Hierarchy-Based Incremental Analysis of Communication Protocols*

Kuo-Chung Tai

Department of Computer Science
North Carolina State University
Raleigh, NC 27695-8206
e-mail: kct@csc.ncsu.edu

Pramod V. Koppol

Department of Computer Science
North Carolina State University
Raleigh, NC 27695-8206
e-mail: gr-kvp@druid.csc.ncsu.edu

Abstract

In this paper we present an incremental strategy for reachability analysis of communication protocols modeled as sets of communicating finite state machines (CFSMs) with synchronous communication and direct naming. A set of CFSMs is organized into a hierarchy. We present an algorithm that, for a given hierarchy of a set M of CFSMs, incrementally composes and reduces subsets of CFSMs in M and finally produces a minimum CFSM describing the external behavior of M. We also show that this incremental reachability analysis guarantees the detection of global deadlocks. We provide an algorithm for selecting a hierarchy for a set of CFSMs and report some empirical results.

1 Introduction

We consider communication protocols modeled as sets of communicating finite state machines with synchronous communication and direct naming. The reachability analysis of a communication protocol P is to identify the behavior of P and verify safety properties, such as freedom from deadlocks or livelocks, without having P's specification. The external behavior of P can also be used to determine the satisfaction of P's specifications [CES86].

For a set M of CFSMs, the conventional approach to reachability analysis, referred to as the *all-at-once* approach, is to compose all CFSMs in M at the same time and construct the reachability graph (RG) of M, which contains all reachable global states of M. The number of states in the RG of M may be as high as the product of the numbers of states of individual CFSMs in M. This state explosion problem makes the all-at-once approach impractical for analyzing large communication protocols.

Let $M = \{M_1, M_2, ..., M_n\}$, be a set of CFSMs, where M_i, $0 < i <= n$, denotes a CFSM. The CFSMs in M can be organized into a hierarchy, which defines a hierarchical structure of M_1, M_2, ..., and M_n. For example, $((M_1, M_3), M_4, (M_2, M_5))$ is a hierarchy of M_1 through M_5. According to this hierarchy, M_1 and M_3 are combined and then minimized to produce the composite CFSM M_{13}. Similarly, M_2 and M_5 are combined and then minimized to produce the composite CFSM M_{25}. Finally, M_{13}, M_4 and M_{25} are combined and minimized to produce a CFSM describing the external behavior of M_1 through M_5.

The organization of this paper is as follows. Section 2 gives basic definitions. Section 3 presents an algorithm that, for a given hierarchy of a set M of CFSMs, incrementally composes and reduces subsets of CFSMs in M and finally produces a minimum CFSM describing the external behavior of M. It also shows that our algorithm guarantees the detection of global deadlocks. Section 4 addresses the problem of how to select a hierarchy for a set of CFSMs. Section 5 provides empirical results. Section 6 concludes this paper with a comparison of different strategies for incremental analysis. Due to the limit on space, proofs of all theorems in this paper are omitted and can be found in [TK93].

2 Preliminaries

In this paper, we consider communication protocols modeled as sets of finite state machines in which send

*This work was supported in part by NSF under grant CCR-8907807 and by the Center for Communication and Signal Processing at NCSU

0-8186-3670-X/93 $03.00 © 1993 IEEE

and receive are blocking and the destination (source) of a send (receive) command is a process. Thus a pair of sender and receiver defines a channel. A send in process P_1 and a receive in process P_2 match if the destination (source) of the send (receive) is P_2 (P_1). A matching operation between a send and receive is referred to as a synchronization operation.

Let (channel name, message) be denoted as an event. For an event u, a send operation for u is denoted as u, a receive operation for u as $\bar{u}$, and a synchronization operation for u as $*u$. A CFSM is a 5-tuple $(\Sigma, V, \sigma, s, t)$, where Σ consists of send, receive, and synchronization operations, V is a finite set of states, σ is a nondeterministic state transition function that maps a state in $(V - t)$ and an element in Σ into a subset of V, s is the initial state, and t is a set of final states, each indicating a termination of the CFSM. In this paper, a CFSM $(\Sigma, V, \sigma, s, t)$ is represented as a directed graph (V, E), where E is the set of transitions, each labeled by a send, receive, or synchronization operation. A transition labeled by a send (receive, synchronization) operation is referred to as a send (receive, synchronization) transition. A transition of a state refers to a transition leaving the state.

