Mega Hub. He models the terminal as a multi-stage transshipment problem where sequence-dependent duration of empty moves, alternative assignments of containers to cranes and a sequence-dependent number of operations have to be handled. An optimization model based on Constraint Satisfaction is formulated and heuristics are developed.

The stated requirements for the transportation problem indicate that the considered transportation systems are quite complex and general, and that the resulting problems form a combination of scheduling and vehicle routing aspects. This combination is hardly considered in the literature. Within the scheduling area, transports or routing aspects are often neglected or only simple variants of these aspects are treated:

- within robotic cells only simple movements by a robotic arm are possible (for a survey cf. Crama et al. [8])
- shop problems with transportation times incorporated consider basicly the vehicles as additional machines, i.e. they assume fixed transportation times and neglect the parking aspects within terminals and the handling of empty vehicles (see e.g. Bilge & Ulusoy [3] or Hurink & Knust [13])
- shop problems with buffers consider the parking storage aspects between machines but do not combine these aspects with transportation issues or vehicles (see e.g. Nowicki [17] or Brucker et al. [5]).

On the other hand, in the routing area capacities within locations or timing restrictions are often neglected or treated very simplified:

- in the vehicle routing problem with time windows only timing restrictions on the load/unload operation are given (see e.g. Cordeau et al. [7])
- the capacitated vehicle routing problem considers only capacities for the vehicles but not for the locations (see e.g. Ralphs et al. [18])

Thus, the considered problem forms a new challenging problem where two areas with different approaches are combined. For this problem it is already difficult to develop a reasonable constructive heuristic which takes into account all the given constraints.

The remainder of the paper is organized as follows. In Section 2, we will give a more detailed description of the problems that we consider. Then, in Section 3, we present our approach to model these problems. Section 4 describes how we construct a feasible solution and we apply this approach in Section 5 to a real-life application. In Section 6, we discuss some model extensions, stemming from practical cases. Section 7 ends this paper by giving a number of conclusions and suggestions for further research.

2 Problem description

Transportation systems in general have two main layers: the physical layer and the logistic layer. The physical layer of the system is considered to be fixed. It specifies the locations, where decisions with respect to the transportations have to be made, the possible connections between these locations, and the equipment (vehicles) which is used for the transportations. The logistic layer uses the physical part as input and specifies the orders or tasks which have to be performed by the system. In the following three subsections we specify the transportation system considered in this paper by describing the components of the two layers in more detail. Based on the specifications given in these three subsections, in Subsection 2.4 we discuss the type of decisions which have to be taken within the considered transportation model.

2.1 The system network

The system network is the part of the physical layer which specifies the 'decision' locations (called **terminals** in this paper) and connections between them. In our model, terminals are locations where orders may depart and arrive and, thus, loading and/or unloading operations take place. These terminals may have a rather complex substructure. In the considered transportation system, a terminal consists of a terminal parking to store vehicles and a set of docks to handle load and unload operations. Within the terminal, connections between the parking and the docks are given, i.e. the terminal contains a small 'system network' itself. We assume that the structure within a terminal can be expressed by travel times necessary to travel between the parking and the docks. In Figure 1 a layout of one of the planned terminals for the cargo transportation system around Schiphol airport (see also Section 5) is given. Although, in general, terminals may have a rather complex structure, also special terminals which contain only a parking or only docks may exist.

Between the terminals connections exist. We assume that each existing direct connection between two terminals consists of a unidirectional single track without any intermediate terminals on this track. For each connection a length is given. This length is used to estimate the travel time for the connection.

It remains to describe the ingredients of a terminal - docks and parkings - in more detail.

Docks: Docks are the places where the vehicles are physically loaded and unloaded. They consist of a dock parking (often very small - place for one or two vehicles or even no parking) and a set of parallel servers (often only one). A vehicle enters a dock via the parking and will be directed to one of the servers, where the load or unload operation takes place. Within a dock the times for loading or unloading a vehicle on a server and the setup time between two consecutive vehicles on a server (minimal time between the departure of a vehicle and the arrival of the next vehicle on the server) are given.

Parkings: A parking (terminal parking as well as dock parking) is specified by the number

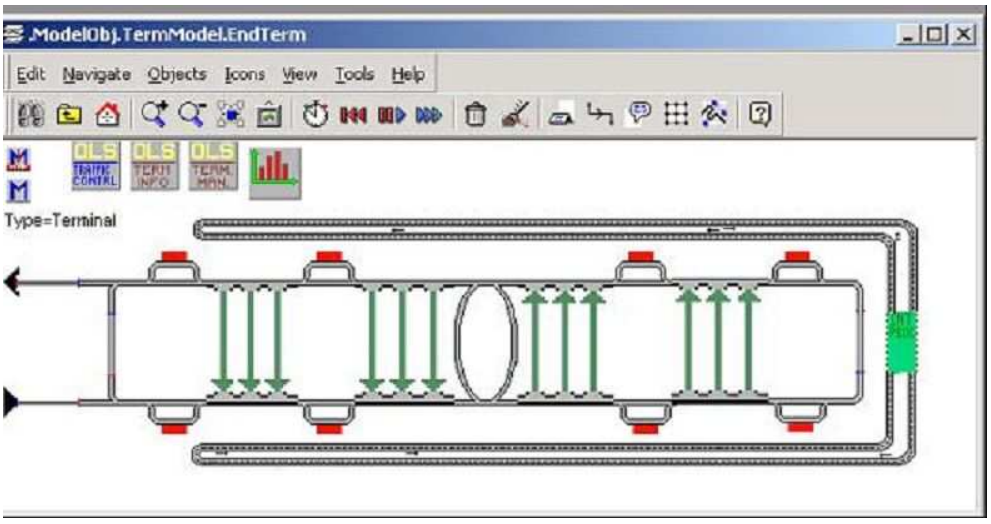


Figure 1: Layout of a terminal.

of vehicles which jointly may occupy the parking and by the operating mode of the parking. As operating modes we consider FIFO (the vehicles have to leave the parking in the same order in which they entered it) or an arbitrary mode.

