Chapter 1

Introducing the ADAC-Problem

The German Automobile Association *ADAC* (*Allgemeiner Deutscher Automobil-Club*[1]) – 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. Due to their color and the fact that they often bring needed help, people call these service vehicles affectively “yellow angels”. In the sequel, we simply refer to them as *units*, for short. Figure 1.1 shows a picture of them. Every unit is equipped with more than 300 tools and a GPS system which allows to precisely locate its position at any time. Incoming help calls from anywhere in Germany are transferred to one of five help centers, where human dispatchers process them. Their task is to assign to each customer a unit capable of handling his *request*, and to predict the unit’s 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 requests that otherwise could not be served in time.

Figure 1.1: The *yellow angels* fleet from ADAC. Every service unit is equipped with more that 300 tools which together weigh about 280 kg. The cumulative distance traveled by all units in one year exceeds 56 million kilometers. (Taken from ADAC’s press office).

[1] Am Westpark 8, 81373 München, Germany, `http://www.adac.de`

Due to increasing operational costs and request volume, this dispatching system has come under stress. As part of a cooperation project between ADAC, the *Konrad-Zuse Zentrum für Informationstechnik Berlin*[2] (ZIB) and the software development company *Intergraph Public Safety*[3] (IPS), we have been working on the design of an automatic online dispatching system to relieve the human operators and to reduce costs while ensuring a specified level of service quality (measured for instance in the average wait times for the customers). Additionally, we seek to exploit the new optimization potential that arises when schedules for large areas are computed globally. (To keep the problem instances at sizes that are manageable for human dispatchers, the approach that has been taken before is to divide the geographic region covered by a help center into many small sectors).

A Working Day at ADAC

The dispatching problem at ADAC – which we simply call the *ADAC-Problem* from now on – is an *online optimization problem*: we are looking for a reasonable strategy to control a system which is dynamically changing. To explain this in more detail, let us follow a dispatcher at one of the ADAC's help centers through his typical working day. In Figure 1.2 we can see what his workplace looks like. (Actually, several dispatchers will take turns at one of these places in the course of a day).

Figure 1.2: An ADAC dispatcher at work. The computer system registers via GPS the exact position of the service units at any time. Information about emerging help requests is received by phone from the call centers. Orders are transmitted to the units over radio. (Taken from ADAC's press office).

We start early in the morning, say at 2 a.m. For simplicity, we assume no help requests have been reported yet and only a few units are currently active. Some thirty minutes later, the first call of the day is received. Somewhere in Berlin an unlucky driver, who wants to return home after a long party-night, cannot get his auto started. On the phone, the dispatcher gets information about the request: its

[2] Takustr. 7, 14195 Berlin, Germany, `http://www.zib.de`

[3] Huntsville, AL 35894 USA, `http://www.intergraph.com`

geographical position, the kind of failure, etc. Then he checks the available units that are technically equipped for attending this request. Depending on the type of the failure, the service might require application of specialized tools. (In the worst-case, it might be necessary to haul the damaged car to a garage). Fortunately for our driver, this is not the case, since every unit is capable of providing start help "on-site". The dispatcher assigns the request to a unit, estimates the arrival time at the customer's position, and tells him when he can expect the "yellow angel" to be there.

After that, nothing happens for the next half an hour. Then the computer system informs the dispatcher a new unit has logged in and is ready for taking orders. Every unit is equipped with a GPS, which allows the computer to automatically track its precise position at any time. Now the dispatcher is aware that he can include this unit in future plannings. A couple of minutes later, the unit assigned to the first request calls and reports it has successfully finished its assigned task. As the day approaches, more and more units log in, and at the same time the units who have worked in the night-shift log off and return to their depots.

It is about 7 a.m. People are on their way to work, the streets get crowded, the traffic increases – and with it the number of help requests. Every time, the cycle is the same: the call is taken, information about the request is collected, a new service dispatch is computed to assign the request to an adequate unit, the arrival time of the unit at the customer's site is predicted, and the customer is informed. What varies is the size (and hence the complexity) of the successive assignment problems that the dispatcher has to solve.

Between 9 a.m. and 4 p.m., the system is working at its maximum load. On some days, there may be situations where as many as 300 requests are waiting for service, and about 80 units are active; with new requests arising at a rate of about 4 per minute. Figure 1.3 (elaborated from recorded real-world data) shows how the picture typically looks like in practice. The small crosses indicate the positions of ADAC's units while the small circles are used to illustrate the positions of waiting requests. The figure contains 40 units and 195 requests, and yet it only shows a partial view of the whole situation: at that time, there were in total 364 waiting requests and 82 units distributed over an area covering approximately one fifth of Germany. Under this conditions, the task of computing a service dispatch gets very complicated. And yet it has to be done quickly – in less than 5 seconds, according to ADAC's specifications.

Two additional aspects increase the palette of options a dispatcher has to take into account. At first, there is the possibility to hire service contractors, which can take those requests that cannot be handled in time by the ADAC's own units. Secondly, it might be convenient to reassign requests: Suppose, for instance, that a unit u has been assigned a request r_1. While u is on the way to getting there, a second request r_2 arises very close to the unit's current location. Now the dispatcher has to consider the possibility of changing orders and telling u to serve r_2 first. This is called a *preemption*, and we shall say more about it in Chapter 7.