Fig. 1 shows three CFSMs using direct-naming. A transition labeled by $-(i, j, m)$ indicates a send operation with M_i as the sender, M_j the receiver, and m the message. Similarly, a transition labeled by $+(i, j, m)$ indicates a receive operation with M_i as the sender, M_j the receiver, and m the message. Also, a transition labeled by $*(i, j, m)$ indicates a synchronization for the delivery of message m from M_i to M_j.

In this paper, we use the notation of CCS (Calculus of Communication Systems)[Mil89] to formally define incremental reachability analysis. A synchronization operation in a CFSM is referred to as an internal operation and is denoted as τ when the event associated with this operation is ignored (Thus, a synchronization operation for event u in a CFSM is denoted either as $*u$ or as τ). Let M be a set of CFSMs. A CFSM M' is referred to as the *minimum CFSM* for M if M' is the minimal machine that is observational equivalent [Mil89] to M. In this paper, $\sim$ denotes observational equivalence between two CFSMs. A brief introduction to the theory of CCS including observational equivalence can be found in [CPS93].

Let M be a set of CFSMs. A channel in M is referred to as an internal channel in M if it involves CFSMs in M, but not CFSMs outside M. Since direct-naming is used, a channel in M involves two specific CFSMs and thus the set of internal channels of M can be determined directly from M. As an example, for two CFSMs M_1 and M_2 communicating with each other by using direct-naming, {(1,2),(2,1)} is the set of internal channels. $RG(M)$ denotes the reachability graph of M with each channel involving some CFSMs in M as an internal channel. More precisely, $RG(M)$ is the minimum CFSM that defines the set of sequences of (1) synchronization operations among CFSMs in M and (2) send and receive operations of CFSMs in M that involve external channels. The minimum CFSM that is observational equivalent to $RG(M)$ is denoted by $MRG(M)$.

A hierarchy of CFSMs denotes a set of CFSMs with a hierarchical structure. Formally, a hierarchy H of CFSMs M_1, M_2, ... and M_n, $n > 1$, is denoted as $(H_1, H_2, ..., H_m)$, where each Hj, $0 < j <= m$, is either a hierarchy or M_i for some $0 < i <= n$, and each M_k, $0 < k <= n$, occurs only once in H. Each H_j, $0 < j <= m$, is said to be a component of H. H is said to be a sub-hierarchy of itself. For each H_j, $0 < j <= m$, (1) H_j is said to be a proper sub-hierarchy of H, (2) a proper sub-hierarchy of H_j, if it exists, is also said to be a proper sub-hierarchy of H, and (3) H is said to be the parent-hierarchy of H_j. For example, $(((M_1, M_3), M_4), M_2)$ has (M_1, M_3) and $((M_1, M_3), M_4)$ as its proper sub-hierarchies.

3 Hierarchy-based incremental reachability analysis

For a set M of CFSMs, the all-at-once approach is to construct $RG(M)$, and then reduce it to a minimum CFSM. This approach is very time- and space-consuming. An alternative is to select a hierarchy of M and incrementally compose and reduce the CFSMs in M according to this hierarchy. In this section, we present an algorithm for this incremental approach.

Let H be a hierarchy of a set M of CFSMs M_1, M_2, ..., and M_n, $n > 1$, with synchronous communication and direct naming. The level of a hierarchy H of CFSMs with respect to itself is one. For each proper sub-hierarchy of H, its level with respect to H is one plus the level of its parent-hierarchy with respect to H. The depth of a hierarchy H of CFSMs, or $D(H)$,

is defined as the maximum level of sub-hierarchies of H. A bottom-up traversal of a hierarchy H is to visit the level $D(H)$ sub-hierarchies of H, then the level $(D(H)-1)$ sub-hierarchies of H, and so on.

