RE-ROUTING ANALYSIS OF A FAULT-TOLERANT ATM SWITCH BASED ON A PARALLEL ARCHITECTURE

S. Segkhoonthod and M.C. Sinclair[1]

1 Introduction

One of the approaches for ATM switch design is to use a number of routing networks (multistage interconnection networks: MINs) in parallel so that multiple paths are created between inputs and outputs [1, 2]. This technique, called a replicated network, results in an improvement in both performance and reliability. However, it should be noticed that with this configuration, a great deal of complexity is needed in the Input Modules (IMs), which are used to distribute packets among routing networks, as they have to decide to which routing network packets should be sent.

In [3], we proposed a fault-tolerant ATM switch which adapts the idea of using several routing networks in parallel. The switch has a distribution network which is in charge of dispatching packets among the routing networks. As a result, the IMs are simply required to submit packets to the distribution network. Since the distribution network also carries out part of routing process, the routing networks themselves are truncated—resulting in a low overall complexity for the switch. In terms of fault tolerance, the distribution network is also in charge of re-routing packets in the case of faulty switching elements or links. Unlike some other fault-tolerant switches, where additional effort is required to avoid faults, in the proposed switch there is no modification in the routing algorithm when faults occur. In [3], we presented a probabilistic analysis of the proposed switch with re-routing deactivated, as well as simulation results for re-routing in the absence of faults. It was shown that the proposed switch gives both improved performance and reduced complexity when compared with replicated networks.

In this paper, we present a performance analysis of the switch with re-routing in the absence of faults, based on the method of state diagrams, and compare the results with simulation. As in [3], these show that the proposed switch gives improved performance when compared with replicated networks. The paper is organised as follows. In section 2, the switch architecture and operation are presented. Section 3 gives the evaluation of the switch performance. The results obtained from both analytical models and simulation are presented and discussed in section 4. Finally, conclusions and future work are given in section 5.

2 Switch Architecture and Operation

Fig. 1 depicts the general configuration of an l-level $N\times N$ version of the proposed switch which consists of an $N\times K$ distribution network (DN), K $N\times N$ routing networks (RNs) and N $K\times 1$ output modules (OMs) where $K = 2^l$. The output of each IM is connected to the input of the DN. The IMs are in charge of receiving packets from the incoming lines and submitting them to the DN. The DN is constructed based on an l-level binary tree where each node is a plane of $N/2$ 2×4 switching elements (SEs). In stage s, there will be 2^s such planes. The 2×4 SE used in the DN is not a typical 2×4 crossbar; it behaves like a 2×2 SE, but with two identical sets of two outputs, instead of only one set as in case of a 2×2 SE. At a given node (a plane that belongs to stage s), one set of outputs of each SE are connected to the inputs of the SEs in its left child node (a plane in stage $s+1$), whereas the other set are for the right one (another plane in stage $s+1$). In addition, link patterns from a parent node to both its left and right child nodes are the same. The link pattern between a node in stage s and another in stage $s+1$, is identical to the link pattern between stages s and $s+1$ in an $N\times N$

[1]The authors are with Dept. of Electronic Systems Engineering, University of Essex, Wivenhoe Park, Colchester, Essex C04 3SQ.

Banyan network. At stage $l-1$, the outputs of the 2×4 SEs of the DN are connected to the inputs of the 2×2 SEs of the RN. An $N \times N$ Banyan network is used as a RN. However, it is not a complete one, instead each routing network is truncated, with stages 0 to $l-1$ absent, because these stages are already embedded in the DN. Each output of the RNs is connected to one of the inputs of the OMs. Here an OM can accept K cells at the same time. A 16×16 version of the proposed switch with $l=2$ and $K=4$ is shown in Fig. 2. It can be seen that there are 4 Banyan networks embedded with their stages 0 and 1 overlapped in the DN.

The DN and RNs are both unbuffered. The switch operation is based on a request/acknowledgement protocol, consisting of two phases: path-establish and packet-transfer [4]. At the beginning of the first phase, an IM that has a packet generates a request, a form of short packet, and then submits it to the DN and waits for an acknowledgement (ACK). If the returned ACK is a positive one, the IM can transfer the packet to the appropriate OM during the second phase. If not, it waits for the next path-establish phase and starts the same procedure again. After being admitted to the DN, the request goes along stage by stage attempting to set up a path across the DN and RN until reaching its required OM where a positive ACK is now generated and sent backward following the path that the request has set up. However, in some cases when setting up the path, the request might be unable to get through because of losing contention to another request that also wants the same output of an SE at that stage. In this case, a negative ACK is generated.

