OR Spectrum (2004) 26: 579–601
DOI: 10.1007/s00291-004-0174-6

OR Spectrum

Scheduling the factory pick-up of new cars*

Christoph Mellentien[1], **Christoph Schwindt**[2], **and Norbert Trautmann**[1]

[1] Institut für Wirtschaftstheorie und Operations Research, Universität Karlsruhe, 76128 Karlsruhe, Germany (e-mail: {mellentien,trautmann}@wior.uni-karlsruhe.de)
[2] Institut für Wirtschaftswissenschaft, Technische Universität Clausthal, 38678 Clausthal-Zellerfeld, Germany (e-mail: christoph.schwindt@tu-clausthal.de)

Abstract. Car manufacturers increasingly offer delivery programs for the factory pick-up of new cars. Such a program consists of a broad range of event-marketing activities. In this paper we investigate the problem of scheduling the delivery program activities of one day such that the sum of the customers' waiting times is minimized. We show how to model this problem as a resource-constrained project scheduling problem with nonregular objective function, and we present a relaxation-based beam-search solution heuristic. The relaxations are solved by exploiting a duality relationship between temporal scheduling and min-cost network flow problems. This approach has been developed in cooperation with a German automaker. The performance of the heuristic has been evaluated based on practical and randomly generated test instances.

Keywords: Customer relationship management – Factory pick-up – Resource-constrained project scheduling – Minimum-cost flows

1 Introduction

Within the last two decades, car manufacturers have outsourced a large part of their development and production activities to suppliers. From 1980 to 2001, the manufacturing penetration has decreased from 38% to 25% (cf. VDA, 2003). This means that today, automakers contribute to less than one fourth of the total gross-value added. As a result of this evolution and the increasing trend towards mass customization in automotive industry (see Meyr, 2004), cars of different brands are more and more based on the same standardized components. Thus, besides classical

* The authors would like to thank Margit Frank (Porsche AG) and Benjamin Müller (University of Bern) for their valuable contributions to this work.
Correspondence to: C. Mellentien

criteria like technical equipment and quality, nowadays brand-management and especially establishing an emotional relationship between customers and the brand are seen as key marketing factors in automotive industry. Marketing events performed together with customers are gaining increasing importance in brand-management. Such events are intended to translate advertising messages into experienced reality (cf. Diez, 2001). Moreover, the events serve as a platform for interactive communication with the customers.

In automotive industry, the factory pick-up of new cars is increasingly organized as such a marketing event. Besides picking up their new cars, customers are also invited to visit the factory plant, to get familiarized with their new car, and to participate in further event-marketing activities. By combining various program items, each customer chooses in advance his or her individual delivery program. Often a customer is accompanied by family members or friends, which may also participate in the selected program items. The delivery program of some companies is such extensive that customers and accompanying people (henceafter called visitors) even have to pay for participating. Nevertheless, customers choose the factory pick-up more and more frequently.

In general, the visitors perform their delivery programs in groups. The visitors of a group participate simultaneously in the program items chosen. It may happen that a visitor has not selected the full program of the group. In this case, he or she will leave earlier if all program items chosen have been run. Otherwise, he or she will have to wait upon the start of the next program item.

For what follows we assume that visitors have already been assigned to groups. This assignment can, for example, be done based on earliest possible arrival times of the visitors, the program items chosen, or further criteria referring to customer segments. The planning problem under study consists in scheduling the group's program items in such a way that the sum of the visitors' waiting times is kept as small as possible. A small waiting time can be regarded as a key performance indicator of customer satisfaction.

For each of the program items, a minimum and a maximum duration are prescribed. Organizational requirements like the necessity to welcome a customer before starting any other program item give rise to temporal constraints. For executing a program item some staff like driving instructors and different kinds of facilities such as handover bays or a cinema are needed. A staff member can perform only one program item at a time. Moreover, according to the "one face to the customer" principle, all activities of a group must be managed by the same staff members. Certain facilities can be used by a given number of visitors at the same time, whose activities then sometimes need be synchronized (e.g., during a cinema show).

In this paper we show how to model and to plan the car delivery activities in the framework of resource-constrained project scheduling. For a review of recent developments in project scheduling we refer to the survey papers by Herroelen et al. (1998), Brucker et al. (1999), Kolisch (2001), Kolisch and Padman (2001), and Neumann et al. (2002b). An overview of state-of-the-art models, algorithms, and applications of resource-constrained project scheduling can be found in the books by Demeulemeester and Herroelen (2002) and Neumann et al. (2003). Each program

item performed by a customer group is interpreted as an activity of the project. The temporal constraints then correspond to minimum and maximum time lags between start or completion times of activities. Staff and facilities are modelled as renewable resources. Facilities like the cinema which require the simultaneous start and completion of overlapping activities correspond to synchronizing resources. The "one face to the customer" principle is taken into account by modelling the staff as allocatable resources. Synchronizing resources and allocatable resources are special renewable resources, which have been dealt with in the context of scheduling problems in the process industries (cf. Schwindt and Trautmann, 2003).

The basic idea of the solution method is to relax the resource constraints arising from the scarcity of facilities and manpower. The dual of the resulting temporal scheduling problem corresponds to a minimum-cost network flow problem that can be solved efficiently. In general the solution to this relaxation will cause conflicts on some resources. These conflicts are then resolved by introducing appropriate minimum time lags among the activities competing for the same resources. We have implemented this approach as a beam search heuristic enumerating alternative sets of time lags.

The remainder of this paper is organized as follows. In Section 2 we illustrate the problem under study using Porsche's car delivery process in Leipzig as an example. In Section 3 we model the planning problem as a resource-constrained project scheduling problem. Section 4 sketches the beam search planning heuristic. In Section 5 we demonstrate the efficiency of our algorithm on the basis of benchmark instances that have been provided by Porsche. Section 6 is devoted to concluding remarks and directions for further research.

2 Case study: Porsche Cayenne delivery program

In 2002, Porsche has launched the sport-utility vehicle series Cayenne. The Cayenne is the first series of Porsche cars that does not follow Porsche's pure sports cars tradition. According to Porsche's advertising, the Cayenne combines "apparent contradictions such as attractive design, advanced technology and exceptional on- and off-pavement performance". In particular the latter assertion is only rarely verified by the customers in the daily use of the car. Hence, a special emphasis of the delivery program has been placed on making customers aware of the car's capabilities.

Porsche has established a new manufacturing plant near Leipzig in Germany where the assembly of the Cayenne series takes place. Each customer picking up a new Cayenne in Leipzig may take some laps on an on-road and an off-road track under the guidance of a professional driving instructor. One special attraction of the on-road track, which has been approved by the Fédération Internationale de l'Automobile FIA, are the curves modelled on those of famous international racing circuits. The off-road track features eighteen different sections including a torsion track, a ramp crossing, a water ditch, a corduroy road, a boulder track, and a grazing paddock with cattle and wild horses. The main building of the site serves as a customer center. Inside there are a reception area, a cinema, a museum,

a restaurant, a control station for the on-road and off-road tracks, an accessories shop, and several bays for handing over the vehicles (cf. Porsche, 2002).