2.2 The transportation equipment

In general there are three important characteristics of vehicles. One main characteristic is given by the number of items a vehicle can transport. The simplest case occurs if each vehicle can transport only one *transportation unit* (e.g. a pallet or a container) at a time. A transportation unit, however, can consist of several items. AGV systems usually use transportation units to speed up loading and unloading operations. In this single-unit case, the route of a vehicle is characterized by a sequence of orders assigned to the vehicle. The vehicle has to travel from the origin to the destination of an order and, afterwards, to the origin of the next order. In our model we assume that there is exactly one route between each pair of locations. The more general case where each vehicle can transport multiple transportation units at a time leads to a more complex routing problem: for the vehicle a sequence in which the orders assigned to the vehicle are picked up and delivered has to be determined. This sequence must be chosen such that the capacity of the vehicle is respected.

A second important characteristic of the transportation equipment is given by the speed of the vehicles. Again, the homogeneous case, where all vehicles have the same speed, is the easiest to deal with. For the heterogeneous case (different speeds for the vehicles) congestion may have a large influence on the behavior of the system.

A third characteristic of the transportation equipment is the size of the vehicles. If vehicles

of different size are present, some of the terminals/docks may not be reachable by all vehicles and, thus, side constraints on the assignment of orders to vehicles have to be taken into account.

In this paper we consider the simplest case of homogeneous vehicles with respect to speed and size which can transport only one order/item a time. As a consequence, in our model no congestion on tracks is taken into account and the arrival time of a transportation is equal to the departure time plus a fixed travel time. Furthermore, the only specification for this part of the system is the number of available vehicles. However, the presented approaches can be adapted to different vehicle sizes without much effort (see Section 6). An adaptation to vehicles with different speed and to vehicles that can transport several orders a time is not straightforward and asks for new techniques.

2.3 The orders

Following the choice for the vehicles, the orders are considered to be one-item orders. The origin and destination of an order are specified on terminal level. Furthermore, an earliest departure time (release date) from the origin terminal and a latest arrival time (due date) at the destination terminal are given.

2.4 Decisions

The basic decision within the presented model is when and how to transport the given orders from their origin to their destination terminal.

These decisions lead to

- an assignment of orders to vehicles,
- for each vehicle a sequence in which this vehicle transports the orders assigned to it,
- an assignment of vehicles to parkings and/or docks and servers, and
- a timing of all resulting transportations.

All these decisions have to be done in such a way that the resulting solution is feasible. To achieve feasibility, one has to ensure that

- the capacities of the parkings are respected,
- the parking type (FIFO, arbitrary) is respected,
- servers handle only one vehicle at a time,

- the 'service' times (load, unload, setup, ...) and transportation times are respected, and
- the release dates are respected.

We do not consider due dates as a restriction, but we incorporate them in a performance measure.

3 Representation of solutions

In the previous section we have seen that solutions consist of assignments, sequences, and timing of orders, vehicles, and transportations taking into account the capacities within the terminals and the timing constraints. Such a solution may be represented by a set of transportations and a corresponding timing of them. However, for the considered problem it is already difficult to develop a reasonable constructive heuristic which takes into account all the given constraints. This difficulty arises from the fixed travel times between locations and the finite capacities of the parkings. Furthermore, in the area of scheduling it is not very handy to use the concrete timing of activities within the representation of solutions since iterative as well as enumerative methods have problems with such representations. As a consequence, mainly assignments and sequences are used to represent solutions. These representations are chosen such that a corresponding timing can be calculated efficiently.

In the following, we give such a representation of solutions for the considered transportation model (Section 3.1) and a corresponding method for the calculation of the timing (Section 3.2). The basic idea is to represent solutions by sequences of transportations. For a given set of sequences a graph is constructed and via a longest path calculation in this graph a timing of the transportations (i.e. a feasible schedule) is achieved (see Figure 2). To construct a feasible schedule using this representation, it remains to give a concrete set of sequences of transportations (first element in Figure 2). In Section 4, we give a corresponding method for constructing such a set of sequences. The basic idea is to use a heuristic to construct a schedule for a relaxation of the considered problem (a relaxation for which a reasonable constructive heuristic is easy to achieve). Then, we extract from this schedule the used transportations and their sequences within the locations. Finally, we calculate for these sequences, with the method developed in this section, a feasible schedule for the whole problem (i.e. the non-relaxed problem).