The DN is in charge of distributing, routing and re-routing. Basically, it both distributes an incoming request to one of the RNs according to some rule, and at the same time routes the request. After receiving a request from the DN, the RN then directs it to an appropriate OM. The routing algorithm used is the same as that of a Banyan network. The 2×4 SEs employed in the DN have two sets of outputs, referred to here as first and second choices. In each set, there are two outputs—upper and lower outputs like those of a 2×2 SE. As an example, when a request arrives at the upper input of a 2×4 SE in stage s, if bit s of its routing tag is 0, then the request will be routed to one of the upper outputs. But which upper output? There are at least two ways of selecting the choice: *random* and *partially-deterministic* selections. With random selection, a request is randomly granted one of the choices; while with the partially-deterministic policy, a request is forced to take the first choice, but if two requests want the same output, say the upper one, one gets the first choice and the other the second on a random basis. Another option in the function of a 2×4 SE is how it behaves after receiving a negative ACK. In *no re-routing mode*, the 2×4 SE simply returns a negative ACK to a 2×4 SE in the previous stage. In *re-routing mode*, the SE attempts to route the request through the other choice. If still unsuccessful (because the request has been blocked or there is a faulty SE in a further stage), a negative ACK is generated back to the previous stage.

3 Performance Evaluation

3.1 Assumptions about Switch Operation

Before the analysis, a few definitions are required. A *stage cycle* is the time interval during which a request (or an acknowledgement) moves from one stage to an adjacent stage. A *network cycle* is the time interval for a packet received at an IM to pass through to an OM. In terms of the switch's operation described above, one network cycle then consists of the sum of the path-establish and packet-transfer phases. Further, we assume here that a packet may only arrive at an IM at the beginning of a network cycle. In which case, one network cycle consists of a finite number of stage cycles[2]. Another assumption, which applies to the situation when a 2×4 SE performs re-routing, is best explained in terms of an example. Referring to Fig. 2, suppose that at the beginning of cycle 0, a request is sent from IM A to SE B. Then at the beginning of cycle 1, it is sent to the input of SE C and, after having been accepted, it is assigned to go to SE D during cycle 2. At the beginning of cycle 2, the request is received at the input of 2×2 SE D. If there is no other request arriving at the same cycle or there is one, but it requires the other output of the SE, the request will be accepted, and assigned to go to SE E during cycle 3. If unsuccessful, SE D will send a negative acknowledgement back to SE C

[2]Hereafter, the term "cycle" refers to the stage cycle, unless otherwise stated.

at the beginning of cycle 3; that is the request is still at SE C. In which case, SE C will attempt to re-route the request to SE G. But first it has to check whether the link between SE C and SE G is available or not. Suppose that the request is now at SE C during cycle 3, *i.e.* it was blocked at SE D, so an attempt will be made to re-route the request. If the required link is free, the request will be in a position to proceed again; it will be sent to the input of SE G at the beginning of cycle 4. If not, the request will stay at the same place during cycle 4 and then it will be sent back to SE B in cycle 5. Assume that the request is received at SE G at the beginning of cycle 4 and that its required output of SE G is already occupied. Then, a negative acknowledgement is sent back to SE C in cycle 5. In which case, the request will return to SE B. Implicit in the above description is that being in an SE in stage s refers to a situation where the request has actually been allowed to occupy the output of the SE. We have also assumed that re-routing cannot take place during the same cycle. This assumption, which was also adopted in [4], is needed to simplify the analysis, although in practice a request may be able to try an alternative path within one stage cycle.

3.2 Assumptions

The proposed switch is analysed under the following assumptions.

- Packets arriving at different IMs in the same network cycle are mutually independent.
- Packets arriving at an IM or at different IMs in different network cycles are mutually independent.
- The destination of a packet is uniformly addressed to one of the OMs with equal probability.
- Packets arrive at an IM in a given network cycle with probability P. This is the input rate of the switch.
- A blocked packet, after returning to its corresponding IM, is discarded without being re-submitted.
- The buffer size of an OM is infinite [2].
- Since there are two modes of selections of a 2×4 SE (random and partially deterministic), in this analysis it is assumed that the random selection is adopted in order to simplify the analysis. However, it has been shown by simulation that there is only a slight difference in terms of performance between the two modes when used with re-routing [3].

It was observed in [4, 5, 6] that the above assumptions taken together imply that the states of the IMs are statistically independent and SEs in the same stage or different stages also behave independently. It follows that it is sufficient to study only one SE per stage, as it represents the behaviour of all SEs in the same stage. The first two assumptions are unrealistic in practice but, it is shown in [6] by means of simulation, the results obtained are only slightly different with those two relaxed. The third assumption is a uniform traffic pattern, which is widely used when comparing different networks. The fifth assumption is in fact implicit in the second one. The sixth assumption is there to simplify the analysis. The performance measure used is the *probability of acceptance* (P_{A}) which is defined as the probability that an arbitrary request will be accepted [6, 7].