When registering for the delivery program, the customer is asked to communicate the number of accompanying people and to arrange the delivery program by combining up to 12 of the following program items:

1. a welcome in the reception area,
2. some formalities like an identity check or a payment verification,
3. a presentation of the customer center,
4. a film about the manufacturing of some parts shown in the cinema,
5. a tour around the assembly plant,
6. a visit of the accessories shop,
7. a tour of the museum,
8. catering in the restaurant,
9. a visit of the track control station,
10. a briefing with instructions for use of the car,
11. a training session on the on-road and off-road tracks, and
12. the final handing over of the car.

The welcome, the formalities, the briefing, the training session, and the handing over must be part of each customer's delivery program. All other items are optional. In order to create an exciting atmosphere and due to some organizational requirements, the visit of the track control station, the briefing, the training session, and the handing over have to be performed in that order. The same holds for the film and the plant tour. The formalities have to be carried out immediately after the welcome and before any other program item.

There is a given number of staff members and facilities available. For executing the delivery program items, two types of staff members are needed: Each group is accompanied by a customer advisor guiding the group through the entire program. In addition, each customer is assigned to a driving instructor being responsible for the last part of the program including briefing, training session, and handing over of the car. Several program items in addition require specific facilities:

- the formalities are carried out at a check-in desk,
- each visitor needs a seat in the cinema and a seat in the restaurant,
- a handover bay is required for the briefing and the handing over, and
- a Cayenne with the delivered motorization is used for the training session.

The car used for the training has to be cleaned immediately after the end of the training session. Immediately before the start of the briefing and of the handing over, the handover bay has to be prepared.

The scheduling problem under study consists in assigning a start and a completion time to each program item performed by a group such that

- the sum of the customers' waiting times is minimized,
- given precedence relationships between program items are respected,
- the durations of the program items are within the prescribed bounds,
- no staff member performs more than one program item at a time,
- each group is accompanied by the same customer advisor during the whole program,

– each customer performs items 10 to 12 with the same driving instructor,
– no more facilities than available are used at a time, and
– the delivery programs of all groups are completed within a working day.

Figure 1 shows an example run of the delivery programs of two groups. The shaded rectangles indicate requirements for facilities and driving instructors. Group 1 consists of three customers A, B, and C. We assume that customers A and B, who both pick up a Cayenne with "S"-motorization, have selected all 12 items of the delivery program. Customer C, who has ordered a Cayenne with "Turbo"-motorization, has not selected the visit of the shop and the catering. Group 2 consists of two customers D and E. In addition to the mandatory program items, both have chosen the catering and the visit of the track control station. Whereas customer D has bought a Cayenne "S", customer E picks up a Cayenne "Turbo". Figure 1 illustrates that customer C incurs waiting times between the end of the tour around the plant and the start of the museum tour and between the end of the museum tour and the start of the track control station visit. Note that this waiting times could be avoided if the visit of the shop and the catering were scheduled as the last activities of group 1.

3 Model

3.1 Basic concepts

For a given day, we model the delivery programs of all groups as a resource-constrained project. For each group of visitors, we introduce a project activity (an *activity*, for short) per program item that has been chosen by at least one customer of the group. Moreover, we introduce extra project activities modelling

– the preparation of the handover bay for the briefing,
– the cleaning of the car used for the training session, and
– the preparation of the handover bay for the handing over.

Finally, for modelling reasons we need a dummy activity of duration zero, which represents the completion of the last program item of the group.

For convenience we shall represent the project as a collection of *events*. Let A be the set of all activities and C be the set of customers under consideration. Each activity $i \in A$ gives rise to a start event $\sigma(i)$ and a completion event $\gamma(i)$. For each customer $c \in C$ we consider two events $\alpha(c)$ and $\omega(c)$ corresponding to the arrival and the departure, respectively, of customer c. Finally, we define events α and ω representing the beginning and the end of the working day. Then

$$V := \{\alpha, \omega\} \cup \{\alpha(c), \omega(c) \mid c \in C\} \cup \{\sigma(i), \gamma(i) \mid i \in A\}$$

is the set of all project events to be scheduled.

Now let $t_e \geq 0$ denote the time of occurrence of event $e \in V$. Then $t_{\sigma(i)}$ is the start time and $t_{\gamma(i)}$ is the completion time of activity $i \in A$. A vector

$$T = (t_e)_{e \in V}$$

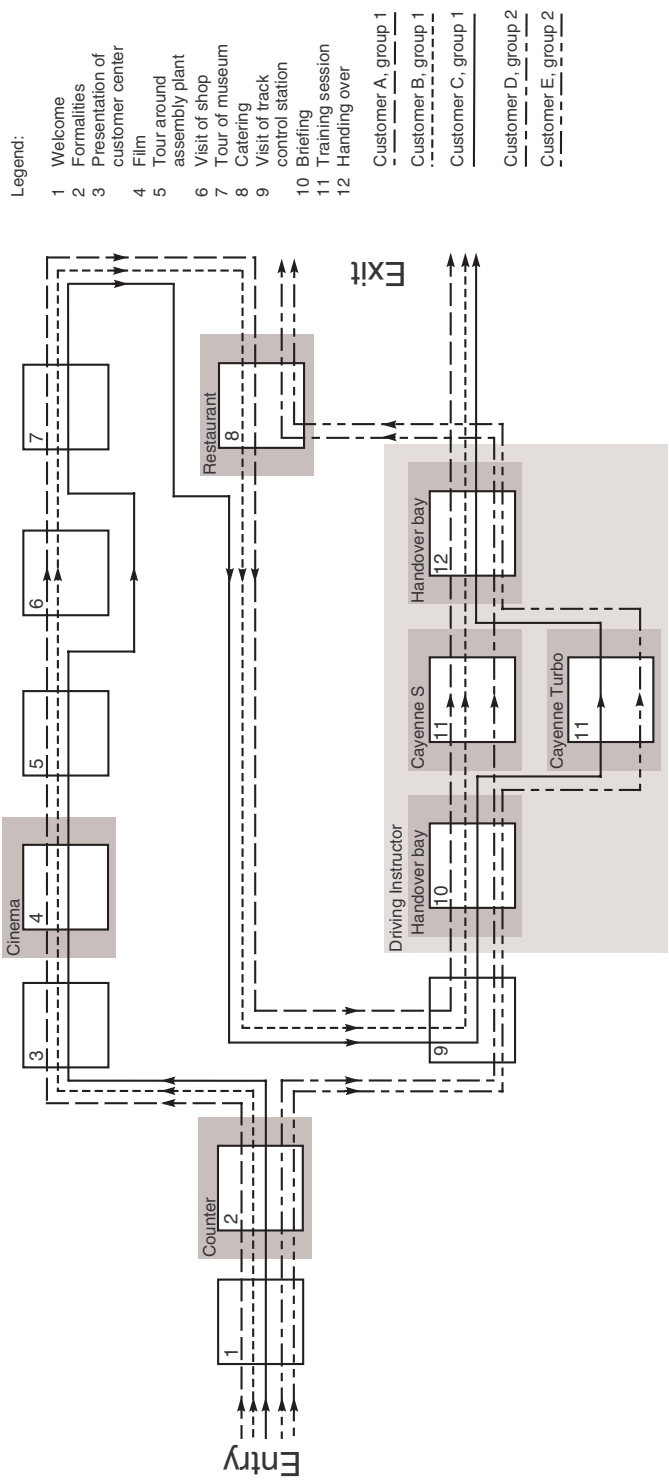

Fig. 1. Example run of two delivery programs

of occurrence times is termed a *schedule*. We assume the day to start at time zero, i.e., $t_\alpha := 0$. The objective function to be minimized is the sum of all waiting times

$$f(T) = \sum_{c \in C} \left(t_{\omega(c)} - t_{\alpha(c)} - \sum_{i \in A_c} \left(t_{\gamma(i)} - t_{\sigma(i)} \right) \right) \tag{1}$$

where $A_c \subset A$ designates the set of activities in which customer $c \in C$ is participating.