Every time a dispatch is computed, a *cost function* has to be minimized, which

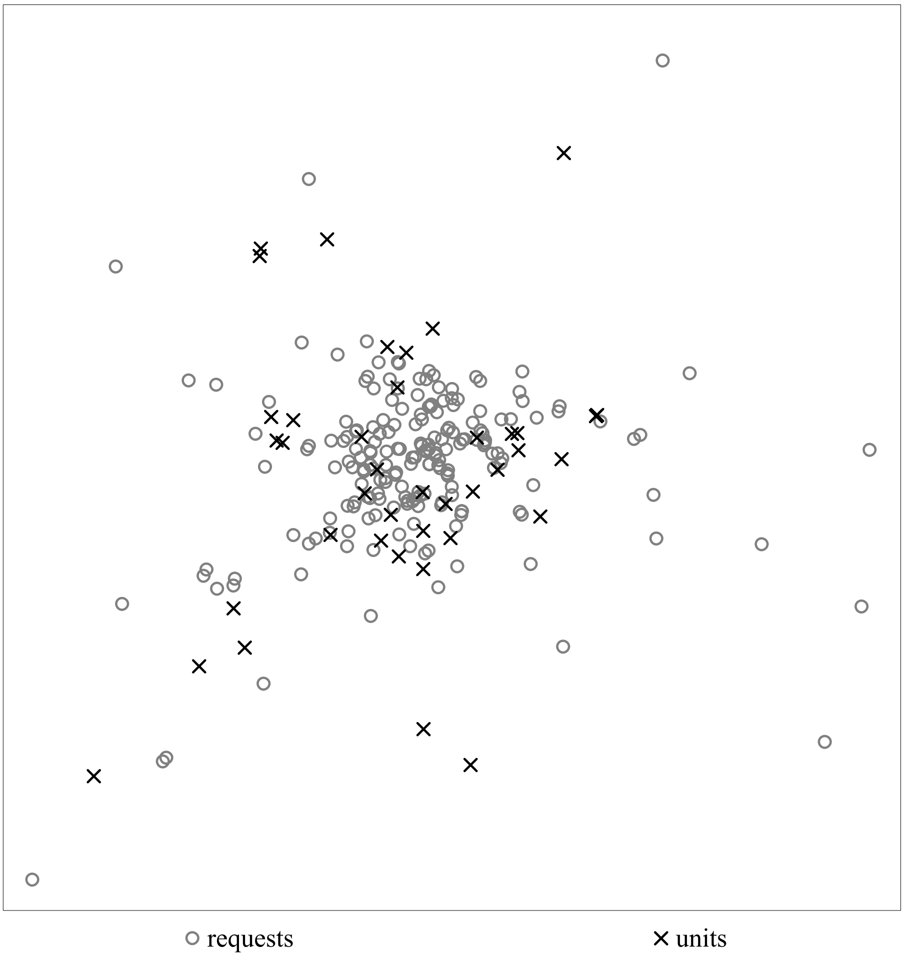

Figure 1.3: Partial view of a real-world snapshot in the dispatching process at ADAC. Crosses indicate the geographical positions of units, small circles the positions of help requests. The complete snapshot consisted of 364 requests and 82 units.

involves several components and is used as a measure of both the "real" operational costs of the system and some "fictive" costs that account for the quality of the service. (Again, a precise definition will be provided in Chapter 7, after introducing more background information on the problem).

After 7 p.m., the number of requests will eventually start to decrease. At some time, units that have been active during the day will finish their shifts and log off, and the units for the night-shift will log on again. Our working day at the ADAC's dispatching center ends here.

A first immediate observation when we look at the ADAC-Problem is that we are dealing with a *discrete online problem*. To steer the system, we do not need to take actions continuously over time, but only whenever its state changes; and this can happen only as a consequence of one of the following *events*:

- a new request emerges,
- a unit finishes service of a request,
- a unit logs in,
- a unit finishes its shift and logs off.

Hence, we may model the dynamic planning process as a collection of *snapshots* in time. Every snapshot specifies a set of units, a set of requests, and a set of contractors; together with additional information concerning for instance their positions, their technical compatibilities (which requests may be assigned to which units/contractors), the time every request has been waiting in the system, etc. Our task is to compute on the basis of this information a minimum cost feasible service dispatch. Similar *vehicle routing* (or *vehicle scheduling*) problems constitute a stand-alone topic of investigation in Operations Research. In this case, we are dealing with a *real-time* routing problem, where the available time for finding a solution is strongly constrained. For the rest of this thesis, we shall use the term *Vehicle Dispatching Problem* (VDP) to refer to any of these snapshot problems. We will discuss an example in a moment.

A second issue to be addressed regards the embedding of a solution algorithm for the VDP into an *online strategy*. It might be a bad idea to change the schedule of the whole system at every new snapshot. Even if we were able to compute the optimum solution for each snapshot (which in high-load situations is unlikely to happen, due to the restricted running time), we have no guarantee that this would turn out to be the best choice in the end. Maybe it is better to incorporate incoming requests into the current plan and compute a new schedule only from time to time. We define the *Online Vehicle Dispatching Problem* (OLVDP) as the task of determining a good strategy for controlling the system in the long run. In particular, such a strategy has to determine how to integrate a new request into an existing dispatch, and when to globally recompute a new dispatch.

To conclude this initial description of the ADAC-Problem, remark that the input information we are dealing with can be classified into three groups according to its predictability. First, there are *deterministic* parameters which are known in advance with precision, like the shift-duration of the units, the unitary travel costs, etc. Then, there are some *stochastic* parameters that are not fixed beforehand, but may be estimated from information collected in the past, such as the travel time for getting to a certain position, the expected duration of service for a request, etc. Finally, information regarding the distribution of the requests in space and time is mainly of *uncertain* nature. Because of the high variance involved, this distribution cannot be predicted with reasonable exactitude from observations in the past.