The basic elements of the representation are transportations. Each transportation is characterized by its origin and destination location and the order which is transported (also 'empty' orders may be assigned). The basic structures of the representation are sequences of these transportations. The sequences are introduced for certain types of locations (docks, parkings) and for vehicles. For a vehicle, the sequence gives all transportations to be carried out by the vehicle (including empty transportations). For the locations, the sequences are chosen such that they enable us to determine a timing of the transportations within

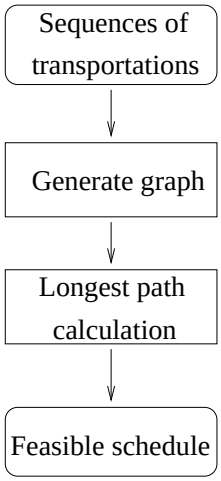


Figure 2: Solution representation

the locations. Whereas for pure scheduling problems or scheduling problems with simple transportation models one sequence per location (machines, robots, etc.) is sufficient (see e.g. [13]), it will turn out that for docks and parkings one insequence and one outsequence are needed. Given a set of transportation sequences for the locations, in a straightforward manner, the assignments of vehicles to parkings/docks can be achieved by just looking in which dock- or parking-sequence a certain vehicle occurs. The assignment of vehicles to servers within a dock and the timing of all the transportations needs some more effort. For each location, the corresponding sequences are checked on consistency (local feasibility). This consistency is necessary but not sufficient for being able to determine a feasible timing on the base of the given sequences. Together with the consistency, for a dock also the assignment of the vehicles to the servers is determined. Furthermore, based on the transportation sequences for the locations and vehicles, timing relations for the transportations are derived. Together with some additional timing constraints resulting from the orders, these relations are represented by a directed graph and the timing of the transportations results from longest path calculations in this graph (during this calculation, also global infeasibility can be detected). In the remainder of this section, we present this approach in more detail.

3.1 Required transportation sequences

As mentioned above, the basic elements of the representation are transportations. A transportation consists of:

- an origin and a destination location: These locations are either a terminal parking or a dock. Between these two locations, a direct transportation (without passing any other terminal) has to be possible;

- a vehicle: It indicates which vehicle handles the transportations;
- an order: It indicates which order is transported (the order may be an 'empty' order);
- a travel time.

For such a transportation the departure time at the origin location and the arrival time at the destination location have to be determined. These times have to differ exactly by the travel time.

By specifying a set of transportations (without timing!), already some decisions are made: For each order, the subset of transportations corresponding to this order specifies the route the order will take through the network and whether or not the order has to go via a terminal parking. Furthermore, the transportation starting at the origin terminal specifies the dock at which the order is loaded and the transportation ending at the destination terminal specifies the dock at which the order is unloaded. Finally, the assignment of vehicles to orders has been made.

To achieve a timing of the transportations and, thus, a complete schedule, we have to couple the transportations to avoid conflicts at the different resources (vehicles, parkings, servers, etc.). This is realized by sequences of transportations.

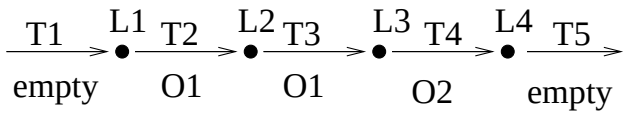
- Each vehicle gets one sequence. This sequence indicates which transportations are carried out by the vehicle and the corresponding ordering.
- Each dock and each terminal parking get an insequence and an outsequence. They indicate the ordering in which the transportations (i.e. the vehicles which carry out the transportations) enter and leave the dock or parking.

Each transportation is inserted in the outsequence of the origin location, in the insequence of the destination location, and in the sequence of the vehicle which handles the transportation.

The sequences indicate the ordering in which the transportations will be handled. However, they do not directly lead to a schedule (start times for the transportations). In the remainder of this section, we present necessary conditions which have to be fulfilled by the sequences belonging to a location or to a vehicle to enable a feasible schedule respecting a given set of transportation sequences and a method to calculate a timing of the transportations.

For consistency, the following conditions have to be satisfied:

- **Vehicle sequence**:
  Two consecutive transportations in a vehicle sequence must 'fit' together; i.e.
  - the destination location of the first transportation must be equal to the origin destination of the second,



Order O1: from location L1 to location L3

Order O2: from location L3 to location L4

Figure 3: A sequence of transportations for a vehicle

  - if the orders of the two transportations are different (see transportations $(T1, T2)$, $(T3, T4)$, $(T4, T5)$ in Figure 3),
    - the first transportation either must be an empty transportation (transportation $T1$ in Figure 3) or its destination location must be equal to the destination location of the first order (where it will be unloaded) (transportations $T3$ or $T4$ in Figure 3) and
    - the second transportation either must be an empty transportation (transportation $T5$ in Figure 3) or its origin location must be equal to the origin location of the second order (where it will be loaded) (transportations $T2$ or $T4$ in Figure 3),
  - all transportations related to an order must be sequenced consecutively.

- **Parking**:
  If the operating mode of the parking is FIFO, the in- and outsequence of the parking must be equal in the sense that the vehicles corresponding to the ith transportations in both sequences are the same. If we have a parking with arbitrary operating mode and capacity c, we have to ensure that if a vehicle arrives at the parking, not more than $c - 1$ vehicles are allowed to be in the parking and, thus, the vehicle can be no more than $c - 1$ positions earlier in the outsequence than in the insequence, i.e. if a transportation occurs within the outsequence at position i, its vehicle has to correspond to a transportation which occurs in the insequence not later than at position $i + c$. Figure 4 gives an example with $c = 3$. If in this example Vehicle 5 would be at position 2 of the outsequence (i.e. change its position with Vehicle 2), then at the time Vehicle 5 arrives at the parking, Vehicles 1, 2, and 4 have to be waiting in the parking. But since the parking has only capacity 3, Vehicle 5 has no free track to pass these vehicles.