Our algorithm, referred to as HBIA_DN, is as follows: Perform a bottom-up traversal of H to visit sub-hierarchies of H. For a sub-hierarchy B, where $B = (B_1, B_2, \ldots, B_m)$, $m > 1$, let $RG(B) = RG(MRG(B_1), MRG(B_2), \ldots, MRG(B_m))$,
Let $MRG(B)$ be the minimum CFSM that is observational equivalent to $RG(B)$.

Theorem 1. Let H be a hierarchy of a set M of CFSMs $M_1, M_2, \ldots, and\ M_n$, $n > 1$, with synchronous communication and direct naming. After algorithm HBIA_DN has been applied to H, (1) $RG(H)$ defines the set of sequences of (i) synchronization operations involving components of H and (ii) send and receive operations that involve CFSMs not in M, (2) $RG(H) \sim RG(M)$, and, (3) $MRG(H) = MRG(M)$. □

Figures 2 through 4 illustrate our incremental approach by considering the hierarchy $((M_1, M_2), M_3)$, where M_1, M_2, and M_3 are the CFSMs shown in Fig. 1. Since M_1, M_2 and M_3 use direct-naming, the determination of internal channels for incremental analysis is easy. We first compose M_1 and M_2 with (1,2) and (2,1) as internal channels. The resulting CFSM is $RG(M_1, M_2)$, which is shown in Fig. 2. Then we derive the minimum CFSM that has the same external behavior as $RG(M_1, M_2)$. This minimum CFSM, shown in Fig. 3, is $MRG(M_1, M_2)$. The next step is to compose the CFSM in Fig. 3 and M_3 with {(1,3),(2,3),(3,1),(3,2)} as the set of internal channels. The resulting CFSM of this composition is given in Fig. 4. Since the CFSM in Fig. 4 has synchronization transitions only, its minimum observational equivalent CFSM has only one state with no transitions. In figures 5 through 11, we apply our incremental analysis algorithm to generate the CFSM representing the external behavior of the alternating bit protocol (ABP). ABP consists of three component processes *sender*(S), *receiver*(R) and *medium*(M). The sender accepts messages from a user entity $E1$ and transmits the message to the receiver, which in turn delivers the message through the medium to user entity $E2$.

Incremental reachability analysis using algorithm HBIA_DN guarantees the detection of global deadlocks and may also detect local deadlocks. A state in $RG(M)$ is called a global deadlock state if (i) S has no successor state, (ii) S is not a final state (i.e., at least one CFSM in M is not in a final state when M is in state S), and (iii) at least one path from the initial state of $RG(M)$ to S contains only synchronization transitions. Condition (iii) is required because if every path from the initial state to S contains some send or receive transitions involving external channels, then whether S will be entered depends upon the interaction between M and other CFSMs which constitute the *environment*. Assume that in Fig. 1, the label associated with the transition from state S_{33} to state S_{34} is changed from $+(2,3,e)$ to $+(2,3,g)$. Then the state (S_{14}, S_{23}, S_{33}) in Fig. 4 becomes a global deadlock state.

Since observational equivalence is used in algorithm HBIA_DN and in the proof for theorem 1, one important question is whether two observational equivalent CFSMs have the same result on deadlock. Consider $RG(M)$ and $MRG(M)$. If $RG(M)$ contains some send or receive transitions involving external channels, then $RG(M)$ contains a global deadlock state if and only if $MRG(M)$ contains a global deadlock state. (The deadlock state in $MRG(M)$ is entered from the initial state through the internal operation τ). If $RG(M)$ has only synchronization transitions, then $RG(M)$ has empty external behavior and $MRG(M)$ contains only one state and no transitions. The only state in $MRG(M)$ satisfies the definition of a global deadlock state, although $RG(M)$ does not necessarily contain a global deadlock state. Thus, $RG(M)$ and $MRG(M)$ have the same result on deadlock only if they have non-empty external behavior. To deal with this situation, whenever $MRG(M)$ has empty external behavior, we perform deadlock detection using $RG(M)$ instead of $MRG(M)$.