3.2 Temporal constraints

Between the occurrence times of two events $e, f \in V$ a minimum time lag $d_{ef}^{min} \geq 0$ or a maximum time lag $d_{ef}^{max} \geq 0$ may be given. In project scheduling, it is customary to represent the events and time lags in between by a network. To this end we assign the project events to the nodes of the network and for simplicity identify node e with event e ($e \in V$). If a minimum time lag d_{ef}^{min} between events $e, f \in V$ is prescribed, we introduce an arc $\langle e, f \rangle$ with weight $\delta_{ef} := d_{ef}^{min}$. A maximum time lag d_{ef}^{max} between events $e, f \in V$ corresponds to an arc $\langle f, e \rangle$ with weight $\delta_{fe} := -d_{ef}^{max}$. Let E denote the set of all arcs obtained in that way. The *project network* N consists of node set V, arc set E, and arc weights δ_{ef} for $\langle e, f \rangle \in E$.

A given schedule T is called *time-feasible* if it satisfies the temporal constraints

$$t_f \geq t_e + \delta_{ef} \quad (\langle e, f \rangle \in E) \tag{2}$$

Due to the occurrence of maximum time lags, network N generally contains directed cycles. It can be shown that a time-feasible schedule exists if and only if N does not contain any directed cycle of positive length (cf. Bartusch et al., 1988).

For our project, the following time lags have to be taken into account:

- The length $\overline{d}$ of the working day gives rise to the maximum time lag $d_{\alpha\omega}^{max} := \overline{d}$ between the beginning and the end of the day.
- The delivery programs of all groups have to be completed within the working day. Thus, for each customer $c \in C$ we introduce the minimum time lags $d_{\alpha\alpha(c)}^{min} := 0$ and $d_{\omega(c)\omega}^{min} := 0$.
- For each activity $i \in A$ a *minimum duration* $\underline{p}_i \geq 0$ and a *maximum duration* $\overline{p}_i \geq \underline{p}_i$ are given. The actual duration $t_{\gamma(i)} - t_{\sigma(i)}$ of activity i is subject to optimization and must be chosen within these bounds, i.e., $\underline{p}_i \leq t_{\gamma(i)} - t_{\sigma(i)} \leq \overline{p}_i$. This can be achieved by adding a minimum time lag $d_{\sigma(i)\gamma(i)}^{min} := \underline{p}_i$ and a maximum time lag $d_{\sigma(i)\gamma(i)}^{max} := \overline{p}_i$ between the start and completion events of activity i.

There exist some additional time lags among the activities of a group:

- Before a group can be welcomed, all customers of the group must have arrived at the site.
- The formalities must be carried out immediately after the welcome.
- The formalities have to be completed before any other program item can start.

- The film about the manufacturing has to be shown before the tour around the assembly plant is performed.
- The visit of the track control station, the briefing, the training session, and the handing over have to be executed in that order.
- The handover bay has to be prepared immediately before the start of the briefing and immediately before the start of the handing over.
- The car used for the training needs to be cleaned immediately after the end of the training session.
- A customer cannot leave the site before all program items that he or she has selected have been completed.
- The group's delivery program ends when all group members have left the site.

All these time lags can be modelled as minimum time lags $d_{ef}^{min} = 0$ between the start or completion events $e \in \{\sigma(i), \gamma(i)\}$, $f \in \{\sigma(j), \gamma(j)\}$ of the corresponding activities i and j. Figure 2 shows the part of the project network N which corresponds to the activities of customers A, B, and C of our example from Section 2. Activities $i = 1, \ldots, 12$ correspond to the 12 program items. Extra activities $i = 13, 14, 15$ correspond to the preparation of the handover bay for the briefing, the cleaning of the car used for the training session, and the preparation of the handover bay for the handing over, respectively. Activity 16 denotes the end of the last activity of the group.

3.3 Resource constraints

All facilities and staff members correspond to *renewable resources*, whose capacity is available at each point in time independently of their previous utilization (cf. Domschke and Drexl, 1991; Brucker et al., 1999). The set of renewable resources is denoted by $\mathcal{R}$. Each resource $k \in \mathcal{R}$ possesses a finite capacity $R_k \in \mathbb{Z}_{>0}$, which corresponds to the number of resource units available. An activity $i \in A$ takes up $r_{ik} \in \mathbb{Z}_{\geq 0}$ units of resource $k \in \mathcal{R}$ during its execution. $A_k := \{i \in A \mid r_{ik} > 0\}$ is the set of activities using resource k.

Given a schedule T, $\mathcal{A}(T,t) := \{i \in A \mid t_{\sigma(i)} \leq t < t_{\gamma(i)}\}$ is the set of activities being in progress at time $t \geq 0$ and $r_k(T,t) := \sum_{i \in \mathcal{A}(T,t)} r_{ik}$ is the total requirement for resource $k \in \mathcal{R}$ at that time. A schedule T is called *capacity-feasible* if for each renewable resource k, no more than R_k units of k are required simultaneously, i.e.,

$$r_k(T,t) \leq R_k \quad (k \in \mathcal{R};\ t \geq 0) \tag{3}$$

Notice that with respect to the temporal constraints, a customer may attend certain program items simultaneously, e.g., items 3, 4, 6, 7, 8, and 9 (compare Fig. 2). To prevent the overlapping execution of items being performed by one and the same customer, we introduce a renewable resource with capacity one for each group and assign a resource requirement of one to each of the group's items. In this way, we ensure that any capacity-feasible schedule will arrange the program items of a customer into some sequence, possibly with waiting times in between.

We model the check-in desks as a renewable resource whose capacity equals the number of counters available. The requirement for this resource by a formalities