- **Docks**:
  A dock consists of a dock parking (denote its capacity by c) and a set of servers (denote the number of servers by s). Within the dock a vehicle may pass only a limited number of other vehicles. This number depends on s and, if the operating mode of the dock parking is not FIFO, also on c: if the parking is a FIFO parking, a

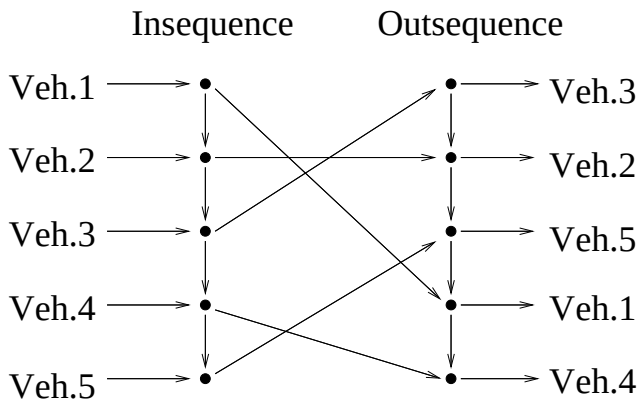


Figure 4: Arbitrary Parking with capacity $c = 3$

vehicle may pass $s - 1$ vehicles and, otherwise, it may pass $s + c - 1$ vehicles. Thus, the restrictions on the sequences can be derived in the same way as for parkings.

3.2 Timing relations

After checking the consistency of the sequences, it remains to determine a timing of the transportations. The sequences at one location or for one vehicle lead to 'local' timing relations between transportations. These relations may be represented by a directed graph. Using this graph, a schedule respecting the given sequences can be calculated by longest path calculations (similar to critical path calculations for project scheduling, cf. [6]) and feasibility checking is reduced to checking whether the graph contains a positive cycle or not. In the following we will describe the construction of the graph, representing the timing relations resulting from a given set of sequences, in more detail.

The timing relations will be represented by a graph $G = (V, A, l)$, where V denotes the set of vertices, A the set of directed arcs, and $l : A \to \mathbb{R}$ is the length function for the arcs. Each vertex represents either a departure or an arrival of a transportation and a solution is given by an assignment of times $s(i)$ to all vertices $i \in V$. An arc $(i, j) \in A$ with length l_{ij} expresses a timing relation of the form

$$s(i) + l_{ij} \leq s(j),$$

i.e. event j cannot start before the start of the event i plus l_{ij}. Since the above timing relations correspond to longest path conditions, a schedule respecting all timing relations from the arc set A can be achieved by longest path calculations.

It remains to explain how the graph corresponding to a fixed set of transportations and sequences is constructed (we assume that the given sequences are consistent; see Subsection 3.1).

- **Vertices**:
  For each transportation t two vertices t_d and t_a are introduced. Vertex t_d corresponds to the departure and vertex t_a to the arrival of t.

- **Travel times**:
  Since the travel time (let Δ be this time) is fixed for a given transportation t, the times $s(t_d)$ and $s(t_a)$ must differ exactly Δ; i.e.

  $$s(t_d) + \Delta = s(t_a).$$

  This relation can be expressed by two arcs (t_d, t_a) and (t_a, t_d) of length $l_{t_d,t_a} = \Delta$ and $l_{t_a,t_d} = -\Delta$. The arc (t_d, t_a) of length Δ ensures that $s(t_a) \geq s(t_d) + \Delta$, whereas the arc (t_a, t_d) of length $-\Delta$ ensures that $s(t_d) \geq s(t_a) - \Delta$. Combining these two inequalities results in the required equality.

- **Order**:
  We have to ensure that the departure of the first transportation belonging to an order does not depart before the release date. This can be ensured by an arc from a dummy node representing the start of the schedule and the node corresponding to the departure of the first transportation with a length equal to the release date.

- **Parking**:
  If the parking has FIFO mode, the in- and outsequence of the parking must be the same. First, we have to ensure that the transportations arrive in the order given by the insequence, depart in the order of the outsequence, and that between two consecutive arrivals (departures) a minimal time lag s_{in} (s_{out}) is taken into account. These time lags represent, e.g., safety distances. More precisely, if $\tilde{t}$ and $\hat{t}$ are two consecutive transportations in the insequence (outsequence), the timing relation

  $$s(\tilde{t}_a) + s_{in} \leq s(\hat{t}_a) \quad (s(\tilde{t}_d) + s_{out} \leq s(\hat{t}_d)) \tag{1}$$

  has to be fulfilled. Furthermore, we have to ensure that a vehicle does not leave the parking with a transportation $\hat{t}$ before the vehicle has arrived with its previous transportation $\tilde{t}$ i.e.

  $$s(\tilde{t}_a) + s_{park} \leq s(\hat{t}_d), \tag{2}$$

  where s_{park} denotes a minimal time needed between the vehicle entering and leaving the parking. Finally, we have to ensure that the parking capacity is respected, i.e. for a parking with capacity c the $(i+c)$th transportation $\tilde{t}$ cannot enter the parking before the ith transportation $\hat{t}$ has left the parking; i.e.

  $$s(\tilde{t}_a) \geq s(\hat{t}_d). \tag{3}$$

- **Dock**:
  In the same way as for parkings, we ensure that the transportations respect the in-

and outsequence and corresponding minimal time lags, the vehicle sequence, as well as the capacity of the dock (see (1), (2) and (3)). It remains to take into account the minimal time distance between two transportations using the same server. These minimal time distances represent, e.g., setup times between two consecutive loading or unloading operations on a server. If in the dock only one server is available, these timing relations can be incorporated in the relation (1) for the outsequence. If more than one server is available, first an assignment of vehicles to servers is calculated and then the minimal time lags for two transportations processed consecutively on a server are added. A best assignment respecting the given in- and outsequence can be calculated as follows: when a server gets free (the order in which they get free is determined by the outsequence) we try to put directly onto the free server that vehicle (transportation) which comes first in the outsequence and which has not been assigned to a server yet. If this is not possible (the capacity of the dock parking does not allow this vehicle to enter the dock) we first move vehicles from the dock parking to servers until the specified vehicle can enter the dock and move to the server. The choice, which vehicles (transportation) leave the dock parking is again determined by the order in which the transportations occur in the outsequence. In Figure 5 an example of a dock with 2 servers and a parking of capacity 2 is given. Using the given in- and outsequences as input for the above procedure, we get that Vehicle 1 and 2 enter the parking, that Vehicle 3, 5 and 2 are handled by Server 1, and that Vehicle 4 and 1 are handled by Server 2. The resulting arcs together with their corresponding length now can be added to the graph G. Note that the given construction is not based on a concrete timing of the transportations, but considers only sequences in which tasks are executed.

