Summary

The application of Approximate Dynamic Programming techniques

Optimization problems arise in many different areas, and while many different solution techniques exist, a particular problem requires a specific approach. Many problems that we encounter in real-life are of stochastic nature, i.e., they are driven by random events. Although one can speak about the average time to handle a call, the individual service times fluctuate. Therefore, we focus on stochastic optimization problems in which decisions are made to control the system. Those decisions are made at specific points in time, and affect the way the system behaves. Markov Decision Processes (MDPs) provide a framework for modeling such decision-making where the behavior of the system is covered by random events and under the control of a decision maker. To optimize a certain performance measure, the decisions should be made based on the current state of the system, but also taking into account the future behavior of the system. This idea is the key for the solution technique known as Dynamic Programming (DP).

Besides DP as a solution technique that solves the problem using iterative methods and requires that the system dynamics are known, other techniques exist that learn by interaction with the system, and searches for the optimal decisions in that way. To solve certain decision problems, and to derive optimal policies or the action to choose if the system is in a particular state, the relative value function plays an important role. The value function is a function that assigns a real-valued number to each state of the system, and can be interpreted as the additional costs or rewards by letting the system start in another state than the reference state. If the value function is known, it is used in order to derive optimal decisions for each state of the system. However, to solve the value function analytically is often very hard, or even impossible. Although there exist numerical algorithms to determine the value function, these algorithms require a lot of computational time and store the values for each state in a look-up table struc-
Summary

ture. It is not difficult to imagine that this causes problems if the number of possible states is very large.

Since the computational time and the memory requirements are very crucial aspects for the numerical algorithms, we search for a closed-form expression of the value function, which can then be used to determine the optimal decisions. However, to obtain a closed-form expression is analytically very hard. On the other hand, we could use a parameterized function, and determine the parameters in such a way that this parameterized function is a very good approximation to the relative value function. Other methods to handle the problem of large state spaces are methods that try to reduce the number of states by state aggregation, and methods that try to divide the system into multiple smaller systems for which the value functions are known or are easy to obtain analytically or numerically.

Some questions arise when looking at the methods to cope with large state spaces. For example on what structure to use for the parameterized function and on how to find the best parameters, since the goal is to find the value function which is not known beforehand. Other questions are related to the set of states that should be used in order to approximate the value function. In order to determine the parameters one does not need to consider all states. A parameterized function with a good fit on this set of states, would generalize for other states as well.

In this thesis we look at a number of those questions. After an introduction in Chapter 1 and a mathematical description of the methods in Chapter 2 we look in Chapter 3 at the structure of the parameterized function and discuss problems when one focuses on decision problems with average costs and countably infinite state spaces. Next, in Chapter 4, we look at state space aggregation in order to reduce the size of the state space. We also show that increasing the state space can be beneficial as well. The method of state space aggregation is applied to a realistic problem in Chapter 5. In the following chapter, in Chapter 6, we consider a system with time-varying parameters, of which a call center is an obvious example. In Chapter 7 we come back to our very first example in Chapter 1 of a multi-skill call center and we take decisions based on several approximations that are close to optimal.