Konrad-Zuse-Zentrum
für Informationstechnik Berlin

Takustraße 7
D-14195 Berlin-Dahlem
Germany

SVEN O. KRUMKE [1] JÖRG RAMBAU LUIS M. TORRES [2]

Online-Dispatching of Automobile Service Units [3]

[1] Research supported by the German Science Foundation (DFG, grant Gr 883/10)
[2] Research supported by the German Academic Exchange Service (DAAD, grant A/99/03594)
[3] to appear in: Operations Research Proceedings 2002

ZIB-Report 02-44 (November 2002)

Online-Dispatching of Automobile Service Units

Sven O. Krumke, Jörg Rambau, and Luis M. Torres

Abstract. We present an online algorithm for a real-world vehicle dispatching problem at ADAC, the German Automobile Association.

1 Problem Description

The German Automobile Association *ADAC* (*Allgemeiner Deutscher Automobil-Club*), the second largest such organization in the world, surpassed only by the American Automobile Association, maintains a heterogeneous fleet of over 1,600 service vehicles in order to help people whose cars break down on the road. All ADAC service vehicles (called *units* in the following, for short) are equipped with a GPS system which allows to locate their positions at any time. In five help centers (*Pannenhilfezentralen*) distributed throughout Germany, human dispatchers have to reply to incoming help requests (*events*) instantly. Their task is to assign a unit to serve each customer and to predict the estimated time of arrival at the customer's location. In addition to the ADAC fleet, about 5,000 units operated by service *contractors* can be employed to cover events that otherwise could not be served in time.

There is no unique objective. The goals are high-quality service (e.g., short waiting times for the customers) and low operational costs (e.g., short tour lengths and small overtime costs). With increasing costs and request volume, ADAC's dispatching system has come under stress. The task is to design an automatic system that guarantees small waiting times for events and keeps operational costs low. Such a system must address *two* different issues:

Realtime-Problem Given a "snapshot" of the situation at some moment in time, compute an optimal dispatch for attending all pending events with the available units and (if required) contractors. Since it contains the classic vehicle-routing problem as a special case, this problem is NP-hard. The difficulty is that such a dispatch has to be returned in a very short time, usually no more than 15 seconds for a system load of about 200 events and 100 units.

Online-Problem Once an algorithm for the first task has been found, design a good strategy for embedding it within the constantly running dynamic planning process, i.e., decide how often a new dispatch should be computed in response to incoming events, and/or how far changes should be admitted in a previous computed dispatch. The main challenge lies in the impossibility to predict if, where and when events in the near future will take place.

The modeling of the first task, which we shall in the following call the *vehicle dispatching problem* VDP, involves many technical and organizational side constraints—some hard, others soft—, and it takes some time to figure out which restrictions and objectives really count. As one example, we have considered constraints arising from a management decision: ADAC's imposition of a soft deadline on the service time of an event, which may be missed at the cost of a linearly increasing lateness penalty (*soft time windows*).

Using an approach based on column generation, it was possible to design an optimization algorithm for the VDP capable of finding optimal or near-optimal solutions for real-world instances and which was compliant with the real-time requirements of the problem. In section 3, we give a brief description of the algorithm (see [5] for the details). The main purpose of this paper is to present our first results concerning the *online-problem*. In section 2, we describe the criteria we used for evaluating different possible online solution schemes and introduce the strategy that we have chosen. Experimental competitive results for the application of this strategy on real-world instances are reported in section 4. Section 5 summarizes the key points of this paper.

2 Online Strategy

We postpone until the next section the discussion about *how* the VDP instances corresponding to snapshot situations are solved and focus first on the more subtle online-problem: the search for a strategy to carry out the actual planning without knowing where future events will pop up. A decision that is "optimal" at some point in time can prove later to have been unwise. In particular, even if we were able to compute locally optimal dispatches for any snapshot situation this does *not* mean that we obtain a dispatch which is (in hindsight) optimal for the whole planning period.

A by now standard tool for measuring the "goodness" of an online algorithm is *competitive analysis* [6]. Basically, one compares the solution provided by the algorithm for an instance of the online problem with the solution provided by a "hindsight" adversary, which solves the same problem instance but with knowledge of the whole input data in advance. It is usually almost impossible to obtain theoretical proofs of (useful) competitive results, except for elementary problems. Nevertheless, the concepts arising from competitive analysis can still be used in practice, in the form of experimental a-posteriori analysis of online strategies (see, e.g., [2] for some real-world examples), provided a measured hindsight adversary can be found.

In our case, this a-posteriori analysis was carried out over input data collected during one (resp. two) hour(s) of operation of the current system at ADAC. This data was fed to our algorithm in a way that simulated how it would occur in practice: each event was labeled by a time-stamp that indicated the moment at which it became known to the system. Only after that time the algorithm was allowed to incorporate information from this event in a solution. After having briefly considered several simple online heuristics, we took the decision to follow a REPLAN strategy.