3.3 State Diagram Technique

As it is sufficient to study only one SE per stage, this then implies that we can focus on the progress of a single request through stages in the switch. In this context, we can make the observation that the next "state" of a request depends only on its current state and the current state of the switch. Thus, the progress of the request can be modelled by means of state digram. In the method of state diagrams, two things are required [8]: (1) a finite number of states in a diagram and (2) transition probabilities between states. In the state diagram, the future state of a system depends only on its current state. Once the above two parts are completed, one can then obtain the probability of being in any desired state. To apply the method of state diagrams to the analysis of a re-routing version of the proposed switch, it is appropriate that the state diagram for the analysis be a diagram whose

states are the "state" of a request in a particular stage (in a particular cycle) where the future state of the request (in the next cycle) only depends on its current state and the current state of the switch. So the task here is to first build a diagram with appropriate states, and then work out transitions probabilities among the states. Such a state diagram of an $l = 2$, $K = 4$, 16×16 version of the proposed switch is depicted in Fig. 3.

3.3.1 State Description

In Fig. 3, the states are identified by several components as follows:

- **Stage Number** (s) indicates the stage number where the request is.
- **Path Number** (p) indicates the path number a request is using. Recall that in the proposed switch, there exist 2^l paths (not disjoint) between every IM and OM pair. Here path number 1 is defined as the first path that a request takes in attempting to go its required OM, path number 2 is the second one and so on.
- **Cycle Number** (c) indicates the cycle number that a request is in.
- **Blocking Number** (b) indicates the stage number where a request has been most recently blocked.

Consider an l-level $N \times N$ switch with $n = \log_2 N$ stages and $K = 2^l$. By means of the above components, the state of a request can be uniquely identified and classified into four groups:

Set-up : Being in a Set-up state means that a request has so far been successful in establishing a path up to the stage where it is currently, and it will be trying to establish a path to the next stage during the next cycle. In terms of the physical switch, this refers to when a request is allowed to occupy an output of an SE in a given stage. For example, state 1 in Fig. 3 may be interpreted as a request now being at one of the outputs of SE B. Another example is state 7, which is a scenario when, after having been blocked previously, the request is able to be re-routed (*i.e.* set up). Set-up states can be further divided into three types as follows:

S_0 Request is at the IM and is being submitted to the first stage of the DN in cycle 1. This is state 0 in Fig. 3.

$S_{s,p,c}$ Request is in stage s, it is currently using path number p, it has not been blocked before, and it will be attempting to set up a path to stage $s+1$ in cycle $c+1$. These are states 1, 2, 3 and 4 in Fig. 3.

$S^b_{s,p,c}$ Request is in stage s, it is currently using path number p, it has been blocked most recently at stage b and it will be attempting to set up a path to stage $s+1$ in cycle $c+1$. Examples include states 7, 8 and 9.

Blocked : Being in a Blocked state has one of three meanings. The first refers to a situation when a request in stage s gets blocked when trying to establish a path to stage $s+1$; a negative acknowledgement is received at stage s. From Fig. 3, states 6, 14 and 18 belong to this category. The second is when a request, after having been blocked previously, is blocked again as a result of the other output in a 2×4 SE being unavailable in an attempt to re-route the request. Examples include states 12 and 26. The third is when a request has already been blocked at some stage ahead of where it currently is and it is on the way back to some stage in the DN from where it may be re-routed. These include states 30 and 33. A Blocked state is defined as below:

$B^b_{s,p,c}$ Request is in stage s during cycle c, it was blocked at stage b when attempting to set up a path to stage $b+1$ with path number p.

Hold : Being in a Hold state means that the request is at its required OM and a path is being held between its IM and OM for a number of cycles before the packet-transfer phase starts. Once it is in a Hold state, it will be in that state until the end of the path-establish phase. The Hold states are divided into two types:

$H_{p,c}$ Request has been successful in setting up a path from its IM to its requested OM by using path number p, and the path is being held between this source-destination. This is state 5 in Fig. 3.

$H^b_{p,c}$ Request is in its required OM after having been successful in setting up a path from its IM to the OM by using path number p, it has been blocked most recently at stage b and the path is being held between this source-destination. Examples include states 10, 24, 39 and 44.

Lost : When a request is in a Lost state, it is back at its originating IM after having failed on all paths. Once in a Lost state, it will be there until the last stage cycle of the path-establish phase. A Lost state is defined as below:

$L^b_{p,c}$ Request is at its originating IM after being unable to set up a path, the most recent path number used was p and it was most recently blocked at stage b. Examples include states 15, 19 and 45.

