Abstract

A distributed system is composed of multiple independent machines that communicate using messages. Faults in a large distributed system are common events. Without fault tolerance mechanisms, an application running on a system has to be restarted from scratch if a fault happens in the middle of its execution, resulting in loss of useful computation. Checkpoint and recovery mechanisms are used in distributed systems to provide fault tolerance for such applications. A distributed application is composed of several processes, running on different nodes of a distributed system and communicating via messages. A checkpoint of a process is the information about the state of a process at some instant of time. A checkpoint of a distributed application is a set of checkpoints, one from each of its processes, satisfying certain constraints. If a fault occurs, the application is started from an earlier checkpoint instead of being restarted from scratch to save some of the computation. Several checkpoint and recovery protocols have been proposed in the literature.

The performance of a checkpoint and recovery protocol depends upon the amount of computation it can save against the amount of overhead it incurs. The performance of a protocol is dependent on several system and application characteristics, as well as protocol specific parameters. To improve the throughput of a distributed system, it is important to choose a checkpoint and recovery protocol with the best performance. Therefore performance evaluation and comparative study of the protocols is an important issue. Performance evaluations of checkpoint and recovery protocols on real distributed systems with distributed applications are realistic. But results of such studies are only valid for the systems in which the experiments are carried out and the set of test applications used, and cannot always be directly extended to other system and application characteristics. A simulation tool which can simulate any system and application characteristic is therefore important. Such a simulation tool enables one to

evaluate the performance of a checkpoint and recovery protocol under different system and application characteristics. In this thesis we present such a tool, named dPSIM. dPSIM enables us to simulate a wide variety of system and application characteristics. It also allows us to easily implement checkpoint and recovery protocols and measure different overheads related to the protocols. With the help of dPSIM we have studied a set of five checkpoint and recovery protocols belonging to different classes under different system and application characteristics, and analyzed their performance.

Since application characteristics affect the performance of checkpoint and recovery protocols, a single protocol cannot always provide the best performance to a system which runs different applications. Therefore, a system should automatically determine the best checkpoint and recovery protocol suitable for the application running in the system, and dynamically employ it. We have proposed a scheme for automatic identification of the best checkpoint and recovery protocol for the current system and application characteristics. One important part of the scheme is the solution of a communication pattern comparison problem. We have proposed a technique for comparing a pair of communication patterns which is based on a *graph similarity* problem. Detailed experiments were carried out which shows that the scheme can automatically identify the best checkpoint and recovery protocol for a variety of application characteristics.

In distributed systems consisting of more than one clusters, nodes within a cluster are connected via high speed links, while the clusters themselves can be connected via lower speed links. The low speed links create bottlenecks for the checkpoint and recovery protocols and increase their overheads. We propose two checkpoint and recovery protocols with low overheads for such systems. In a cluster-based distributed system, different clusters may reside under different administrative domains. As a result different checkpoint and recovery protocols may be running in different parts of the system. Therefore, it is necessary to provide a system-wide consistent fault tolerance by combining different checkpoint and recovery protocols. We present a checkpoint and recovery protocol which allows an individual cluster to run any one of two different protocols, namely a coordinated checkpointing protocol and a receiver based pessimistic message logging protocol, and still provides system-wide consistent recovery.