This strategy consists of recomputing from scratch, at certain moments in time, a completely new dispatch for the current VDP snapshot (the so-called *offline* problem). REPLAN assumes that an algorithm for this offline problem is at hand, which is capable of delivering solutions with certain guaranteed quality under real-time conditions (see [4] for more details). Fortunately, this is the case for the instances arising at ADAC, as we shall see in the next section.

3 Auxiliary Optimization Problem

In the following, we briefly specify the form of VDP that is tackled by our algorithm. More details concerning both the model as the solution method can be found in [5].

An instance of the VDP consists of a set of units, a set of contractors, and a set of events. Each unit u has a current position o_u, a home position d_u, a logon time t_u^{start}, a shift end time t_u^{end}, and a set of capabilities F_u. Moreover, the costs related to using this unit are specified by values for costs per time unit for each of the following actions: driving c_u^{drv}, serving c_u^{svc}, and overtime c_u^{ot}. Each contractor v has a home position d_v and a set of capabilities F_v. The costs for booking the contractor are specified by a fixed value per service c_v^{svc}. Each event e has a position a_e, a release time θ_e^{r}, a deadline θ_e^{d}, a service time δ_e, and a set of required capabilities F_e. Moreover, missing the deadline of an event means incurring in a lateness cost equal to the delay times the value of a lateness coefficient c_e^{late}.

A feasible solution of the VDP (a *dispatch*) is an assignment of events to units and contractors capable of serving them, as well as a tour for each unit such that all events are assigned, the service of events does not start before their release times (waiting of units is allowed at no extra-costs), and all tours for all units start at their current positions not before their logon times and end at their home positions. The costs of a dispatch are the sum of all unit costs, contractor costs, and event costs.

Following a common approach in the vehicle routing literature (see, e.g., [1] and the references therein), we state our model using binary tour variables. The VDP can then be formulated as a set partitioning problem, where the ground set of events and units has to be partitioned using a family of subsets that represent unit tours (plus some special subsets to account for the alternative of assigning events to contractors). The advantage of such a model lies in its flexibility to incorporate complicated technical and organizational side constraints within the individual tours and the possibility to tweak the cost function to achieve certain desired properties of the online behaviour (e.g., non-linear lateness penalty). This set partitioning model may be written as a huge integer linear program where the 0/1 restriction matrix contains one row for each event and unit and one column for each tour. To solve this problem, we use a column generation approach. Starting from some initial columns produced heuristically, the main iteration of our algorithm consists of adding new columns and resolving the linear relaxation of the problem until a certain stopping criterion is met. Whenever a new integral solution is found we output the corresponding dispatch.

The search for columns is done by enumeration in a *depth-first-search branch&bound tree* (*search tree*, for short) for each unit. To prune the search tree, a lower bound on the cost of a tour is used, which is based on the dual prices of events and units obtained from the previous solution of the linear problem. Our main contribution lies in what we call *dynamic pricing control*. The depth and degree of the search tree, as well as the value of a (negative) acceptance threshold imposed on the reduced cost of new columns are adjusted at each iteration according to the number of columns produced in the previous one. This ensures that (i) the effort of finding new columns is small in the beginning, when the dual variables are not yet in good shape, (ii) the dual variables are updated often in the beginning, (iii) this update is fast since the number of columns in the LP is still small, (iv) we can enforce the output of a feasible integer solution early, and (v) the search is exact later in the run when the dual information is reliable.

Our algorithm ZIBDIP turned out to be very efficient on real-world snapshot-instances provided by ADAC. In all tested cases, provably optimal or near-optimal (<1%) solutions were found in less than five seconds on state-of-the-art personal computers, even for high load situations containing about 200 events and 100 units. This figures ensure compliance with the real-time requirements of the application. The behavior of the algorithm remained stable for problem instances with (artificially augmented) extreme load: ZIBDIP found a dispatch to attend 770 events with 200 units whose quality was within 12% from optimum after 5 seconds, within 5% from optimum after 15 seconds, and within 2% after one minute.

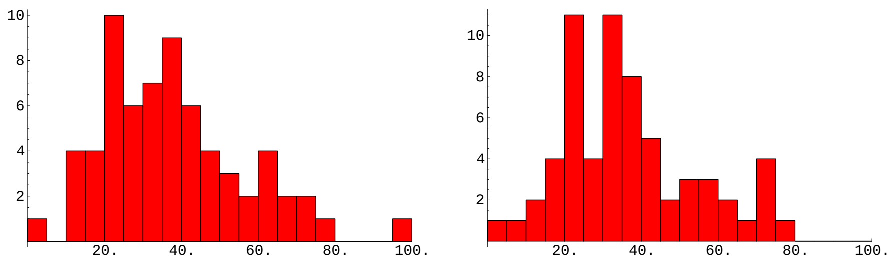
Besides of solving the real-time "snapshot" problems, our algorithm is also used for the evaluation of online strategies: by running it a-posteriori on jobs collected during one/two hours, we found lower bounds on the value of the hindsight-optimum discussed in section 2.