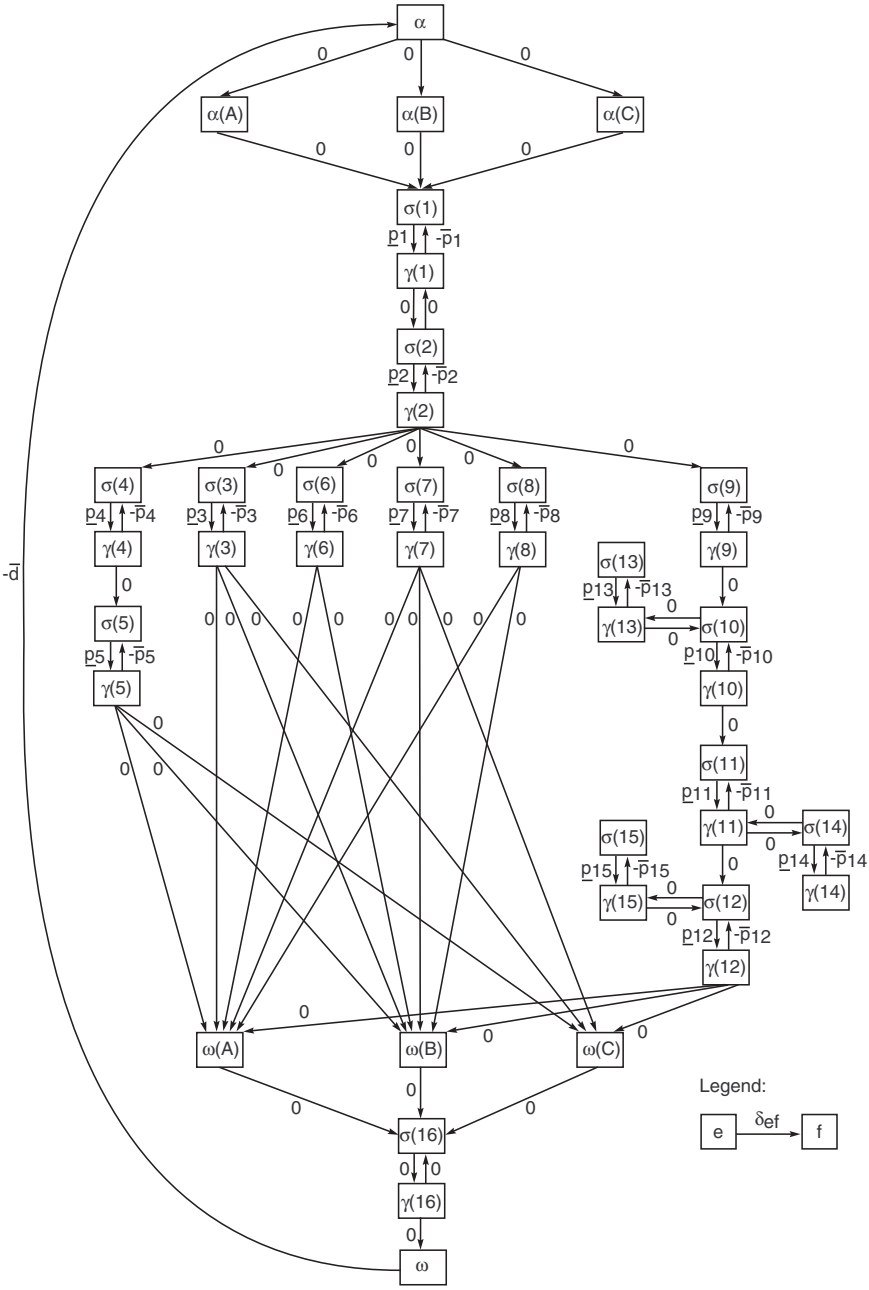

Fig. 2. Part of the example project network

activity equals one for each group. In the same way we define a renewable resource grouping the handover bays. The requirements for this resource by the briefing, handing-over, and corresponding preparation activities are equal to the number of customers in the group.

We combine the cars available for the training session to form two car pools differing in motorization. Each car pool corresponds to a renewable resource, whose capacity is chosen to be equal to the number of cars in the pool. The resource requirement by a training-session activity is given by the number of cars of respective motorization that are picked up by the group. The restaurant is modelled as a renewable resource as well. The capacity of this resource is equal to the number of seats in the restaurant, and the requirement by a catering activity coincides with the number of visitors in the group that have selected the catering program item.

Next, we turn to *allocatable resources* (cf. Schwindt and Trautmann, 2003), which are special renewable resources. The set of allocatable resources is denoted by $\mathcal{R}^\alpha \subseteq \mathcal{R}$. The r_{ik} units of resource $k \in \mathcal{R}^\alpha$ processing an activity $i \in A$ remain occupied from the start of a given allocating activity $a_k(i)$ starting no later than i up to the completion of activity i, which then releases the allocated resource units (see Fig. 3). For simplicity we assume that activity $a_k(i)$ itself does not use resource k, i.e., $r_{a_k(i)k} = 0$. The total requirement for an allocatable resource k at time t is $\overline{r}_k(T,t) := \sum_{i \in \mathcal{A}_k(T,t)} r_{ik}$ with $\mathcal{A}_k(T,t) := \{i \in A_k \mid t_{\sigma(a_k(i))} \leq t < t_{\gamma(i)}\}$. A schedule T is called *allocation-feasible* if at any point in time, no more than R_k units of a resource $k \in \mathcal{R}^\alpha$ are allocated, i.e.,

$$\overline{r}_k(T,t) \leq R_k \quad (k \in \mathcal{R}^\alpha;\ t \geq 0) \tag{4}$$

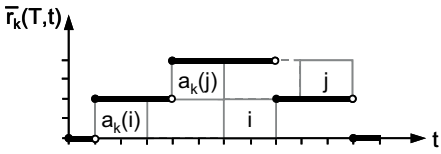

Fig. 3. Allocatable resource

Note that since $\overline{r}_k(T,t) \geq r_k(T,t)$ for all $k \in \mathcal{R}^\alpha$ and all $t \geq 0$, the allocation-feasibility of schedule T implies its capacity-feasibility with respect to all allocatable resources. The concept of allocatable resources allows us to integrate the "one face to the customer" principle into our model. Recall that as a consequence of this principle, a group is constantly accompanied by the same customer advisor or driving instructors. Accordingly, the customer advisors form an allocatable resource k whose capacity is equal to the number of customer advisors. One unit of this resource is to be allocated at the start of the welcome activity of each group and released after the group has performed all program items selected. This can be achieved by choosing, for each group, allocating activity $a_k(i)$ to be the welcome and releasing activity i to be the dummy program end activity of the group. Since one advisor is needed per group, we have $r_{ik} = 1$. The driving instructors are grouped into a second allocatable resource k', which is used by the briefing, training, and handing-over activities. Since for each group those activities must be

carried out in that order, we choose the briefing as the allocating activity $a_{k'}(i)$ and the handing over as the releasing activity i, the resource requirement $r_{ik'}$ being equal to the number of customers in the group.

Finally, we consider *synchronizing resources* (cf. Neumann et al., 2003, Sect. 2.13), which are special renewable resources as well. We denote the set of synchronizing resources by $\mathcal{R}^\sigma \subseteq \mathcal{R}$. A schedule T is called *synchronization-feasible* if activities being executed on some synchronizing resource in parallel are started at the same time, i.e.,

$$t_{\sigma(i)} = t_{\sigma(j)} \quad (i, j \in \mathcal{A}(T,t) \cap A_k \text{ with } k \in \mathcal{R}^\sigma \text{ and } t \geq 0) \tag{5}$$

We use a synchronizing resource k for modelling the cinema. Of course, for all groups watching the film jointly, the projection starts at the same time, or, to put it differently, the film activities have to be synchronized on resource k. The capacity R_k of resource k equals the number of seats in the cinema. The number r_{ik} of units taken up by the film activity i of a group corresponds to the number of visitors in the group that are going to watch the film.

In summary, Table 1 lists the resources, resource types, and activities requesting the individual resources for the Cayenne delivery program. Recall that allocatable and synchronizing resources are special renewable resources. Moreover, notice that aside from the group resources, each resource is used by customers of different groups. Hence, the resource constraints establish implicit dependencies between activities that are unrelated in the project network. Those dependencies form the combinatorial core of our scheduling problem.