3.3.2 Transition Probabilities Derivation

Denote the probability that a request will be in state A by $P[A]$ and the transition probability from state B to state A by $P[A|B]$ where A and B are any of the states defined in the previous section and A is not the same state as B. The transition probabilities of each state are given and described below:

A. Request is currently in a Set-up state

$$P[S_{0,1,1}|S_0] = 1 \quad \text{(At an IM)} \tag{1}$$

(1) Request has not been blocked before

$$P[S_{s+1,1,c+1}|S_{s,1,c}] = 1 \quad (0 \le s \le l-2) \tag{2}$$

$$\left.\begin{aligned} P[S_{s+1,1,c+1}|S_{s,1,c}] &= (1 - P[S_{s,1,c}]/K) + 0.75(P[S_{s,1,c}]/K) \\ P[B^s_{s,1,c+1}|S_{s,1,c}] &= 1 - P[S_{s+1,1,c+1}|S_{s,1,c}] \end{aligned}\right\} \quad (l-1 \le s \le n-2) \tag{3}$$

$$P[H_{1,c+1}|S_{n-1,1,c}] = 1 \quad (s = n-1) \tag{4}$$

Eq. 1 states that whenever there is a request, the IM will always submit it to the first stage of the DN. $P[S_0]$ is also the input rate of the switch. Eq. 2 is derived from the fact that a request is always successful in establishing a path to a 2×4 SE in a stage of the DN because there are at most two requests coming to the inputs but there are four outputs available so there is always a free output. Eq. 3 comes from the fact that when a request is in stage s and wishes to set up a path via a 2×2 SE in stage $s+1$, it can do so if firstly there is no other request that will be arriving at this particular SE (the first term), or secondly there is another request that will arrive at the SE but it will want the other output of the SE that is not wanted by the considered request (the second term). The probability that there is another request arriving at the other input of the 2×2 SE under consideration is expressed as $P[S_{s,1,c}]/K$ since the random selection assumption is adopted. Eq. 4 means that a request in stage $n-1$ can always set up a path to its addressed OM as a result of the multiple acceptance assumption. This is state 4 of Fig. 3.

(2) Request has been blocked before

In order to derive transition probabilities for states that have been blocked before, the following terms are defined:

$$\mathrm{F1}(s,c) = \sum_{i=s}^{n-1}\sum_{p=1}^{K}\sum_{b=0}^{n-2}\Big(P[S^b_{i,p,c}] + P[S_{i,p,c}] + P[B^b_{i,p,c}] + P[H_{p,c}] + P[H^b_{p,c}]\Big) \tag{5}$$

$$\mathrm{F2}(s,c) = \sum_{p=1}^{K}\sum_{b=0}^{n-2} P[S^b_{s,p,c}] \tag{6}$$

Eq. 5 gives the probability that in cycle c, there is a request which may be in one of the stages starting from stage s and beyond (including the OM). In other words, it gives the probability that there are other requests ahead of the considered one. As in Fig. 3, suppose that we are considering state 7, then states that are ahead of it during the same cycle are states 4 and 18, *e.g.* $\mathrm{F1}(2,4) = P[S_{3,1,4}] + P[B^2_{2,1,4}]$. Eq. 6 gives the probability that, in stage cycle c, requests in stage s will be setting up a path to stage $s+1$ during the next cycle.

$$P[S^b_{s+1,p,c+1}|S^b_{s,p,c}] = 1 \quad (0 \le s \ \le l-2) \tag{7}$$

Eq. 7 states that when a request is blocked, and then is in a Set-up state again in a stage in the DN and is attempting to set up a path to the next stage, it will always be successful in doing so. Considering a request which is now in a Set-up state in stage s, this means that the request is at the output of a 2×4 SE in that stage and is going to one of the inputs of a 2×4 SE in stage $s+1$. Since there is at most one other request in the SE in stage $s+1$ that can be in competition, the request always gets through stage $s+1$. Examples include states 35 and 71.

$$\left.\begin{array}{rcl} P[S^b_{s+1,p,c+1}|S^b_{s,p,c}] &=& 0.75\big(P[S_{s,p,c}] + \mathrm{F2}(s,c)\big)/K \\ &+& 0.5\big(\mathrm{F1}(s+1,c)\big)/K \\ + \quad \big(1 - P[S_{s,p,c}]/K &-& \mathrm{F2}(s,c)/K - \mathrm{F1}(s+1,c)/K\big) \\ \\ P[B^s_{s,p,c+1}|S^b_{s,p,c}] &=& 1 - P[S^b_{s+1,p,c+1}|S^b_{s,p,c}] \end{array}\right\} (l-1 \le s \le n-2) \tag{8}$$