4 The hierarchy selection problem

One major issue in incremental reachability analysis is the selection of a hierarchy for a set of CFSMs. If an arbitrary hierarchy is used, incremental analysis may take more space than all-at-once analysis. For example, for the set of CFSMs in Fig 1, the space required for incremental reachability analysis using the hierarchy $((M_1, M_2), M_3)$ is more than that required for all-at-once reachability analysis. In this section, we provide an algorithm for selecting a hierarchy for a set of CFSMs with synchronous communication and direct naming.

A Hierarchy H of a set M of CFSMs may be viewed as a tree, with each leaf corresponding to a distinct CFSM in M, and each internal node corresponding to a sub-hierarchy of H. Our incremental analysis algorithm is as follows: assuming that the tree representing the hierarchy H has depth d, the CFSMs corresponding to internal nodes at level $d-1$ are generated first and then those at level $d-2$ are generated and so on. The CFSM corresponding to each internal node is generated by performing the composition of the CFSMs which are the children of that node and then minimizing the composite CFSM with respect to observational equivalence. The algorithm terminates when the CFSM corresponding to the root node is generated.

Let S_i represent the number of states in the CFSM (before minimization) at node i in the tree. The *cost* of a *node* is defined as follows: If i is a leaf node, $Cost(i) = S_i$, otherwise, $Cost(i) = Max\{S_i, Max\{Cost(j) \mid j \text{ is a child of } i\}\}$. The *cost* of a *hierarchy* H is defined as the cost of the root node of the tree representing the hierarchy H. For a given set of CFSMs, the hierarchy with the minimum cost is said to be *optimal*. Given a set of CFSMs, the *hierarchy selection problem* is to select an optimal hierarchy. The straightforward approach for selecting an optimal hierarchy is to compute the costs of all possible hierarchies and then choose the one with the minimal cost. However, the cost of an internal node S_i can only be computed by performing the composition of its children and hence, the straightforward approach, or any other approach that requires computing the cost for internal nodes, is impractical. In section 4.1, we define several metrics for measuring the complexity of synchronization for a set of CFSMs. In section 4.2 we apply these metrics to develop an algorithm for selecting a hierarchy for a set of CFSMs.

4.1 Metrics for measuring complexity of synchronization

Let M be a set of CFSMs with synchronous communication and direct naming. We define the following metrics for M.

Send/Receive Count (SRC): The number of send and receive transitions in M is considered as a metric for measuring the complexity of synchronization operations that may take place during the composition of the CFSMs in M. SRC for M is defined as, $SRC(M) = \#$ *of send/receive commands involving CFSMs in* M.

Normalized Send/Receive Count (NSRC): The $NSRC$ for M is the proportion of the total number of send and receive transitions in M that may be involved in synchronization operations. It is defined as,

$$NSRC(M) = \frac{SRC(M)}{Total\ \#\ of\ transitions\ in\ M}$$

Send/Receive Density (SRD): The SRD for M is the average contribution of each CFSM in M towards the send/receive count of M. SRD for M is defined as,

$$SRD(M) = \frac{SRC(M)}{\mid M \mid}$$

Normalized Send/Receive Density (NSRD): The $NSRD$ for M is the average contribution of each CFSM in M towards the normalized send/receive count of M. It is defined as,

$$NSRD(M) = \frac{NSRC(M)}{\mid M \mid}$$

$$= \frac{SRD(M)}{Total\ \#\ of\ transitions\ in\ M}$$

4.2 An algorithm for hierarchy selection

For a set M of CFSMs, our approach to selecting a hierarchy is as follows: we select a subset of M which has the maximum $NSRD$ and generate the composite CFSM for this subset and find its minimum CFSM. We then replace the subset with the minimum CFSM and repeat the above procedure for the new set of CFSMs. Thus, in our approach, hierarchy selection and incremental reachability analysis proceed simultaneously.