Table 1. Resources, resource types, and requesting activities

Resource	Type	Requested by
Group	renewable	all program items of a group
Check-in desks	renewable	formalities
Handover bays	renewable	briefing, handing over, preparations
Car pool “Turbo”	renewable	training session
Car pool “S”	renewable	training session
Restaurant	renewable	catering
Advisors	allocatable	dummy program end
Instructors	allocatable	handing over
Cinema	synchronizing	film

3.4 Optimization problem

A schedule that is time-, capacity-, allocation-, and synchronization-feasible is termed *feasible*. An *optimal* schedule is a feasible schedule with minimum objective

function value. The scheduling problem (P) under consideration consists in finding an optimal schedule, i.e.,

$$\left.\begin{array}{l}\text{Minimize } f(T)\\ \text{subject to } (2) \text{ to } (5)\\ \qquad\qquad t_\alpha = 0\end{array}\right\} \tag{P}$$

Due to the continuity of objective function f and the compactness of the feasible region, problem (P) is solvable exactly if there exists a feasible schedule. The test for the existence of such a schedule, however, constitutes an NP-hard problem. The reason for this is that the length $\overline{d}$ of a working day imposes an upper bound on the project duration and that minimizing the duration of a project with scarce renewable resources is NP-hard (see e.g., Garey and Johnson, 1979, Sect. A5).

4 Solution procedure

In this section, we sketch a beam search planning heuristic for solving the factory pick-up scheduling problem presented in Section 3.

4.1 Basic principle

The heuristic procedure relies on a general relaxation-based approach to project scheduling with renewable resources (cf. Bell and Park, 1990; De Reyck and Herroelen, 1998; Franck et al., 2001). This approach has been adapted by Schwindt and Trautmann (2000) and Neumann et al. (2002a) to scheduling problems with different types of scarce resources. The intractability of problem (P) is due to the limited availability of the renewable resources and the simultaneous start condition for synchronizing resources. By relaxing the corresponding resource constraints (3), (4), and (5) we obtain a so-called temporal scheduling problem. Generally, the optimal solution to the temporal scheduling problem violates one or more resource constraints and thus is not feasible with respect to problem (P). In this case, a corresponding resource conflict can be removed by first, defining appropriate time lags between events of competing activities and second, re-performing temporal scheduling for the expanded project network which additionally contains the arcs belonging to the new time lags. Those two steps are iterated until either temporal scheduling yields a feasible schedule or the expanded project network contains a cycle of positive length. In the latter case, the corresponding temporal constraints are contradicting, and the temporal scheduling problem is unsolvable.

4.2 Temporal scheduling

Temporal scheduling is concerned with the problem of minimizing the objective function subject to the temporal constraints. In our case the temporal scheduling problem (TSP) reads as follows:

$$\left.\begin{array}{ll} \text{Minimize} & \displaystyle\sum_{c\in C}\Big(t_{\omega(c)} - t_{\alpha(c)} - \sum_{i\in A_c}\big(t_{\gamma(i)} - t_{\sigma(i)}\big)\Big) \\ \text{subject to} & t_f \geq t_e + \delta_{ef} \quad (\langle e, f\rangle \in E) \\ & t_\alpha = 0 \end{array}\right\} \quad \text{(TSP)}$$

Recall that a schedule satisfying the temporal constraints of (TSP) is called time-feasible. A time-feasible schedule with minimum objective function value is referred to as a time-optimal schedule.

In most project management applications, temporal scheduling amounts to computing the earliest or the latest occurrence times of the project events, which can be done efficiently by longest path calculations in the project network. In what follows we consider a small example showing that in our case, generally neither the earliest nor the latest schedule are time-optimal.

The example consists of two activities with a start and a completion event each and several temporal constraints. The first activity has a minimum (maximum) duration of 10 (20) and the second activity of 20 (30). Furthermore, the first activity has to be completed at time 20 at the latest and the second activity cannot be started before that time. The project must be completed by time 50. Figure 4 shows the associated project network. The earliest and latest schedules are $ET = (0, 0, 10, 20, 40, 40)$ and $LT = (0, 10, 20, 30, 50, 50)$. Both schedules yield an objective function value of 10 caused by a waiting time of 10 units of time between the occurrence of events 2 and 3. There exist, however, time-feasible schedules avoiding any waiting-time, e.g., schedule $T = (0, 10, 20, 20, 50, 50)$.

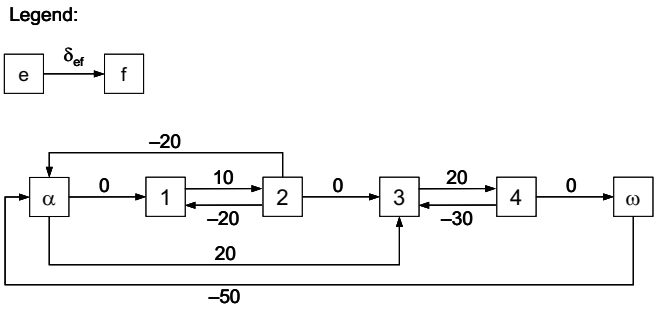

Fig. 4. Example project network

Our temporal scheduling procedure follows an approach by Russel (1970) to solving the project net present value problem (see also Schwindt, 2002, Sect. 3.2). It is based on a dualization of the temporal scheduling problem (TSP). For the moment we disregard the constraint $t_\alpha = 0$. The linear programm (TSP) then can be written as

$$\begin{aligned} &\text{Minimize} && c^\top T \\ &\text{subject to} && -\Delta(N)^\top T \geq \delta \end{aligned}$$

where $\Delta(N)$ denotes the incidence matrix of project network N, $T = (t_e)_{e \in V}$ is the vector of event occurrence times, and $\delta = (\delta_{ef})_{\langle e,f\rangle \in E}$ stands for the vector of arc weights. As the dual ($\overline{\text{TSP}}$) of the equivalent problem

$$\begin{aligned} &\text{Maximize} && -c^\top T \\ &\text{subject to} && \Delta(N)^\top T \leq -\delta \end{aligned}$$

we obtain

$$\left.\begin{aligned} &\text{Minimize} && -\delta^\top \phi \\ &\text{subject to} && \Delta(N)\phi = -c \\ & && \phi \in \mathbb{R}_{\geq 0}^{|E|} \end{aligned}\right\} \quad (\overline{\text{TSP}})$$

with $\phi = (\phi_{ef})_{\langle e,f\rangle \in E}$.

The latter problem represents a minimum-cost flow problem in project network N with unit costs $-\delta_{ef}$ and infinite upper flow bounds on arcs $\langle e, f\rangle \in E$ as well as supplies $-c_i$ at nodes $i \in V$. Such a network flow problem can be solved quite efficiently by polynomial-time cost-scaling algorithms, see e.g., Goldberg (1997). Based on a minimum-cost flow ϕ, a time-optimal schedule T can readily be constructed by exploiting the complementary slackness conditions, from which it follows that $T_f - T_e = \delta_{ef}$ for all $\langle e, f\rangle \in E$ with $\phi_{ef} > 0$.

Eventually we explain why, without loss of generality, we have cancelled equation $t_\alpha = 0$ in primal problem (TSP). The latter equation would give rise to an unrestricted variable in the equation

$$\sum_{\langle \alpha,i\rangle} \phi_{\alpha i} - \sum_{\langle i,\alpha\rangle} \phi_{i\alpha} = -c_\alpha \tag{6}$$

of ($\overline{\text{TSP}}$). Since this variable would not occur in the objective function, equation (6) could be eliminated from ($\overline{\text{TSP}}$). On the other hand, from $\sum_{i \in V} c_i = 0$ it follows that equation (6) is redundant because it is implied by the remaining equations of ($\overline{\text{TSP}}$), which is due to the singularity of incidence matrix $\Delta(N)$ (see, e.g., Murty, 1992, Sect. 1.2.2).