Eq. 8 is the probability that the request is successful in setting up a path to a 2×2 SE in the next stage. In attempting to set up a path to the next stage, the request has to encounter one of the three situations depicted in Fig. 4. Case (c) will occur when there is another request in the same stage which is also going to set up a path in the same cycle. This is expressed as $(P[S_{s,p,c}] + \mathrm{F2}(s,c))/K$. The probability that case (b) will occur is the probability that there is another request in some stage ahead of the considered one during the same cycle, and is expressed as $\mathrm{F1}(s+1,c)/K$. For case (a), this is the probability that both cases (b) and (c) do not happen, which is expressed as $(1 - (P[S_{s,p,c}] + \mathrm{F2}(s,c) + \mathrm{F1}(s+1,c))/K)$. These probabilities are then weighted with the pass probabilities for each case as shown in the Fig. 4. With regard to Fig. 3, an example transition from state 7 to state 8 has the probability of case (c) as $P[S^1_{1,2,4}]/K$, and the probability of case (b) as $\mathrm{F1}(2,4)/K = (P[S_{3,1,4}] + P[B^2_{2,1,4}])/K$.

$$P[H^b_{p,c+1}|S^b_{n-1,p,c}] = 1 \quad (s \ = n-1) \tag{9}$$

Eq. 9 means that the request always gets through the OM when it is in the last stage of a RN. State 9 is an example of such a state.

B. Request is currently in a Blocked state

$$\left.\begin{array}{rcl} P[S^b_{0,p+1,c+1}|B^b_{0,p,c}] &=& 1 - 0.5(\mathrm{F1}(1,c)) \\ P[B^0_{0,p_{\mathrm{new}},c+1}|B^b_{0,p,c}] &=& 1 - P[S^b_{0,p+1,c+1}|B^b_{0,p,c}] \end{array}\right\} \quad (s=0,\ p \ \neq K) \tag{10}$$

$$P[L^b_{p,c+1}|B^b_{0,p,c}] = 1 \quad (s=0,\ p=K) \tag{11}$$

Eq. 10 expresses the probability of a request being in a position to be re-routed via an alternate path (one of the outputs of a 2×4 SE in stage 0). This is the probability that a particular output of the

SE in stage 0 is not occupied. Recall that F1$(1, c)$ is the probability that, in cycle c, there is another request at some stage ahead of stage 0. In addition, there is 50% chance that this request has made use of the particular output required. An example is the transition from states 11 to 35. Eq. 11 applies when a request has failed on all paths, so it has to return to its IM; examples include states 15 and 19.

$$\left.\begin{array}{rcl} P[S^b_{s,p_{\rm new},c+1}|B^b_{s,p,c}] & = & 1-0.5(\mathrm{F1}(s+1,c)/(2^s)) \\ P[B^s_{s,p_{\rm new},c+1}|B^b_{s,p,c}] & = & 1-P[S^b_{s,p_{\rm new},c+1}|B^b_{s,p,c}] \end{array}\right\} \quad (1 \le s \ \le l-1,\ p \bmod 2^{(l-s)} \neq 0) \tag{12}$$

$$P[B^b_{s-1,p,c+1}|B^b_{s,p,c}] = 1 \quad (1 \le s \ \le l-1,\ p \bmod 2^{(l-s)} = 0) \tag{13}$$

Eq. 12 follows the same argument as Eq. 10 except that the probability that there is another request already occupying the required output is reduced by the factor of 2^s where s is the stage number. As an example, this refers to a transition from state 6 to state 7 in Fig. 3 in which the probability of another request already being ahead is the probability of its being in state 3 (*i.e.* a request in stage 2). Recall that there are 2^s planes in stage s $(l-1 \le s \le 0)$. As in Fig. 2, suppose that there is another request in stage 2, the 2×4 SE in stage 1 where this request has just successfully set up a path could have been either the one on the left plane or the other on the right. Thus, there is a 50% chance that this particular request has been occupying the same SE in which the considered request is going to be re-routed. Eq. 13 applies when a request has failed on all the paths that are available at a particular stage, thus it has to go back to a previous stage where it can try other paths. Examples include states 12 and 26.

$$P[B^b_{s-1,p,c+1}|B^b_{s,p,c}] = 1 \quad (l \le s \ \le n-2) \tag{14}$$

When a request is in a Blocked state at a stage in the RN, it simply returns to the previous stage.

C. Request is currently in a Hold state

(1) Request arrived at an OM without being blocked

$$P[H_{p,c+1}|H_{p,c}] = 1 \tag{15}$$

(2) Request arrived at an OM having been blocked

$$P[H^b_{p,c+1}|H^b_{p,c}] = 1 \tag{16}$$

D. Request is currently in a Lost state

$$P[L^b_{p,c+1}|L^b_{p,c}] = 1 \tag{17}$$

3.3.3 Derivation of $P_{\rm A}$ of a Proposed Switch

$P_{\rm A}$ can be computed by adding all the probabilities that the request is in a Hold state.

$$P_{\rm A} = \Big(P[H_{1,m}] + \sum_{p=2}^{K}\sum_{b=0}^{n-2} P[H^b_{p,m}]\Big)/P[S_0] \quad (c=m) \tag{18}$$

where m is the last cycle of the path-establish phase.

