Using Consistent Subcuts for Detecting Stable Properties*

Keith Marzullo Laura Sabel

Abstract

We present a general protocol for detecting whether a property holds in a distributed system, where the property is a member of a subclass of stable properties we call the \textit{locally stable properties}. Our protocol is based on a decentralized method of constructing a maximal subset of the local states that are mutually consistent, which in turn is based on a weakened version of vector time stamps. The structure of our protocol lends itself to refinement, and we demonstrate its utility by deriving some specialized property-detection protocols, including two previously-known protocols that are known to be efficient.

1 Introduction

Chandy and Lamport [4] give a simple protocol that can be used to determine whether or not the global state of an asynchronous distributed system satisfies any given stable property. For most stable properties of interest (e.g., deadlock, termination, and lack of a token) there exist specialized protocols that are more efficient than a straightforward application of [4].

It would be useful if one could derive such special-purpose protocols by refinement of a general protocol. Unfortunately, the protocol of [4] was not developed with this in mind, and we have not found it conducive to such refinement. In order to facilitate refinement, we present a different protocol for detecting stable properties.

A naive general detection protocol is as follows: each process records in its local memory its entire local history. A separate process $p_0$ periodically retrieves these

\*This work was supported by the Defense Advanced Research Projects Agency (DoD) under NASA Ames grant number NAG 2-593, and by grants from IBM and Siemens. The views, opinions, and findings contained in this report are those of the authors and should not be construed as an official Department of Defense position, policy, or decision. Authors' address: Cornell University Department of Computer Science, Upson Hall, Ithaca, New York 14853-7501, USA.
local histories and extracts from them the latest global state. Process $p_0$ then tests to see if the property holds in this global state. Unfortunately, this naive protocol is impractical since it requires unbounded local memory. This can be fixed (at a cost of generality) by having each process record only its current state and having $p_0$ consider some subset of these local states that could be part of a sensible global state of the system. Not all stable properties can be detected this way, but it turns out that most stable properties that have been discussed in the literature can.

In this paper, we present a protocol that can be used to detect such properties. The protocol can be easily decentralized and can be customized for different properties in order to yield efficient special-purpose protocols. We demonstrate its utility by using it to derive such protocols including two previously-known protocols that are known to be efficient.

2 Definitions

We consider an asynchronous distributed system consisting of a set of $n$ nonfaulty processes $P = \{p_1, p_2, \ldots, p_n\}$. Between any two processes $p_i$ and $p_j$ there exist two unidirectional fault-free FIFO channels: $C_{i,j}$ from $p_i$ to $p_j$ and $C_{j,i}$ from $p_j$ to $p_i$. These channels have unbounded delivery time, and processes communicate only by sending and receiving messages over these channels.

Processes execute events, which are partitioned into send events, receive events, and local events. Unless stated otherwise, $e_i$ is an event of process $p_i$. The execution of an event $e_i$ produces the local state $\sigma_i$ of $p_i$. Thus, each local state $\sigma_i$ has a corresponding event $e_i$ that resulted in that state.

An arbitrary collection of local states may not constitute a sensible global state: the local state of a process in the collection may reflect the receipt of a message while no process' local state reflects the sending of that message. Such sets of local states are called inconsistent; a sensible collection of local states is called consistent.

A global state is defined to be a consistent set $\Sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$ of the processes' local states. We assume that channel states are captured in the local states of the processes. There are many ways to do this, e.g., by having each process maintain a history of all messages that it sends and receives. In practice, one must ensure that the representation of the channel states uses bounded local memory.

A consistent cut is defined to be a set of events $C = \{e_1, e_2, \ldots, e_n\}$ such that the set of states $\{\sigma_1, \sigma_2, \ldots, \sigma_n\}$ produced by $C$ is a global state. Thus, each consistent cut has a corresponding global state, and vice versa.

A property is a predicate over the global state of the system. A stable property