We illustrate this method by using the example introduced above. The flow network with a corresponding minimum-cost flow ϕ is depicted in Figure 5. Arcs $\langle e, f\rangle$ drawn in bold belong to flows $\phi_{ef} > 0$ and thus binding temporal constraints between events e and f in (TSP). One solution to the resulting equation system is schedule $T = (0, 10, 20, 20, 50, 50)$ with an objective function value of 0. Note that the schedule constructed need not be unique. More precisely, there exists more than one time-optimal schedule exactly if the minimum-cost flow is degenerate. For example, schedule $T' = (0, 5, 20, 20, 45, 45)$ is an optimal solution to our example as well.

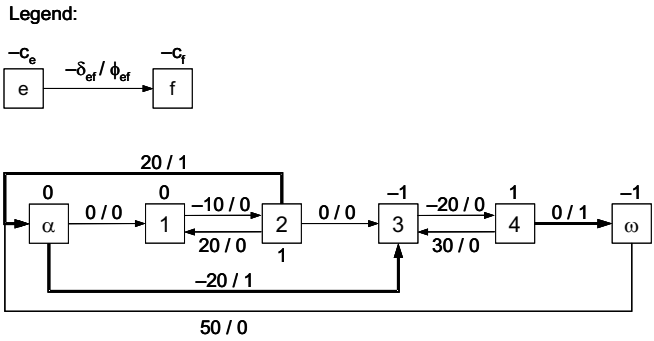

Fig. 5. Flow network

4.3 Resource-constrained scheduling

A time-optimal schedule found by the temporal scheduling procedure may be infeasible with respect to the resource-constrained problem (P) due to violations of the resource constraints (3), (4), or (5). Roughly speaking, we remove such conflicts by defining time lags between appropriate events, depending on the type of conflict. In general, a given conflict may be resolved in different ways, leading to time lags between different events. The most promising alternatives are enumerated within the beam search heuristic outlined in Subsection 4.4. To simplify the presentation we assume that the conflict is caused by two activities. The general case involving more than two activities can be treated by iteratively applying the techniques described in what follows.

At first, we consider *capacity conflicts*, i.e. violations of capacity constraint (3) for a renewable resource $k \in \mathcal{R}$ at some time $t \geq 0$. The conflict can be removed by selecting two activities i, j from set $\mathcal{A}(T,t) \cap A_k$ and defining time lag $d^{min}_{\gamma(i)\sigma(j)} = 0$ between the completion of i and the start of j. Figure 6 depicts a capacity conflict caused by two activities i and j on a renewable resource $k \in \mathcal{R}$ as well as the two alternatives of resolving the conflict.

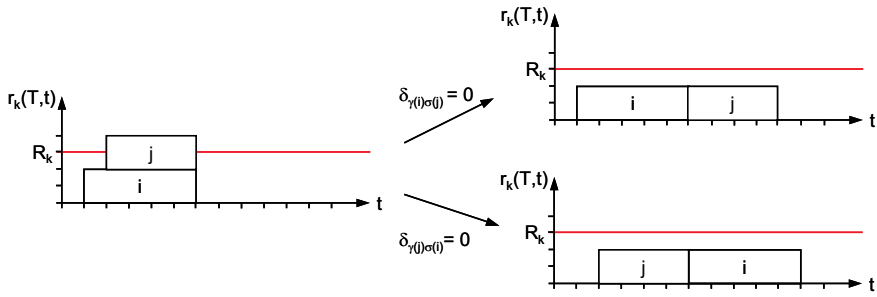

Fig. 6. Removing a capacity conflict

Next, we investigate the case of an *allocation conflict* referring to allocation constraint (4). Suppose that at some time $t \geq 0$, the number of allocated units of

an allocatable resource $k \in \mathcal{R}^{\alpha}$ exceeds the resource capacity R_k. Then we select two activities i, j from set $\mathcal{A}_k(T,t)$ and add time lag $d^{min}_{\gamma(i)\sigma(a_k(j))} = 0$ between the completion of i and the start of the allocating activity $a_k(j)$ of j. Figure 7 shows an allocation conflict on an allocatable resource $k \in \mathcal{R}^{\alpha}$ together with two alternatives of removing it.

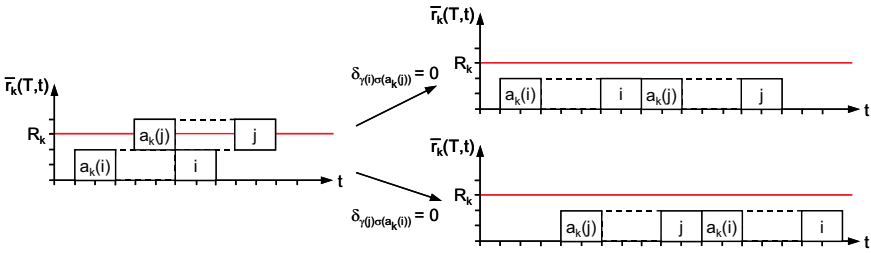

Fig. 7. Removing an allocation conflict

Finally, we move to *synchronization conflicts* occurring when the synchronization constraint (5) is violated. To remove a synchronization conflict for a synchronizing resource $k \in \mathcal{R}^{\sigma}$ at some time $t \geq 0$, we have either to ensure the simultaneous start or to avoid the overlapping of two activities i, j from set $\mathcal{A}(T,t) \cap A_k$ with $t_{\sigma(i)} \neq t_{\sigma(j)}$. Without loss of generality we assume that $t_{\sigma(i)} < t_{\sigma(j)}$. Then we must add either time lag $d^{min}_{\sigma(j)\sigma(i)} = 0$ (simultaneous start of i and j) or time lag $d^{min}_{\gamma(i)\sigma(j)} = 0$ (j starts after completion of i). These two alternatives are illustrated in Figure 8.

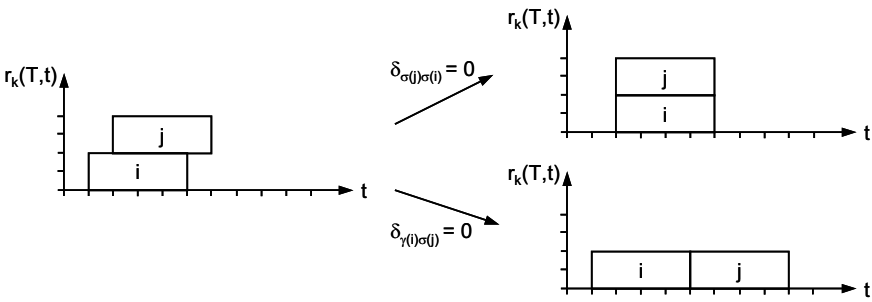

Fig. 8. Removing a synchronization conflict

4.4 Beam search heuristic

Using the temporal scheduling method from Section 4.2 and the techniques for removing resource conflicts discussed in Subsection 4.3 we are now ready to describe a general scheme for generating a feasible solution to problem (P) (see Fig. 9).