3.3.4 Example

Given below is an example of how to derive probabilities of some states in the diagram in Fig. 3. We assume that the input rate is 1.

Cycle 0;
$P[S_0] = 1$

Cycle 1;

$P[S_{0,1,1}] = P[S_{0,1,1}|S_0]P[S_0] = 1.$

Cycle 2;

$P[S_{1,1,2}] = P[S_{1,1,2}|S_{0,1,1}]P[S_{0,1,1}] = 1.$

Cycle 3;

$P[S_{2,1,3}] = P[S_{2,1,3}|S_{1,1,2}]P[S_{1,1,2}] = ((1 - (P[S_{1,1,2}]/4)) + 0.75(P[S_{1,1,2}]/4))(1) = 0.9375$

$P[B^1_{1,1,3}] = (1 - P[S_{2,1,3}|S_{1,1,2}])P[S_{1,1,2}] = 0.0625$

Cycle 4;

$P[S_{3,1,4}] = P[S_{3,1,4}|S_{2,1,3}]P[S_{2,1,3}] = ((1 - (P[S_{2,1,3}]/4)) + 0.75(P[S_{2,1,3}]/4))(0.9375) = 0.8826$

$P[B^2_{2,1,4}] = (1 - P[S_{3,1,4}|S_{2,1,3}])P[S_{2,1,3}] = 0.05493$

$P[S^1_{1,2,4}] = P[S^1_{1,2,4}|B^1_{1,1,3}]P[B^1_{1,1,3}] = (1 - 0.5(P[S_{2,1,3}]/2))(0.0625) = 0.0479$

$P[B^1_{1,2,4}] = (1 - P[S^1_{1,2,4}|B^1_{1,1,3}])P[B^1_{1,1,3}] = 0.0146$

Cycle 5;

$P[H_{1,5}] = P[S_{3,1,4}] = 0.8826$

$$P[S^1_{2,2,5}] = P[S^1_{2,2,5}|S^1_{1,2,4}]P[S^1_{1,2,4}]$$
$$= \big(0.75(P[S^1_{1,2,4}]/4) + 0.5(P[S_{3,1,4}] + P[B^2_{2,1,4}])/4 +$$
$$(1 - (P[S^1_{1,2,4}] + P[S_{3,1,4}] + P[B^2_{2,1,4}])/4)\big)(0.0479) = 0.0421$$

$P[B^1_{1,2,5}] = (1 - P[S^1_{2,2,5}|S^1_{1,2,4}])P[S^1_{1,2,4}] = 0.0058$

.

.

3.4 Simulation

Simulation was conducted in order to verify the analysis. The switch was simulated under the same assumptions as those adopted in the analytical model. Two independent runs were carried out, each consisting of five batches—altogether giving ten values to be averaged for each simulation point. Each batch consisted of 10,000 cycles. In each run, an additional 1,000 cycles were completed before data was collected in order to avoid transient affects. The 90% confidence intervals are very tight and therefore will not be shown. However, due to time limitations, only 16×16 and 32×32 switches were modelled.

4 Results and Discussion

Fig. 5 shows the P_A of the switch in re-routing mode as obtained from both analysis and simulation. It shows reasonable agreement between the results. In both cases though, the analytical results gave a higher P_A than the simulation, which we presume to be due to failure of the independence assumptions. Curves of P_A of the proposed switch in re-routing with random selection, no re-routing with partially deterministic selection, and a replicated MIN, as a function of the number of stages and the number of routing networks, are plotted in Fig. 6, with the results for the K-replicated MIN (multiple acceptance) taken from [2]. Clearly, Fig. 6 indicates that the proposed switch in re-routing mode offers a significant improvement in performance when compared to the no re-routing mode and the replicated MINs. A $K = 4$ re-routing version of the switch has a performance close to that of a no re-routing version with K=8. This comes as no surprise because with re-routing, a request has more than one way to get through to its required output.

5 Conclusions and Future work

In this paper, we have presented an analysis of a fault tolerant ATM switch design based on the use of a number of routing networks in parallel when re-routing is performed. The analysis is based on

the state diagram method. The results obtained from the analysis were compared with the simulation results. It was shown that they were in reasonable agreement. These results also show that with re-routing, the switch gives a much better performance when compared with no re-routing mode. There are a number of other aspects to be investigated in future. These include an analytical model for re-routing mode in the presence of faults and reliability analysis of the switch.