The rationale behind using the $NSRD$ for selecting the subset is the intuition that the subset with maximum normalized send/receive density would generate the maximum average number of synchronization transitions per process and hence provide scope for a larger gain with respect to reduction in state space size. This intuition is consistent with the core idea behind incremental analysis which is to hide as much internal detail of subsystems as possible in the early stages of analysis.

To find a subset of M with the maximum normalized send/receive density, we represent M as a matrix C of size $n \times n$, where n is the number of CFSMs in M. C is referred to as the *interaction matrix* of M and its contents are as follows: $C[i,i] = 0$, and

for $i \neq j$, $C[i,j] = (\#$ *of sends from* M_i *to* $M_j) +$ $(\#$ *of receives in* M_j *from* $M_i)$. Let *Trans* be a vector such that, $Trans[i] = \#$ *of transitions in* M_i

A ***sub-matrix*** A' of a matrix A is defined as a matrix which is formed by deleting zero or more rows, and corresponding columns, from A. (Note that a matrix is a sub-matrix of itself) Thus, a sub-matrix of C is the interaction matrix for the corresponding subset of CFSMs in M.

Let $MatSum(C)$ denote the sum of all elements in matrix C. For a $k \times k$ sub-matrix C' of an $n \times n$ matrix C,

$$SRD(C') = \frac{MatSum(C')}{k}$$

For the subset of CFSMs represented in a sub-matrix C', the normalized send/receive density (NSRD) is computed as follows

$$NSRD(C') = \frac{SRD(C')}{\Sigma\{Trans[i] \mid Mi \text{ is represented in } C'\}}$$

Given the interaction matrix C for a set of CFSMs, in order to select a subset of CFSMs with the maximum normalized send/receive density, we select a sub-matrix C' of C with the maximum normalized send/receive density.

Theorem 2. Given the interaction matrix C and vector *Trans* for a set of CFSMs, the problem of selecting a sub-matrix C' with the maximum normalized send/receive density is NP-complete. □

Although the problem is NP-complete, we feel that our approach to selecting a hierarchy is more practical and easier than the straightforward approach or any other approach that requires computing the costs of intermediate nodes in the tree representing the hierarchy. As an example, consider the CFSMs for sender, receiver and medium in the alternating bit protocol of Fig 5, 6 and 7. The interaction matrix is (row 1 corresponds to Sender; row 2 to Medium; row 3 to Receiver),

$$C = \begin{pmatrix} 0 & 4 & 0 \\ 4 & 0 & 4 \\ 0 & 4 & 0 \end{pmatrix}$$

and vector Trans is,

$$Trans = [6,\ 8,\ 6]$$

The hierarchy selected for this set is ((Sender, Medium), Receiver).

5 Empirical studies of incremental analysis and deadlock detection

We have compared algorithm HBIA_DN and the all-at-once approach by applying them to the following five concurrent programs written in CCS: *Alternating Bit Protocol (ABP)* (We considered two versions of *ABP*. The difference between the two versions is that the sender and receiver in *ABP-I* have a time out mechanism while those in *ABP-II* do not. Both use a reliable medium, but $ABP - I$ contains deadlocks while $ABP - II$ does not), *Carrier Sense Multiple Access (CSMA) with Collision Detection* [Par86], *Dekker's Mutual Exclusion Algorithm (DME)* [Wal88], and *Peterson's Mutual Exclusion Algorithm (PME)* [Wal88].

For each of these CCS programs, say M, we performed the following tasks:

- (a) selected a hierarchy using the strategy described in the previous section, and applied algorithm HBIA_DN.
  - (a.1) determined the numbers of states in the resulting MRG. This number is referred to as **#EXT**.
  - (a.2) determined the maximum of total number of states at any time during the execution of algorithm HBIA_DN. This number is referred to as **#HBIA_max**.
- (b) applied the all-at-once approach to determine the number of states in $RG(M)$. This number is referred to as **#all-at-once**.