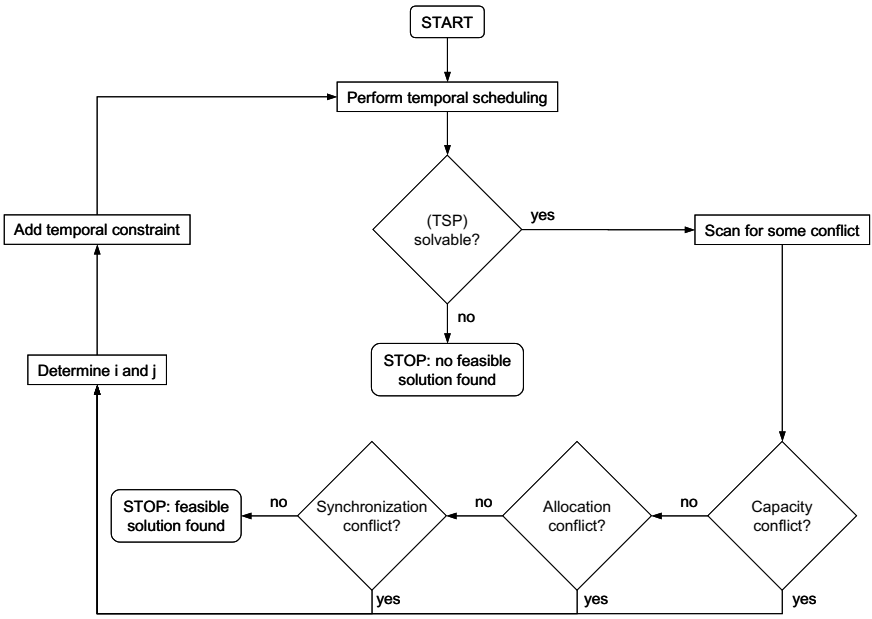

Fig. 9. General schedule-generation scheme

The procedure starts by performing the temporal scheduling step. If temporal scheduling problem (TSP) is not solvable, the minimum-cost flow algorithm detects a directed cycle of negative length in the flow network. This happens exactly if project network N contains a directed cycle of positive length. In the latter case, the schedule-generation scheme terminates without having found a feasible schedule. Otherwise, we scan the resulting time-optimal schedule for a capacity, an allocation, or a synchronization conflict. If no such conflict is found, the schedule is feasible and the procedure stops. Else, the detected conflict is treated by introducing a time lag between two activities i and j into problem (TSP), as explained in Section 4.3. The temporal scheduling and conflict resolution steps are iterated until the refined problem (TSP) becomes unsolvable or a feasible schedule is found.

Based on the above schedule-generation scheme we have implemented a depth-first search branch-and-bound algorithm enumerating alternative sets of time lags for removing resource conflicts. At each enumeration node, the lower bound on the objective function arises from solving the temporal scheduling problem for the expanded project network which contains the supplementary arcs belonging to the new time lags.

Due to the problem's hardness we have reduced the computational burden of our solution procedure by truncating the enumeration tree through beam search (see, e.g., Pinedo, 2001, Sect. 14.3; Schwindt and Trautmann, 2003; Kim et al., 2004). More precisely, we do not consider all alternatives of removing a given resource conflict, but only a given number (called the beam width) of promising candidates. The selection criterion is based on the objective function value (i.e., the lower bound, as primary criterion) and the makespan (as tie-breaker) of the schedule obtained by temporal scheduling.

As beam search is a heuristic procedure, it may happen that the algorithm does not find a feasible solution although there exists one. That is why we have integrated a so-called multi-start routine. If within a given number of iterations or a prescribed period of time no feasible solution has been found or the current best solution could not be improved, we restart the entire beam search from scratch. In order to enable the generation of previously unvisited enumeration nodes in the different passes we have randomly biased the node selection criterion by weighting the lower bound with a $(1, 1.1)$-uniformly distributed random factor.

5 Computational results

In this section we report on the results of an experimental performance analysis of the solution procedure presented in Section 4. We have considered three test sets A, B, and C. Test set A contains 54 practical test instances, which have been provided by Porsche, whereas test sets B and C have been generated by systematically varying the homogeneity of the delivery programs selected, the tightness of the resource constraints, and the problem size.

The instances from *test set A* can be characterized by the following parameters:

- the number of customers: 5, 10, 15, 20, 25, or 30, pooled into a minimal number of groups all but one group including three customers,
- the number of accompanying people per customer: 0, 1, or 3, and
- the number of selected program items per group: 6 (here: items 1, 2, and 9 to 12 only) or 12 (i.e., full program containing all items).

For the instances including six program items per group two scenarios with respect to the number of customer advisors have been investigated. In the first scenario there is one advisor available for each group, whereas in the second scenario the number of advisors equals half the number of groups, rounded up if necessary. For the remaining instances the number of advisors again equals the number of groups. The 54 instances have been generated by considering all combinations of the above parameters. The minimum and maximum durations of the program items in minutes and the capacities of the resources are listed in Tables 2 and 3, respectively.

Groups with one or two customers require one or two cars with "S" motorization, respectively. Groups containing three customers additionally take up one car with "Turbo" motorization. The length of a working day equals 660 minutes.

The tests have been performed on an AMD Athlon personal computer with 2.0 GHz clock pulse and 512 MB RAM using Microsoft Windows 2000 as operating system. The randomized beam search procedure has been implemented in C++ under MS-Visual Studio 6.0. The beam width has been chosen to be equal to 2. We have imposed a CPU time limit of 600 seconds per instance. As long as the time limit has not been reached and no provably optimal solution has been found, we restart the beam search from scratch each time when no improvement could be achieved within 5 seconds (for the small instances with less than 60 activities) or 20 seconds (for the large instances with 60 or more activities).

For all practical instances from test set A the beam search heuristic provides an optimal solution with zero total waiting time in the space of less than 4 seconds on

Table 2. Minimum and maximum durations

Activity i		Min. duration $\underline{p}_i$	Max. duration $\overline{p}_i$
1	Welcome	5	10
2	Formalities	10	10
3	Presentation of customer center	15	30
4	Film	15	15
5	Tour around assembly plant	45	60
6	Visit of shop	15	30
7	Tour of museum	30	60
8	Catering	60	60
9	Visit of track control station	15	30
10	Briefing	15	15
11	Training session	60	60
12	Handing over	15	15
13	Preparation of handover bay	5	5
14	Cleaning of training session car	10	10
15	Preparation of handover bay	5	5

Table 3. Resource capacities for practical instances

No.	Resource k	Capacity R_k
1	Check-in desks	1
2	Handover bays	6
3	Car pool "S"	8
4	Car pool "Turbo"	4
5	Restaurant	40
6	Customer advisors	≤ 10
7	Driving instructors	6
8	Cinema	54

the average. The maximum computation time equals 166 seconds. There are three instances for which the algorithm takes more than 30 seconds to find a first feasible solution. The main reason for the good performance seems to be the homogeneity of the groups. Since all customers of a group are run through the same delivery program, waiting times are only incurred in case of resource conflicts among different groups.

In test sets B and C we have considered the case where the customers of a group have chosen different programs, which independently of the resource availabilities typically leads to positive waiting times. Each group now contains three customers, two of which perform the full program. In addition to the mandatory program items, the third customer randomly selects certain of the optional items, the selection probability for an item being equal to 0.25, 0.5, or 0.75. Moreover,

we have systematically varied the resource capacities according to the scarcity scenarios displayed in Table 4.