The (untill now unknown) values $s(i)$ express the starting times of the corresponding events and have to fulfill all the introduced timing relations. Therefore, these values can be achieved by calculating the longest path from the dummy node representing the start of the schedule to all vertices. If this longest path calculation detects no global infeasibility (positive cycles), the resulting schedule has earliest possible departure and arrival times within the class of all schedules respecting the given vehicle-, dock-, and parking sequences. Note that due to the fixed travel times also negative arc lengths l_{ij} occur and, thus, more general longest path methods like the Floyd-Warshall algorithm (cf. Ahuja et al. [1]) have to be used to calculate a best schedule respecting a given set of sequences.

3.3 Possible objective functions

As indicated above, the longest path calculation ensures that for the achieved schedule the starting times of the transportations cannot be decreased without changing in at least one location one of the sequences of the transportations. Therefore, our approach is able to cope with any objective function for which it is optimal that, *given the sets of sequences for the transportations*, all transportations are performed as early as possible (i.e. for reg-

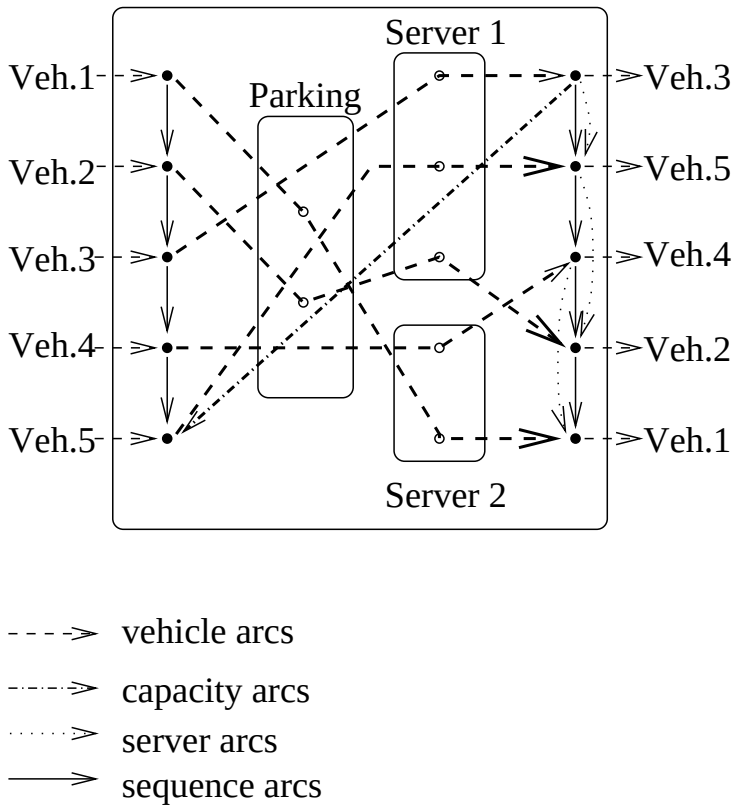


Figure 5: A dock with 2 servers and a parking of capacity 2

ular objectives). Such objective functions include minimizing the number of late orders, minimizing the maximum or mean lateness, minimizing the time to complete all orders ('makespan'), and minimizing the total length of empty vehicle movements. For the latter objective function, the timing of the transportation is not important, given the sets of sequences for the transportations. Therefore, it is also optimal to perform the transportations as early as possible. An example of an objective function that cannot be handled by our model is minimizing earliness costs. In this case, it might be better to postpone a transportation, although it would be feasible to perform it now.

4 Heuristic approach

In this section, we show how the representation given in the previous section can be used to develop in a very simple way a heuristic solution approach for the considered transportation problem. The basic idea is to first treat a relaxation of the problem and solve this relaxation with a constructive heuristic. From the resulting schedule (which in general is infeasible for the original problem) now only the relevant information for a solution representation

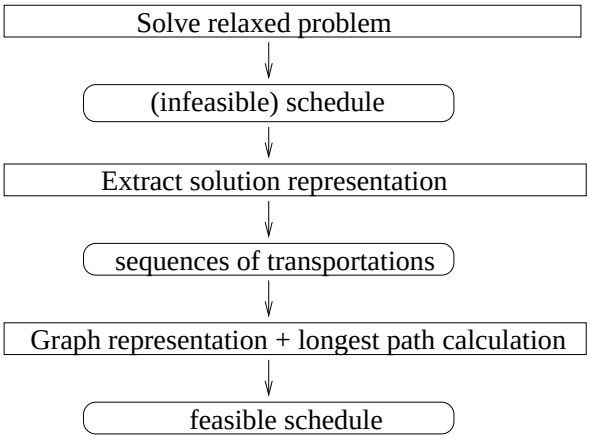


Figure 6: Structure of the heuristic

(vehicle and in- and outsequences) is extracted and used as input for the method presented in the previous section. This results in a feasible schedule for the original non-relaxed transportation problem (see Figure 6). In the following we describe this method a bit more detailed.