Acknowledgements

The research described in this paper was undertaken by the first author (supervised by the second) as part of a Ph.D. project in the Dept. of Electronics Systems Engineering, University of Essex. The first author was initially supervised by Dr. E.A. Medova, and is sponsored by the Thai Government.

References

[1] TOBOGI F.A.: 'Fast packet switch architectures for broadband integrated services networks', *Proceedings of the IEEE*, Vol. 78, 1990, pp. 133–167

[2] VENKATESAN, R. and MOUFTAH, H.T.: 'Performance analysis of multipath Banyan networks', *Proc. IEEE International Conference on Communications*, 1992, pp. 912–915

[3] SEGKHOONTHOD, S. and SINCLAIR, M.C.: 'Analysis of a fault-tolerant ATM switch based on a parallel architecture', *Proc. UK Teletraffic Symposium*, Old Windsor, March 1995, pp. 22/1–22/9

[4] PADMANABHAN, K and LAWRIE, D.H.: 'Performance analysis of redundant-path networks for multiprocessor systems', *ACM Trans. Computer Systems*, 1985, **3**, (2), pp. 117–144

[5] WU, C.-L and LEE, M.-L.: 'Performance analysis of multistage interconnection network configurations and operations', *IEEE Trans. on Computers*,1992, **C-41**, (6), pp. 18–27

[6] PATEL, J.H.: 'Performance of processor-memory interconnections for multiprocessors', *IEEE Trans. on Computers*,1981, **C-30**, (10), pp. 771–780

[7] KUMAR, M. and JUMP, J.R.: 'Performance of unbuffered shuffle-exchange networks', *IEEE Trans. Computers*, 1986, **C-35**, (6), pp. 573–577

[8] HAMMING , R.W.: *The Art of Probability for Scientists and Engineers*, Addison-Wesley, 1992

