Summary

The thesis consists of three parts:

Part I We provide an explanation of the discrete event subset of the hybrid modelling language χ_t. Following that, we present an approach to model time in μCRL. Such an approach allows to model timed systems using an untimed modelling language, thereby enabling the reuse of existing model checking tools. Next, we extend this approach, in order to subsequently provide a general scheme to translate χ_t specifications into μCRL specifications. This translation scheme can be seen as a bridge between the areas of performance analysis and system verification, as χ_t is mainly targeted to the former, while μCRL is mostly used for the latter. Next, the applicability of the translation scheme is demonstrated by translating and verifying a number of χ_t specifications, most notably the specification of a turntable system. Finally, the insights gained by designing the translation scheme lead to an extension of μCRL with a notion of discrete relative timing, called $\mu\text{CRL}^{\textit{tick}}$. It is shown that $\mu\text{CRL}^{\textit{tick}}$ specifications can be translated to μCRL specifications, thereby (again) allowing the use of the μCRL toolset for the verification of timed systems. The $\mu\text{CRL}^{\textit{tick}}$ approach builds on earlier proposals to model time with an untimed process algebra by emphasising ease of use for the modeller, meaning that the modeller does not need to be concerned about the correctness of the time mechanism in a specification, and incorporating time jumps of arbitrary size.

Part II This part starts by providing a uniform presentation of the most prominent state space search algorithms in the field of Directed Model Checking. Many of these algorithms stem from the field of Artificial Intelligence, often incorporating additional information about the problem at hand, such that the search can be directed to interesting areas of the state space. In the presentation, connections between the searches considered are accentuated, highlighting a framework in which many searches can be placed. The overview is concluded by considering action-based guiding, and proposing a further generalisation of best-first search, in which the search may consist of a number of sequential phases, giving rise to the compositionality of state space searches. Finally, a glossary of Directed Model Checking terms is proposed.

After that, the focus is narrowed to the modelling and solving of scheduling problems by means of existing model checkers and their input languages. The techniques available in the model checkers SPIN (with PROMELA) and UPPAAL CORA (with priced

timed automata) for scheduling are discussed, and at times extended to improve efficiency or quality of the solution. Besides that, new techniques are proposed, which have been implemented in the μCRL toolset.

The beam search algorithm is the subject of the next chapter. Traditionally, beam search is applied on highly structured search trees. By extending the basic algorithm in a number of ways, it can be efficiently applied on arbitrary state spaces. Both state-based (detailed) beam searches and action-based (priority) beam searches are developed and discussed. This leads to a spectrum of beam searches. By applying a proposed mechanism to compare search algorithms, we establish that several prominent searches can be seen as specific instances of beam search in this spectrum. Finally, it is shown how these beam search variants, and another, exhaustive, search useful for scheduling, can be adapted to work in a distributed setting, in which a number of computers work together in a cluster to generate, or search, a state space.

The part is concluded by presenting some case studies in which most of the proposed search algorithms have been applied in practice. The most notable case study is a Clinical Chemical Analyser. In all cases the efficiency and effectiveness of the techniques in the μCRL toolset are analysed, while in one case, some of the techniques in the μCRL toolset are compared with techniques available in the model checker SPIN.

Part III Timed behaviour of systems can be compared by means of timed versions of bisimilarity relations. We look at the properties of timed branching bisimilarity, and point out that the existing definition by Van der Zwaag is not transitive in an absolute, continuous time setting. Therefore, the definition needs to be extended in order to ensure that it is an equivalence in an absolute, continuous time setting. A stronger notion is proposed (stronger in the sense that it relates fewer processes), and it is proven that this timed branching bisimilarity is indeed an equivalence, even if the time domain is continuous. Furthermore, we show that in case of a discrete time domain, the notion by Van der Zwaag and our stronger notion coincide. As Appendix C shows, the presented counter-example for transitivity also applies to the notion of timed branching bisimilarity by Baeten & Middelburg in case of a continuous time domain. So that notion does not constitute an equivalence relation as well.

After that, a rooted version of the extended timed branching bisimilarity is defined, and proven to be a congruence over a process algebra with parallelism, successful termination, and deadlock. In a number of ways, the proof differs from the usual congruence proof for untimed branching bisimilarity. For example, due to the presence of successful termination, there is a large number of cases. In fact, the congruence proof for the parallel composition operator is restricted to a setting without successful termination, since the number of cases in a proof considering successful termination is just too large. Furthermore, it is demonstrated that the standard approach for untimed branching bisimilarity, i.e. take the smallest congruence closure and prove that this yields a branching bisimulation, falls short in a timed setting.