Due to the fixed travel times between locations and the finite capacities of parkings, it is not trivial to construct a good feasible schedule for the considered transportation problem. Therefore, we chose for a relaxation where the capacity constraints of a terminal parking and minimal time lags for vehicles arriving at a terminal may be violated. The resulting problem now can be solved using a standard 'event based' heuristic using priority rules for making the decisions. The outcome of this priority driven heuristic is a schedule which consists of a set of transportations and a corresponding timing of these transportations. However, this schedule may violate some of the constraints mentioned in Sections 2 and 3.

To achieve a feasible schedule we neglect the determined timing of the transportations and use only the structural information of the 'infeasible' schedule (the introduced transportations, the vehicle sequences, and the in- and outsequences of the locations). This structural information gives a representation of a solution, and based on the method presented in Section 3 a new schedule is calculated. Although this new solution has the same structure as the previous solution, it may differ from it:

- transportations from different locations may arrive at a parking at the same time in the previous schedule but must have a safety distance in the new schedule
- in the previous schedule, a vehicle may arrive at a terminal at a time that no parking space is available, whereas in the new schedule this vehicle has to leave the origin terminal later to be able to insert it directly in a parking
- in the previous schedule, a vehicle is sent to its next location/order (parking, server, next order) at the time the next location gets free or the next order is assigned to

it, whereas in the new schedule the vehicle can anticipate on the next task already earlier.

Summarizing, the new schedule now takes into account all restrictions and the transportations may anticipate on the 'knowledge' that servers/parkings get free or orders arrive. Consequently, the makespan of the new schedule may differ from that of the previous schedule.

5 An application

In this section, we apply our model and heuristic to instances derived from a practical case. It is one of the planned layouts for an automated cargo transportation system around Schiphol Airport (The Netherlands), which we discussed in the introduction. We will refer to this application as the OLS case, in which 'OLS' is a Dutch abbreviation for underground logistic system. Subsection 5.1 gives some details on this case. Afterwards, in Subsection 5.2, we give some results of computational experiments.

5.1 The OLS case

In this section, we give a rough sketch of the application that we used to test our modeling framework. For details, we refer to Ebben [9].

To avoid road congestion, a highly automated underground transportation system using automatic guided vehicles (AGVs) is being developed around Schiphol Airport, connecting the airport with the world's largest flower auction market in Aalsmeer and a planned rail terminal (near Hoofddorp or Schiphol Airport). The system should be able to handle different order deadlines and should guarantee reliable throughput times. It is unique in its scale, incorporating 16-25 km tubes connecting five to 20 terminals, and it includes up to 200 AGVs to transport an estimated 3.5 million tons of cargo in 2020. To operate this system in a reliable and efficient way, it would need an innovative planning and control system for logistics, which takes the order deadlines and limited resource capacities at the terminals into account.

In the layout, we consider in the experiments in the next subsection, the system consists of 2 terminals with 6 docks at Aalsmeer (VBA), 5 terminals with 2 docks at Schiphol Airport (AAS), which are located on a one-directional loop, and 1 rail terminal with 10 docks at Hoofddorp (RTH) (see Figure 7). There is a central vehicle parking at Schiphol Airport and each terminal has a limited parking capacity. We investigate three different patterns of transportation flows, with the following characteristics:

1. The transportation flows are reasonably distributed over all origin-destination pairs. In this case, the system requires about 120 vehicles to ensure acceptable service levels.

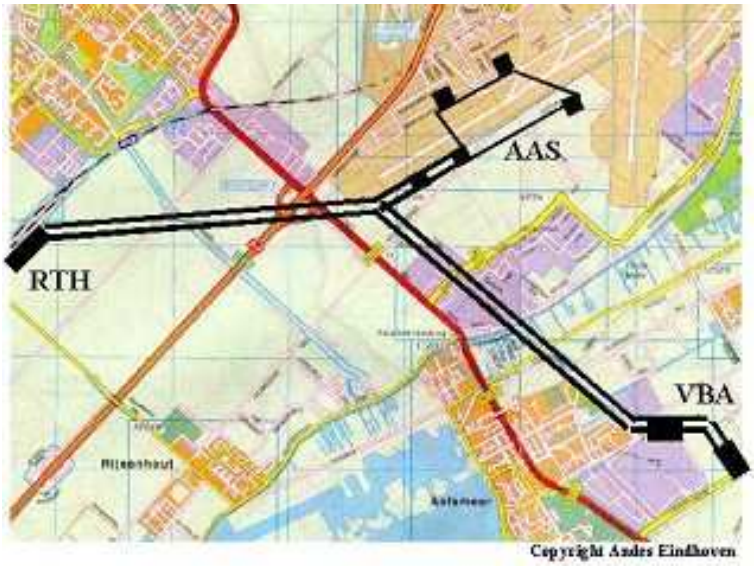


Figure 7: Considered layout at Schiphol.

2. Most transportations are between two locations: the flower auction (VBA) and the rail terminal (RTH). Therefore, most traffic is on one route, which should make the vehicle planning rather easy. The total number of transportations is larger than in the previous case and the distance between the flower auction and the rail terminal is the longest distance in the system; this causes that 200 vehicles are needed.

3. Peak levels in transportation flows move quickly from one route to another, which means that it is essential to anticipate heavily fluctuating transportation demands. To guarantee acceptable service levels 165 vehicles are used.