4 Computational Results on Real-World-Data

The input data for the online-tests consisted of 68 *one-hour instances* and 68 *two-hour instances* that were extracted from accumulated whole-day datasets provided by ADAC. The events occurring in these instances were labeled with time-stamps that represent the moment in time when they pop up. Using this information, a simulation was run on the ADOptCmd implementation of our algorithm to test six variants of the REPLAN strategy mentioned in section 2. The total cost of the dispatch produced during the whole hour (resp. two hours) was compared to a lower bound on the hindsight optimum, which was obtained by solving to optimality the linear programming relaxation of the offline problem (i.e., the problem of finding an optimal dispatch if all events are known from the beginning). In all cases, the online algorithms were required to deliver their (partial) solutions within 15 seconds of running time.

Both for the one-hour as for the two-hours instances, the relative error of the online solution achieved by ADOptCmd for all six alternatives is well below 50% on the average with very rare substantial deviations of up to 230.71%, obtained for the setting `int120` (replan every 120 seconds) in one data set. There is a difference in the performances of the six different settings in favour of `newjob` (replan with every occurrence of a new job) and `int60` (replan every 60 seconds). This difference is, however, small enough to ensure that there will be no serious performance problems in waiting say 60 seconds for an optimization run to finish.

Figure 1 shows the distribution of the relative errors obtained for the best two settings `newjob` and `int60` on the one-hour instances. Setting `newjob` achieves the best results on average (a relative error of 40.21% against 44.33% for `int60`) and produces the least deviation (22.65 against 33.64). This trend continues in the two-hour instances.



(a) `newjob`: $\mu = 40.21$, $\sigma = 22.65$ (b) `int60`: $\mu = 44.33$, $\sigma = 33.64$

Fig. 1. Relative errors in the solutions produced for the one-hour instances.

Another fact to be noticed is that there is no substantial degradation in the performance of the algorithm when switching from the one-hour to the two-hours instances. In fact, the relative error obtained by `newjob` was slightly better in the second case. We expect the same to hold also for larger intervals of time. The computation of the required lower bounds, however, becomes then technically unmanageable.

5 Conclusion

We have developed a specialized column generation algorithm ZIBDIP that solves a real-world large scale vehicle dispatching problem with soft time windows under realtime requirements. The problem arises as a subproblem in an online-dispatching task that was proposed to us by the German Automobile Association (ADAC). A key concept behind the algorithm is the Dynamic Pricing Control, which can significantly speed up convergence of the column generation process, thereby making a method that has proven to be effective for large scale offline problems ready for the use in online-algorithms under realtime requirements. A further advantage of the column generation approach is its flexibility to incorporate complicated restrictions: we are planning to use non-linear lateness penalties in future tests.

Employing a-posteriori analysis on real-world problem instances provided by ADAC, we tested the performance of several settings of the REPLAN strategy (i.e., we determined a kind of *experimental competitivity*). The best results were obtained for the case when a new dispatch was computed either each time a new event was issued or at a fixed frequency of 60 seconds.

Although the development of the final version of the online algorithm is still under way, the first results are promising: On average, the online costs are within 50% above a lower bound on the "hindsight" (offline) adversary—not too bad, in our experience, for an online algorithm. We hope, however, to improve on these figures by utilizing estimates of future events in the snapshot-dispatches. This is work in progress and leads to most interesting questions as to how knowledge about the future can be exploited in combinatorial online optimization.

References

1. Jacques Desrosiers, Yvan Dumas, Marius M. Solomon, and Francois Soumis, *Time constrained routing and scheduling*, Handbooks in Operations Research and Management Science, vol. 8, Elsevier Science B.V., Amsterdam, 1995, pp. 35–139.
2. Martin Grötschel, Sven O. Krumke, and Jörg Rambau, *Online optimization of complex transportation systems*, in *Online Optimization of Large Scale Systems* [3], pp. 714–739.
3. Martin Grötschel, Sven O. Krumke, and Jörg Rambau (eds.), *Online optimization of large scale systems*, Springer Verlag, 2001.
4. Martin Grötschel, Sven O. Krumke, Jörg Rambau, Thomas Winter, and Uwe Zimmermann, *Combinatorial online optimization in real time*, in Grötschel et al. [3], pp. 687–713.
5. Sven O. Krumke, Jörg Rambau, and Luis M. Torres, *Real-time dispatching of guided and unguided automobile service units with soft time windows*, Preprint SC 01-22, Konrad-Zuse-Zentrum Berlin, Berlin, 2001.
6. D.D. Sleator and R.E. Tarjan, *Amortized efficiency of list update and paging rules*, Communications of ACM **28** (1985), 202–208.