Chapter 1

Introduction

1.1 Background

Sophisticated computer and information systems have become of the essence in today’s society, and they are being deployed at an ever increasing rate. Most contemporary computing systems are typically composed of entities that operate concurrently and cooperate through communication. Examples of such concurrent systems are computer and communication networks and protocols, operating systems, asynchronous circuits and many other embedded systems with application areas like process control, telephony and air traffic control to just name a few.

The correct design of concurrent systems is known to be a problem of considerable depth. One major source of difficulties lies in the fact that the functionality of these systems tends to be very large and complex. Traditionally, a sequential system (or program) is transformational and can be thought of as a function: given an input, it may produce an output. The specification of all input-output pairs defines the precise meaning of the system. A concurrent system is yet hard to describe in this way as it operates within an environment over an indefinite period of time. Its functional behavior is defined by the many ongoing interactions with its environment, and these interactions often exhibit complex interdependencie. For this reason, it is difficult to adequately specify, understand and predict the behavior of concurrent systems and, hence, to assess whether they meet their requirements.

Another source of difficulties lies in the distributed nature of concurrent systems. As the constituent entities of a concurrent system are dispersed over different locations, they must also interact with each other in order to realize the functionality of the system as a whole. An important example is found in communication protocols, where protocol entities interact according to strict rules. Indeed, the mere purpose of a communication protocol is to govern the orderly exchange of messages among communicating entities. Both the design of communication protocols and the
assessment of their correctness are certainly delicate tasks, and rigorous automated analysis methods are required to support these tasks.

The work described in this thesis pertains to the (design) verification of concurrent systems, and of communication protocols in particular. Verification refers thereby to the act of proving (or disproving) formally that a system design meets its expected properties, which can range from several types of general consistency requirements to more specific functional requirements asserted in, for instance, a logical language. What is strictly not meant is testing (unless it is exhaustive), or any other method which may indicate that a system design is “probably” correct. In order to prove that a system satisfies some property, all possible executions of the system must be checked to determine whether each and every one of them complies to the property. As such, verification is thus the means to guarantee the correctness of the design of a concurrent system or communication protocol.

1.2 Scope, objective, contributions

Throughout the past twenty years or so, various formal models have been proposed and studied to facilitate the specification and validation of (designs of) concurrent systems. These models differ in their expressiveness in terms of specification and in their tractability in terms of validation (i.e. verification and testing). However, most of the models have in common that their individual semantics renders a translation of the pure syntactic description of a concurrent system into some kind of a transition system, consisting of a set of states, a designated initial state, and a (labeled) transition relation among these states. This transition system represents the behavior of the concurrent system as a whole, i.e. the joint behavior of all the concurrent entities in the system.

A model particularly suited for specifying communication protocols is the communicating finite state machine (CFSM) model [Boc78, ZW+80, BZ81, BZ83]. In the CFSM model, a protocol is specified as a collection of processes (i.e. the protocol entities) that exchange messages over error-free simplex channels. Each process is modeled as a finite state machine (FSM) and each simplex channel is a FIFO queue. A (global) state of the protocol consists of a state for each FSM and a content for each simplex channel. A state transition can occur only when some process is ready to either send a message to one of its output channels, or receive a message from one of its input channels. The CFSM model is well-defined, elegant and rather easy to understand. These features make it attractive for both academia and industry. Indeed, the CFSM model has become a widely established means for specifying, verifying and testing communication protocols. Furthermore, it underlies two standardized specification languages, namely Estelle [BD89] and SDL [BH88]. For these reasons, and because there is no “best” formal model, we have chosen to study primarily the
verification of protocols specified in the CFSM model, although we will extend our scope later in the thesis to the verification of any (concurrent) system that can be viewed as a (labeled) transition system. In particular, state transitions may then represent system events other than just message transmissions and receptions, and system entities may communicate not only by asynchronous message passing but also by synchronous “handshaking”.

One of the most prevalent techniques for the verification of protocols, and concurrent systems in general, is state space exploration. State (space) exploration, which is widely known also as reachability analysis, amounts to exploring in a systematic manner the complete state space of a system, i.e. all states and transitions of the system that can be reached from a given initial state. Many different types of system properties can be verified by reachability analysis. It was originally proposed for verifying so-called logical correctness properties of protocols specified in the CFSM model, namely freedom of deadlocks, non-executable transitions (cf. dead code in a computer program), unspecified receptions, and buffer overflows or unbounded channel growth [Wes78, WZ78, ZW+80, BZ83]. These are general correctness properties that concern concurrent systems at large, albeit that unspecified receptions and buffer overflows or unbounded channel growth are particular to models featuring some form of asynchronous message passing. Reachability analysis can further be employed for the verification of individual, functional correctness properties of concurrent systems and protocols, like temporal safety and liveness properties [Lam80, Lam83, AS87, MP92]. This has emanated in the past decade from the development of model-checking methods for various temporal logics [LP85, CES86, VW86].