5.2 Results

For the OLS case mentioned above, a simulation model in eM-plant is available (cf. Ebben [9]). Using this model, we simulated 30 days for each of the 3 mentioned cases. Each 30 minutes of simulation time, the simulation model saves the current status information to a database. This information primarily concerns the current vehicle activities (current load, position, and destination) and the available orders. This database is then used as input for making a schedule. We start with applying the event-based heuristic, which may result in a non-feasible schedule. From this schedule, we use the found sequences to construct a graph in which we perform a longest path calculation to determine the timing of each transportation (see Section 4). The schedule is then saved to the same database and read by the simulation model which then follows this schedule for the next 30 minutes. After that, new status information is saved to the database and a new schedule

Case	Nodes	Arcs	GC Time	Path Time	Makespan		Late jobs	
					Heur	Path	Heur	Path
Case 1	3,338.3	9,742.1	0.03	0.01	232.3	230.1	23.3	22.6
Case 2	5,311.6	15,438.4	0.08	0.01	274.3	258.1	168.1	151.7
Case 3	5,236.9	15,271.7	0.05	0.01	231.4	230.3	25.9	24.0

Table 1: Average results for the scheduling instances.

Case	Nodes	Arcs	GC Time	Path Time	Makespan		Late jobs	
					Heur	Path	Heur	Path
Case 1	5,066	15,340	0.08	0.02	300	289	199	199
Case 2	9,944	29,877	0.22	0.05	482	421	609	571
Case 3	6,636	19,852	0.08	0.02	273	272	121	116

Table 2: Maximum result values for the scheduling instances.

is constructed. In this way, we generate 1441 scheduling instances for each of the three cases.

Table 1 shows scheduling results for the generated instances. The column 'Nodes' shows the average number of nodes in the constructed graph, whereas the column 'Arcs' specifies the average number of arcs in this graph. The column 'GC Time' gives the average time (in seconds) to construct the graph (using a computer with an AMD Athlon 1800+ processor and 512 MB of memory). Column 'Path Time' shows the average time (in seconds) to do the longest path calculations (using the same computer). The columns under 'Makespan' give the average makespan in minutes for the event-based heuristic (column 'Heur') and for the schedule based on the longest path calculations (column 'Path'). Analogously, the next two columns present this data for the average number of late jobs. Table 2 shows the same results, but now for the maximum values instead of averages.

From Tables 1 and 2, we conclude that most of the required computation time is spent on the construction of the graph and that the time for the longest path calculations is small. Therefore, we see good possibilities to use our model in local search methods. In these methods, the local changes imply small changes in the graph (which can be realised by updates instead of complete new constructions) after which the consequences of the changes can be evaluated by (fast) longest path calculations.

The final schedule is on average and on the maximum always better than the heuristic solution (on makespan and on the number of late jobs). A detailed analysis of the results shows that only for 8 (4) instances out of 4,323 the event-based heuristic had a slightly better makespan (number of late jobs), whereas for 3,413 (2,772) instances the final schedule is better. Therefore, apart from correcting infeasibilities, the final schedule improves the solution quality.

6 Model extensions

The graph representation of a transportation system as presented in this paper has proven to be flexible with respect to incorporating practical extensions. In this section, we will discuss some possible extensions.

In Section 2, we discussed that one of the characteristics of a parking is whether vehicles have to leave the parking in a FIFO order or not. In the OLS case, parkings have an additional characteristic. For each parking it is given what type of vehicles can use this parking: Empty vehicles, loaded vehicles, or both. This feature is easily incorporated in the model by taking care of this in the heuristic. The heuristic produces then in- and outsequences for the parkings taking into account the load status of the vehicles. These sequences are used to perform the longest path calculations.

The second extension in the OLS case is more difficult. Due to the large costs of building underground tubes, in the current plans for the track layout, the connection between Schiphol and Aalsmeer will be a single track to be used in both directions. This means that when a vehicle traveling from Schiphol to Aalsmeer enters the tube, only upon arrival of this vehicle in Aalsmeer another vehicle can enter the tube traveling from Aalsmeer to Schiphol. Therefore, we have to decide when to open a track for vehicles traveling in a certain direction. We modeled this bidirectional track as a terminal (in which important decisions have to be made). A terminal can consist of docks and parkings, but now also of a bidirectional track. Associated with each bidirectional track, there are two FIFO parkings (one for each direction). For the bidirectional track, we consider now *two insequences* and *one outsequence*. Each insequence represents the vehicle arrivals in one direction. The outsequence represents the order in which the vehicles use the bidirectional track. If two consecutive vehicles in the outsequence travel in the same direction, then there should be a small safety time between them. If two consecutive vehicles travel in opposite directions, then there should be a large 'safety time' between them: The first vehicle has to travel the complete track before the second vehicle can enter it. As done in Section 3 for the other terminals, we can derive for this new type of terminal a procedure, which checks the local feasibility of the sequences and a procedure which creates the timing restrictions resulting from the sequence. Furthermore, the event-based heuristic has to be enlarged with the event of an arrival at a 'bidirectional terminal' and a procedure that determines when the track can be used for transportations in a certain direction.

Load and unload times at the servers of a dock may depend on the goods that are loaded or unloaded. For example, large loads may require more load time than small loads. This can easily be incorporated by varying the length of the arcs between the node for the arrival of a vehicle at a dock and the node for the next departure of this vehicle at this dock.

Sometimes, (parts of) loads are consolidated at a terminal to a 'new' load. This means that there are precedence relations between the arrival of the loads that will be consolidated and the departure of the consolidated load. This can easily be incorported in the model by adding arcs between the nodes associated with the arrival of the loads that will be consoli-

dated and the node associated with the departure of the consolidated load. Furthermore, if the consolidation decisions have to be made within the model, the solution representation does not change; only the heuristic has to be adapted.

In some systems, vehicles may have different sizes, which implies that the 'large' vehicles may not be able to enter all terminals. In this case, only the procedure that assigns vehicles to orders has to be adapted.