Table 4. Resource capacities for the four scarcity scenarios

Resource no.	1	2	3	4	5	6	7	8
very tight	1	6	4	2	30	7	6	30
tight	1	6	5	3	35	8	6	35
loose	1	9	6	4	40	9	9	40
very loose	1	9	7	5	45	10	9	45

Test set B contains 10 instances with 30 customers for each combination of the three selection probabilities and the four scarcity scenarios. All 120 instances could be solved to feasibility. Tables 5 and 6 show the corresponding mean total waiting times and mean CPU times for finding the first feasible solutions.

Table 5. Mean total waiting times in minutes for test set B

	very tight	tight	loose	very loose
selection probability = 0.25	575.5	579.0	439.5	456.5
selection probability = 0.5	421.5	408.0	384.0	410.5
selection probability = 0.75	274.5	192.5	205.5	220.5

Table 6. Mean CPU times in seconds until first feasible solution for test set B

	very tight	tight	loose	very loose
selection probability = 0.25	181.8	51.6	22.3	34.6
selection probability = 0.5	214.8	28.3	32.9	35.0
selection probability = 0.75	225.9	24.7	19.0	19.0

The larger the selection probability the higher is the homogeneity of the delivery programs of individual customers within a group. From Table 5 it can be seen that with respect to waiting times, the effect of increasing homogeneity dominates the impact of the growing resource requirements coming along with a larger number of activities performed by the third customer. As expected, the waiting times are generally reduced as the resource availabilities increase. The computation times needed to find a first feasible solution, which are listed in Table 6, indicate that resource scarcity strongly influences the problem hardness, whereas the effect of the selection probability seems to be negligible.

Finally, we study the influence of the problem size on the solution quality and the computation times. *Test set C* has been obtained from test set B by selecting one instance for each of the 12 parameter combinations and choosing the number of customers to be equal to 18, 24, 30, or 36. The instances with 30 customers correspond to original instances from test set B, the remaining instances have been generated by deleting or adding groups. The results for the 48 instances from test set C are shown in Table 7. Due to the maximum project duration of one working day, for some of the large instances no feasible schedule could be found. The mean waiting and CPU times refer to the 43 instances with known feasible solutions.

Table 7. Mean total waiting times in minutes and mean CPU times in seconds until first feasible solution for test set C

# customers	18	24	30	36
# feasible solutions	12	12	12	7
mean total waiting time	106.3	219.2	405.8	574.3
mean CPU time first solution	3.8	4.5	72.3	156.9

The mean waiting times per customer increase from 5.9 minutes (18 customers) to 16.0 minutes (36 customers), which is mainly due to the fact that the resource scarcity augments as the workload to be processed is increased while fixing the maximum project duration. Moreover, the CPU time data indicate that in this case it also becomes more and more difficult to find feasible schedules (recall that the feasibility variant of our scheduling problem is already NP-hard). That is the reason why the practical resource availabilities as given in Table 3 have been sized in a way that allows for serving all customers within one working day without any fail. In conclusion, the performance analysis shows that even in case of tight resource constraints and heterogeneous groups the beam search heuristic provides schedules with moderate waiting times within a reasonable amount of time.

6 Conclusions

In this paper we have considered a scheduling problem arising in the factory pick-up of new cars. The execution of event-marketing activities has been modelled as a resource-constrained project with minimum and maximum time lags and nonregular objective function. Staff and facilities which are needed for the execution of the marketing event, correspond to renewable, allocatable, or synchronizing resources. The basic idea of the proposed solution procedure is to enumerate the alternative ways of replacing the resource constraints by additional minimum or maximum time lags between start or completion times of project activities. In an experimental performance analysis it has turned out that practical test instances can be solved to optimality within less than a minute of CPU time.

In practice, input data like customer arrival times or activity durations are often subject to change. As a consequence, the schedule may become infeasible during

implementation. The development of efficient methods for rescheduling in case of schedule disruptions is an important area of future research. Those methods should take into account alternatives for adapting the schedule as well as the use of additional resource capacities that can be made available in the short term.

References

Bartusch M, Möhring RH, Radermacher FJ (1988) Scheduling project networks with resource constraints and time windows. Annals of Operations Research 16: 201–240

Bell CE, Park K (1990) Solving resource-constrained scheduling problems by A* search. Naval Research Logistics 37: 61–84

Brucker P, Drexl A, Möhring RH, Neumann K, Pesch E (1999) Resource-constrained project scheduling: notation, classification, models, and methods. European Journal of Operational Research 112: 3–41

Demeulemeester EL, Herroelen WS (2002) Project scheduling: a research handbook. Kluwer Academic Publishers, Boston

De Reyck B, Herroelen WS (1998) A branch-and-bound procedure for the resource-constrained project scheduling problem with generalized precedence relations. European Journal of Operational Research 111: 152–174

Diez W (2001) Automobil-Marketing. Moderne Industrie, Landsberg/Lech

Domschke W, Drexl A (1991) Kapazitätsplanung in Netzwerken. OR Spektrum 13: 63–76

Franck B, Neumann K, Schwindt C (2001) Truncated branch-and-bound, schedule-construction, and schedule-improvement procedures for resource-constrained project scheduling. OR Spektrum 23: 297–324

Garey MR, Johnson DS (1979) Computers and intractibility: a guide to the theory of NP-completeness. Freeman, New York

Goldberg AV (1997) An efficient implementation of a scaling minimum-cost flow algorithm. Journal of Algorithms 22: 1–29

Herroelen WS, De Reyck B, Demeulemeester EL (1998) Resource-constrained project scheduling: a survey of recent developments. Computers and Operations Research 25: 279–302

Kim KH, Kang JS, Ryu, KR (2004) A search algorithm for the load sequencing of outbound containers in port container terminals. OR Spectrum 26: 93–116

Kolisch R (2001) Entwicklungen und Anwendungen in der Projektplanung: Ein Überblick. Betriebswirtschaftliche Forschung und Praxis 2001: 212–226

Kolisch R, Padman R (2001) An integrated survey of project scheduling. Omega 29: 249–272

Meyr H (2004) Suppy chain planning in the German automotive industry. OR Spectrum 26

Murty, KG (1992) Network programming. Prentice Hall, Upper Saddle River

Neumann K, Schwindt C, Trautmann N (2002a) Advanced production scheduling for batch plants in process industries. OR Spectrum 24: 251–279

Neumann K, Schwindt C, Zimmermann J (2002b) Recent results on resource-constrained project scheduling with time windows: models, solution methods, and applications. Central European Journal of Operations Research 10: 113–148

Neumann K, Schwindt C, Zimmermann J (2003) Project scheduling with time windows and scarce resources. Springer, Berlin Heidelberg New York

Pinedo M (2001) Scheduling: theory, algorithms, and systems. Prentice Hall, Englewood Cliffs

Porsche (2002) Porsche Club News May 2002, 7–9. Porsche, Leipzig

Russel A (1970) Cash flows in networks. Management Science 16: 357–373

Schwindt C (2002) Introduction to resource allocation problems in project management. Habilitation thesis, University of Karlsruhe

Schwindt C, Trautmann N (2000) Batch scheduling in process industries: an application of resource-constrained project scheduling. OR Spektrum 22: 501–524

Schwindt C, Trautmann N (2003) Scheduling the production of rolling ingots: industrial context, model, and solution method. International Transactions in Operational Research 10: 547–563

VDA (2003) Auto Jahresbericht 2003. Verband der Automobilindustrie, Frankfurt am Main