Reachability analysis is a simple and easy-to-automate verification technique. Moreover, it is fully automatic and thus no user-intervention is required. The effectiveness of reachability analysis has been witnessed by various notable success stories about its application to complex, industrial-size systems (see e.g. [Rud92]): it revealed several subtle design errors that had previously been missed by ad hoc approaches. However, a main limiting factor of reachability analysis is the swift explosion of the state spaces to be analyzed. Simple combinatorics affirm that the size of the state space of a system can be exponential in the size of the description of the system. This phenomenon is well-known as the state explosion problem. It severely hampers the practical usefulness of reachability analysis to industrial-strength applications. Indeed, the state spaces of most realistic systems are excessive in size (hundreds of thousands, or even millions of states) and thereby surpass any conceivable amount of memory available for analysis. Certain systems have in fact infinite state spaces, which causes most of their properties to be undecidable altogether (see e.g. [BZ83]).

Fortunately, the state explosion problem is not entirely inherent. It has long been recognized that many concurrent systems manifest a large number of reachable states and transitions that are redundant for verification purposes. One of the leading causes of this redundancy is the modeling
of concurrency by interleaving or, more accurately, the exploration of all possible interleavings of concurrent events. For instance, the execution of $k$ concurrent events is examined by exploring all $k!$ possible orderings of these events. Many interesting properties of concurrent systems are yet insensitive to the interleaving order of concurrent events. Consequently, for nearly two decades, researchers have put much effort into the development of improved state exploration techniques in order to relieve the state explosion problem [LCL87, Yua88]. Such state exploration based relief strategies reduce the complexity of reachability analysis by examining just part of the state space of a system, a part that is provably sufficient to verify certain properties. Practically, they enable the verification of properties of systems while avoiding most of the cost of modeling concurrency by interleaving.

Despite the various formal models around, virtually all state exploration based relief strategies proposed in the literature adhere in fact to the CFSM model, or a slight variation thereof [RW82, YG82, II83, GY84, GH85, GCL85, KI+85, GC86, ZB86, CR93, LM94, ÖU94, ÖU95, LM96]. The earlier strategies are rather limited in their applicability, as they were devised for protocols with just two communicating processes [RW82, YG82, GY84, GH85, GCL85, GC86, ZB86, CR93], or with otherwise strong conditions on the structural attributes of the individual processes [YG82, II83, KI+85]. It was only recently that several researchers innovated ideas to handle protocols with more complex communication structures. Liu & Miller [LM94, LM96] contributed by generalizing the technique of fair reachability analysis (FRA) proposed for two-process protocols in [RW82, GH85] to $n$-process cyclic protocols, where $n \geq 2$ processes form a unidirectional ring. FRA proves to be a very powerful relief strategy for the detection of deadlocks in cyclic protocols [LM94a], and it also provides a good basis for the efficient detection of non-executable transitions, unspecified receptions and unbounded channel growth [LM94b]. Özdemir & Ural [ÖU94, ÖU95] went further by proposing a relief strategy, called simultaneous reachability analysis (SRA), that is applicable to $n$-process protocols without topological or structural constraints. SRA can significantly reduce the number of stored states and explored transitions for the detection of deadlocks, non-executable transitions, unspecified receptions and buffer overflows in protocols with arbitrary communication structures. Yet another relief strategy developed in recent years is partial-order state exploration, which actually captures a collection of cognate algorithms better known as partial-order reduction methods [God90, Val90, HGP92, KP92a, Val92, Val93, GW93, GW94, HP95, God96, Pel96]. Unlike FRA and SRA, which are intended explicitly for protocols specified in the CFSM model, these methods are largely independent of the model used for specifying concurrent systems. They apply in principle to all specification models whose semantics induce (labeled) transition systems [HP95, God96]. Partial-order reduction methods are also effective as a relief strategy for verifying deadlock-freedom and freedom of non-executable transitions. Moreover, they lend themselves as an
efficient means for LTL model-checking, i.e. for model-checking system properties asserted in linear-time temporal logic (LTL) [Pnu77, Lam80].

