Summary

This dissertation examines Dynamic Programming algorithms for routing and scheduling. These algorithms are based on the famous Dynamic Programming algorithm for the Traveling Salesman Problem already described over 50 years ago by Held and Karp [62] (and also independently by Bellman [17]). This algorithm is largely viewed as theoretical. Based on this algorithm we created new algorithms for the Vehicle Routing Problem and Job Shop Scheduling Problem and variants of these problems.

For several problems such a Dynamic Programming algorithm provides the best known complexity for an algorithm that guarantees to find the optimal value to the problem. For the Traveling Salesman Problem this was already known for the Job Shop Scheduling Problem we proved this in Gromicho, van Hoorn, Saldanha-da-Gama, and Timmer [60]. Most Dynamic Programming algorithms over sets have limited practical use as their running time and memory requirements are exponential. However, we show that by the use of bounding such Dynamic Programming algorithms can become practical applicable. We also show several ways to convert such Dynamic Programming algorithms into heuristic algorithms which then, although the optimality guarantee is lost, have practical value.

The basis of these Dynamic Programming algorithms is recursion over sets. To use Dynamic Programming over sets on a problem a solution for a such problem must be represented as a specific sequence of a set of nodes. The Dynamic Programming algorithm evaluates the best sequence based on the best sequence for each subset of the nodes. The power in the Dynamic Programming algorithm lies in the fact that it enables to consider all sequences by the evaluation of sequences based on each subset. Although there are still exponentially many subsets ($2^n$) this is exponentially less than the number of possible sequences ($n!$).

The Dynamic Programming algorithm can be converted into an iterative process to find all optimal solutions. We show in general how to create such a procedure and for the Job Shop Scheduling Problem we show the procedure in more detail. We also show the results and the total number of optimal solutions for small Job Shop Scheduling Problem benchmark instances.

For the Vehicle Routing Problem we show how to incorporate a large number of extensions to the Vehicle Routing Problem optimally into the Dynamic Programming algorithm. We also show the effects of these extensions on the complexity of the Dynamic Programming algorithm. This creates a general
framework to solve Vehicle Routing Problems as also described in Gromicho, van Hoorn, Kok, and Schutten [59]. We also show briefly how such framework can be used as pricing instrument in a column generation technique. For the Capacitated Vehicle Routing Problem we show with computational results what the effect of bounding can be on the Dynamic Programming state space.

We describe how to create a Dynamic Programming algorithm to solve the Job Shop Scheduling Problem, which provides the best time complexity to solve this problem to optimality. We show computational results for the Dynamic Programming algorithm for the Job Shop Scheduling Problem with and without the use of bounding. For a few Job Shop Scheduling Problem benchmark instances we are able to improve the best known lower bounds.

We create a new extension to the Job Shop Scheduling Problem by adding maintenance times to the machines. For this new problem we create a Mixed-Integer Programming formulation as well as a Dynamic Programming algorithm. We also create a bounding algorithm to be used within this Dynamic Programming algorithm. A comparison of computational results for both algorithms show that Dynamic Programming can outperform a state of the art Mixed-Integer Programming solver using this Mixed-Integer Programming formulation.

For well-known benchmark instances for the Job Shop Scheduling Problem we provide the best known values for the upper and lower bounds as well as the origin of these bounds. This information as well as detailed results of all computational experiments can be found in the appendix.