These CCS program were analyzed using the Concurrency Workbench [CPS93] to determine the values of #EXT, #HBIA_max, and #all-at-once, as well as the existence of global deadlocks. The results of this analysis are shown in Table 1. The column titled **#_of_Procs** gives the number of component processes of the protocol. The number of states in each component process are provided under the title **#_of_States**.

For PME, the hierarchy chosen for incremental analysis is exactly the all-at-once approach and thus the value of #HBIA_max is the same as that of #all-at-once. The reason is that during an intermediate stage of incremental analysis, the number of states may be larger than the value of #all-at-once. This indicates that the selection of a hierarchy for a set of

Protocol	#_of_Proc	#_of_States	#EXT	#HBIA_max	#all-at-once	Deadlock?
ABP-I	3	6,6,5	9	40	41	Yes
ABP-II	3	6,6,5	2	10	12	No
CSMA	3	7,10,7	12	31	36	No
DME	5	5,9,2,2,2	2	114	126	No
PME	5	6,6,2,2,2	2	31	31	No

Table 1: Empirical Results

CFSMs is critical. The above six programs are too small to show a significant reduction of space due to incremental analysis. We are currently using larger programs for empirical studies.

6 Conclusion

In this paper, we have developed a hierarchy-based approach to incremental detection of deadlocks in a set of CFSMs with synchronous communication and direct naming. We have presented an algorithm for the selection of a hierarchy for a set of CFSMs. Also, for a given hierarchy, we have described how to perform incremental analysis. Furthermore, we have proved that our incremental algorithm guarantees the detection of global deadlocks and may also detect local deadlocks.

Incremental analysis of communication protocols has a number of advantages. First, it may take less time and space than all-at-once analysis. Second, incremental analysis can significantly reduce the effort for re-analysis of a large protocol due to correction or enhancement. Third, during an incremental development of a large communication protocol, incremental analysis can be incorporated into the development process. Assume that a module M of a protocol P is modified or replaced. If module M has the same MRG as its original version, then there is no need to re-analyze P for verification of properties such as freedom from global or local deadlock. If module M has a different MRG, then portions of P that are affected by module M can be incrementally re-analyzed in bottom-to-top order. The re-analysis of P stops at a sub-hierarchy of P if the MRG of this sub-hierarchy is not changed due to the changes in module M.

Incremental analysis of a set of CFSMs with synchronous communication has been studied by some researchers. In [LSU89] algorithms for incremental composition and reduction of CFSMs with direct-naming were given. At each composition step, a pair of CFSMs were combined. No quantitative way of ranking CFSMs for incremental composition was used. The paper presented three heuristic rules for reducing a CFSM to a smaller, observational equivalent one. However, the three rules do not necessarily produce a minimum, observational equivalent CFSM.

In [SKB90] an algorithm for incremental composition of CFSMs with direct-naming was given. Each transition of a CFSM may be associated with one input (receive command) and one or more outputs (send commands). At each composition step, a pair of CFSMs were combined into one by (1) matching pairs of transitions in these two CFSMs such that an output of one transition is the input of the other transition, and (2) keeping transitions that neither take input from nor produce output to the other CFSM. There was no discussion of whether the resulting CFSM is observational equivalent to the original set of CFSMs.

In [YY91] a prototype tool for incremental analysis of programs written in an Ada-like design language called PAL was described. For a PAL program, the tool transforms it into a set of process graphs, one for each task, and performs composition and simplification of these process graphs according to program structure. For each subsystem, however, the user needs to provide a simpler process that is equivalent to the process graph of the subsystem. The tool can also determine the equivalence of process graphs. The axioms of composition, simplification, and equivalence used in this tool are based on ACP [BeK84], which is a theory of process algebra different from CCS.

Acknowledgment

We would like to thank Dr. Rance Cleaveland for helpful discussion on CCS and Concurrency Workbench and for providing several CCS programs used in our empirical studies.