The use of FRA, SRA or partial-order reduction methods does in many cases yield substantial savings in the memory and time requirements for state exploration [LM96, ÖU95, HP95, God96, Pel96]. Hence, with the advent of these techniques, the applicability of state exploration based verification has certainly been widened to “larger” concurrent systems and protocols. Nevertheless, since concurrent systems are inherently complex, and since this complexity is there to stay, pursuing additional performance improvements in verification clearly remains of utmost importance. It is this awareness that has motivated us to investigate the possibility of further increasing the effectiveness and efficiency of existing state exploration based relief strategies. Respecting the current state of the art, we have focused in particular on improving FRA, SRA and partial-order methods. While FRA appears to be a very powerful relief strategy indeed for the verification of cyclic protocols, the unidirectional ring topology of these protocols is still very restricted. The natural question is whether the effectiveness of FRA can be extended to protocols in the CFSM model with more complex, or even arbitrary communication topologies. Both SRA and partial-order reduction methods already enjoy such generality, but possible improvements of these two relief strategies may still be found in their performance. The critical issue thereby is the characteristic trade-off in computing between space and time. Although space is the major concern in verification, due to the state explosion problem, extreme care must be taken that potential extra savings in space are not attended by unacceptable expenses in time.

As a thesis, this manuscript presents in detail the results obtained with respect to the above research objective. Our main contributions can be summarized as follows:

• We generalize FRA from cyclic protocols to so-called multi-cyclic protocols in the CFSM model. A multi-cyclic protocol consists of a collection of unidirectional rings, or component cyclic protocols, which are interconnected such that no two rings share more than one process. The communication topology underlying multi-cyclic protocols has a rather wide applicability in practical protocol modeling. It captures not only protocols with a multi-ring topology (in particular all cyclic protocols), but also protocols with other common network topologies like a daisy-chain, a star, and a tree, as well as many combinations of these elementary topologies. We establish that FRA is an effective and efficient relief strategy for the detection of deadlocks of multi-cyclic protocols with a finite fair reachable state space (i.e. the reduced state space of the protocol explored by FRA), which follows the same result obtained for cyclic protocols. Furthermore, we also advocate that FRA is infeasible as a relief strategy beyond the class of multi-cyclic protocols. Albeit a negative result, recognizing the fundamental limitations of a technique is certainly of benefit as well.
• We present a relief strategy called leaping reachability analysis (LRA), which we propose as an incremental improvement of SRA for verifying logical correctness properties of protocols defined in the CFSM model. We prove that, for any protocol in this model, LRA maintains the power of SRA to detect all deadlocks, all non-executable transitions, all unspecified receptions and all buffer overflows of the protocol. Through an analytical comparison of the two relief strategies we show that LRA is an absolutely no-risk improvement of SRA, i.e. using LRA instead of SRA is at no cost whatsoever, neither in space nor in time. We complement the analytical results with an empirical evaluation of the performance of LRA and SRA, which in fact reveals that LRA can in many cases yield important extra savings over SRA in both space and (especially) time.

• We propose an enhancement of partial-order reduction methods for LTL model-checking. More precisely, we present an approach which integrates the concepts underlying LRA with those underlying partial-order reduction methods to enable further savings in both space and time for the verification of linear-time temporal properties of general, finite-state concurrent systems (i.e. any system whose behavior can be defined as a finite transition system). This approach is further fine-tuned for the CFSM model, by harmonizing its formulation with the formulation of LRA. LRA thereby emerges as an effective, uniform relief strategy for the verification of both logical and functional correctness properties of protocols defined in the CFSM model. Empirical results are provided which attest that our approach to LTL model-checking is indeed a notable enhancement of the partial-order reduction approach.

1.3 Organization of the thesis

The remainder of this thesis is organized as follows. Chapter 2 introduces the CFSM model and explains the technique of reachability analysis on the basis of this model. The principle limitations of reachability analysis are thereby addressed, viz. undecidability and state explosion. Chapter 3 provides a extensive survey of techniques proposed in the literature to relieve the state explosion problem, with an emphasis on state exploration based relief strategies. Chapter 4 generalizes fair reachability analysis from cyclic to multi-cyclic protocols in the CFSM model, and advocates the inherent infeasibility of this technique beyond the class of multi-cyclic protocols. Chapter 5 details the technique of leaping reachability analysis as an improvement of simultaneous reachability analysis for verifying logical correctness properties of protocols specified in the CFSM model. Chapter 6 reports on the results of a corresponding empirical comparison between these two relief strategies. Chapter 7 unfolds the proposed enhancement of partial-order reduction methods for LTL
model-checking. Finally, Chapter 8 provides the concluding remarks. The contributions of this thesis are reviewed and several directions for further research are given.