The above discussion shows that the introduced concept of dealing with transportation systems with restricted capacities is quite general and can be adapted to more general systems: If in a transportation system a decision structure comes up, where the local restrictions on feasibility and timing can be expressed by in- and outsequences, these structures can be integrated in the presented approach.

7 Conclusions and further research

We presented a model that is able to deal with a broad class of capacitated transportation systems for integral scheduling. These type of problems require a combination of scheduling and transportation approaches. We have chosen to adapt the graph representation used in scheduling theory to handle combined scheduling and vehicle routing problems. The base of our methodology is a representation of solutions using sequences for the arrivals at and the departures from a location. Based on these sequences, a timing of the transports can be determined by using longest path calculations. We demonstrated the capabilities of our representation by showing that the sequences resulting from an (infeasible) solution generated by a simple event-based heuristic can be transformed to better and feasible solutions.

Experiments for a real-life case have shown that the computational effort for constructing a graph that represents a solution and the longest path calculation in this graph are very small. Moreover, in almost all cases the final schedule improves the solution of the event-based heuristic. In addition, we demonstrated that our approach can be exted to various practical extensions of the real-life case. Therefore, we may state that our structural solution representation has high value in modeling capacitated transportation systems.

Furthermore, our representation facilitates the use of concepts which have led to powerful local search approaches for various scheduling problems. These concepts are based on neighborhoods changing relevant sequences in the graph representation (see, e.g., Vaessens et al. [19]).

Future research may include a local search approach in which the neighborhood structure is based on changes in the sequences. The effect of a change in a sequence can then be evaluated easily by performing a longest path calculation on the resulting graph. For example for the OLS case, it is also interesting how this approach performs in a rolling horizon environment where new orders arrive and all kinds of disturbances may occur. An

important question is then what a good strategy is for rescheduling the system.

Acknowledgements The authors gratefully acknowledge the constructive comments of the anonymous referees, which helped to improve the presentation of the paper a lot.

References

[1] **Ahuja, R.K., Magnanti, T.L., Orlin, J.B. [1993]** Network flows: Theory, Algorithms and Applications, Prentice-Hall, New Jersey.

[2] **Alicke, K. [2002]** Modeling and optimization of the intermodal terminal Mega Hub, OR Spectrum 24, 1-18.

[3] **Bilge, Ü., Ulusoy, G. [1995]** A time window approach to simultaneous scheduling of machines and material handling system in an FMS, Operations Research 43, 1058-1070.

[4] **Bostel, N., Dejax, P. [1998]** Models and algorithms for container allocation problems on trains in a rapid transshipment shunting yard, Transportation Science 32 no. 4, 370-379.

[5] **Brucker, P., Heitmann, S., Hurink, J.L. [2003]** Flow-shop problems with intermediate buffers, OR Spectrum 25, 549 - 574.

[6] **Chao, X. and Pinedo, M. [1998]** Operations Scheduling with Applications in Manufacturing and Services, McGraw Hill.

[7] **Cordeau, J., Desaulniers, G., Desrosiers, J., Solomon, M., Soumis, F. [2002]** The vehicle routing problem with time windows. In P. Toth and D. Vigo (eds.): The vehicle routing problem, SIAM Monographs on Discrete Mathematics and Applications, vol. 9, Philadelphia, PA, 157-193.

[8] **Crama, Y., Kats, V., Van de Klundert, J., Levner, E. [2000]** Cyclic scheduling in robotic flowshops, Annals of Operations Research 96, 97-124.

[9] **Ebben, M.J.R. [2001]** Logistics control of automated transportation networks. PhD thesis, University of Twente, Twente University Press, Enschede, The Netherlands.

[10] **Ebben, M.J.R., van der Heijden, M.C., van Harten, A. [2002]** Dynamic resource-constrained vehicle scheduling, Working paper, University of Twente, Faculty of Technology and Management, The Netherlands.

[11] **Van der Heijden, M.C., Ebben, M.J.R., Gademann, N., van Harten, A. [2002]** Scheduling vehicles in transportation networks, OR Spectrum 24 no. 1, 31-58.

[12] **Van der Heijden, M.C., van Harten, A., Ebben, M.J.R., Saanen, Y.S., Valentin, E., Verbraeck, A. [2002]** Using simulation to design an automated underground system for transporting freight around Schiphol Airport, Interfaces 32 no. 4, 1-19.

[13] **Hurink, J., Knust, S. [2001]** Tabu Search Algorithms for Job-Shop Problems with a Single Transport Robot, Memorandum No. 1579, University of Twente, Faculty of Mathematical Sciences, The Netherlands, to appear in EJOR.

[14] **Kozan, E. [2000]** Optimising Container Transfers at Multimodal Terminals, Mathematical and Computer Modelling 21, 235-243.

[15] **Kozan, E., Preston, P. [1999]** Genetic Algorithms to schedule container transfers at multimodal terminals, International Transactions in Operational Research 6, 311-329.

[16] **Meersmans, P. [2002]** Optimization of Container Handling Systems, PhD thesis, Erasmus University Rotterdam, The Netherlands.

[17] **Nowicki, E. [1999]** The permutation flow shop with buffers: A tabu search approach, European Journal of Operational Research 116, 205-219.

[18] **Ralphs, T.K., Kopman, L., Pulleyblank, W.R., Trotter Jr., L.E. [2003]** On the Capacitated Vehicle Routing Problem, Mathematical Programming 94, 343-359.

[19] **Vaessens, R.J.M., Aarts, E.H.L., Lenstra, J.K. [1996]** Job Shop Scheduling by Local Search, INFORMS Journal on Computing 8, 302-317.