References

[And91] G. R. Andrews, *Concurrent Programming*, Benjamin/Cummings, 1991.

[BeK84] J. A. Bergstra, and J. W. Klop, "Process Algebra for Synchronous Communication", *Information and Control*, pp 109-137, 1984.

[CES86] E. M. Clarke, E. A. Emerson, and A. P. Sistla, "Automatic verification of finite-state systems using temporal logic", *ACM TOPLAS*, Vol. 8, No. 2, pp 244-263, April 1986.

[CPS93] R. Cleaveland, J. Parrow and B. Steffen, "The Concurrency Workbench: A Semantics Tool for the Verification of Concurrent Systems", *ACM Tran. Programming Languages and Systems*, Vol 15, No. 1, pp 36-72, Jan. 1993.

[LSU89] A. M. Lapone, K. K. Sabnani and M. U. Uyar, "An Algorithmic Procedure for Checking Safety Properties of Communication protocols", *IEEE Transactions on Communications*, pp 940-948, Sept. 1989.

[Mil89] R. Milner, *Communication and Concurrency*, Prentice-Hall, 1989.

[Par86] J. Parrow, "Verifying a CSMA/CD Protocol with CCS", *Edinburgh University technical report ECS-LFCS-86-18.*

[SKB90] B. Sarikaya, V. Koukoulidis and G. V. Bochmann, "Method of Analyzing Extended Finite-State Machine Specifications", *Computer Communications*, Vol 13, No.2, pp 83-92, March 1990.

[TK93] K.C. Tai and P.V. Koppol, "Incremental Reachability Analysis of Communication Protocols", *TR-93-13, Dept. of Computer Science, North Carolina State University.*

[Wal88] D. Walker, "Analysing mutual exclusion algorithms using CCS", *Edinburgh University technical report ECS-LFCS-88-45.*

[YY91] W. J. Yeh, and M. Young, "Compositional Reachability Analysis Using Process Algebra", *Proc. ACM fourth Workshop on Software Testing, Analysis, and Verification*, 49-59, 1991.

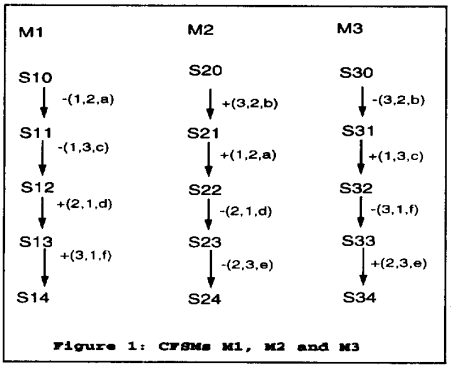

Figure 1: CFSMs M1, M2 and M3

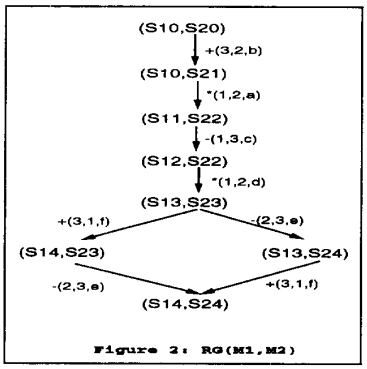

Figure 2: RG(M1,M2)

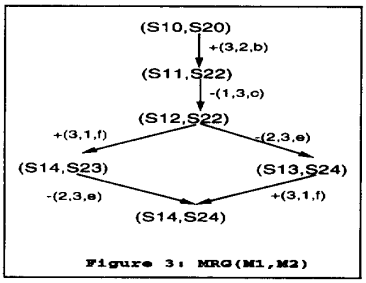

Figure 3: MRG(M1,M2)

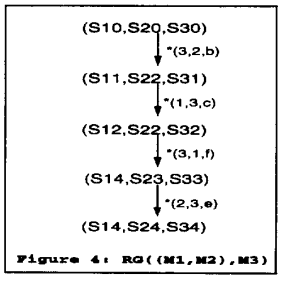

Figure 4: RG((M1,M2),M3)

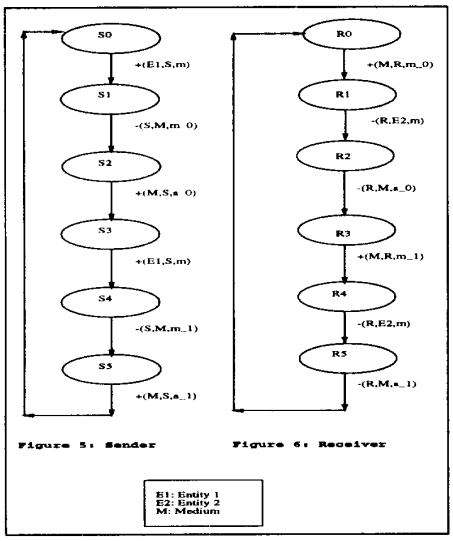

Figure 5: Sender

Figure 6: Receiver

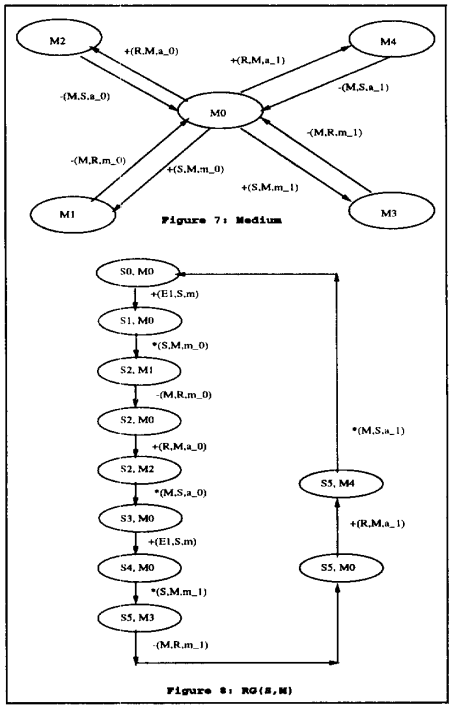

Figure 7: Medium

Figure 8: RG(S,M)

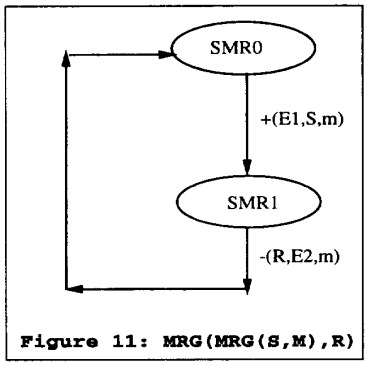

Figure 11: MRG(MRG(S,M),R)

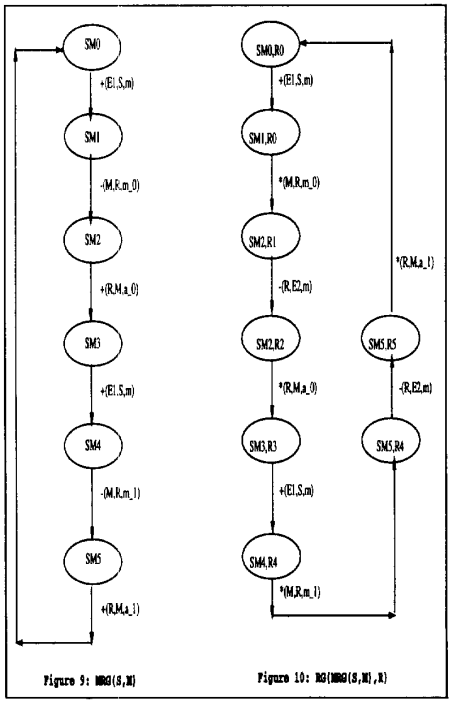

Figure 9: MRG(S,M)

Figure 10: RG(MRG(S,M),R)