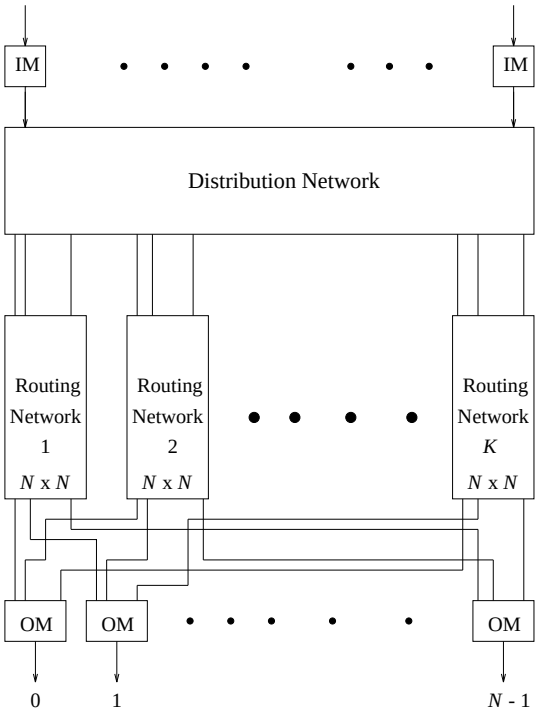

Figure 1: Configuration of the proposed switch

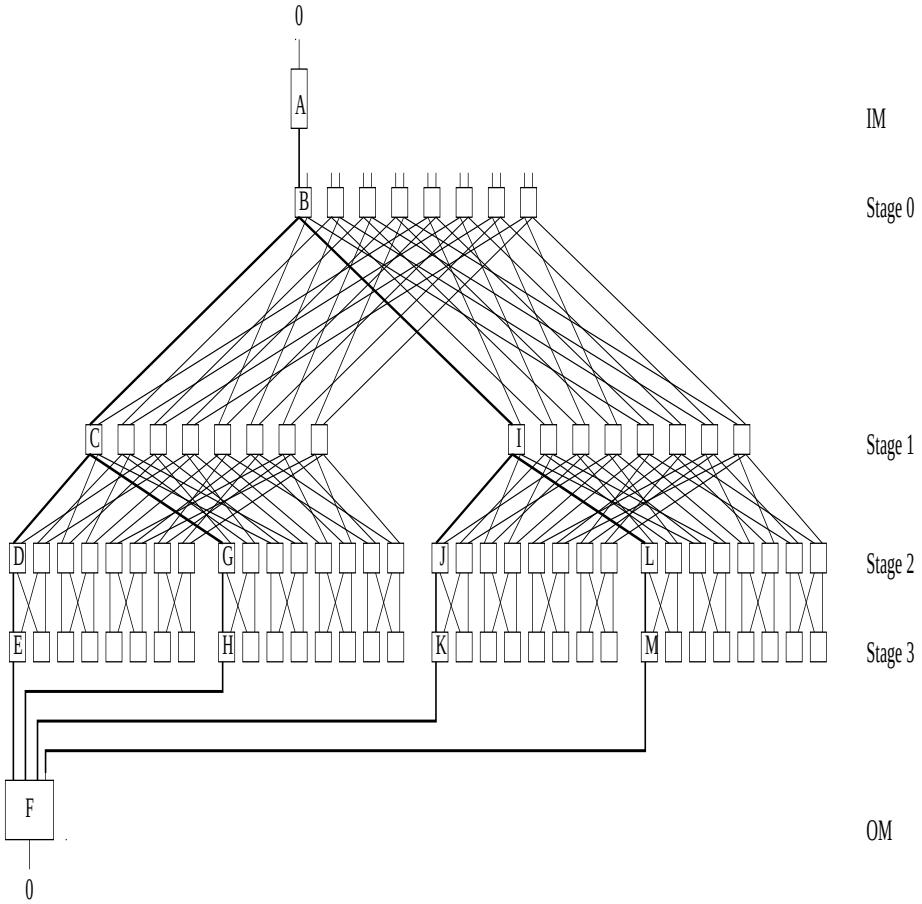

Figure 2: An $l = 2$, $K = 4$, 16×16 switch

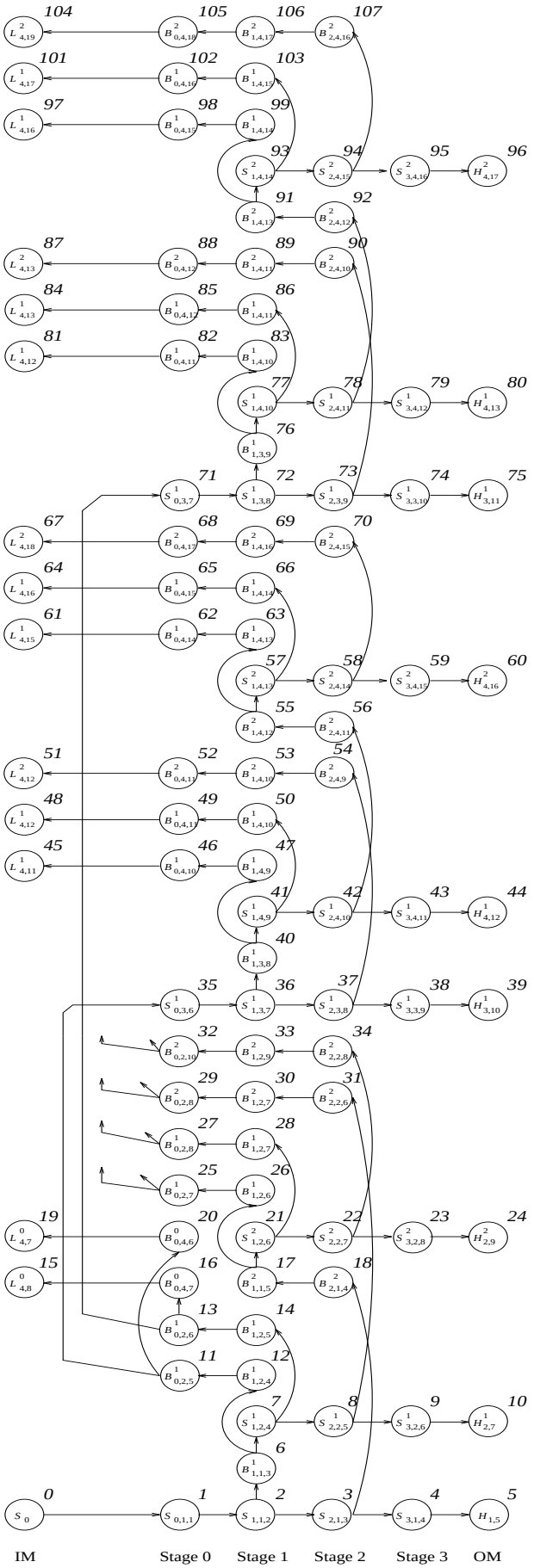

Figure 3: State Diagram of a $l = 2$, $K = 4$ 16×16 switch

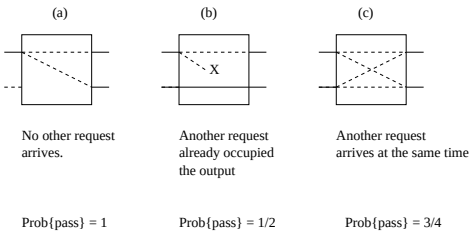

Figure 4: Probability of establishing a path in a 2×2 SE

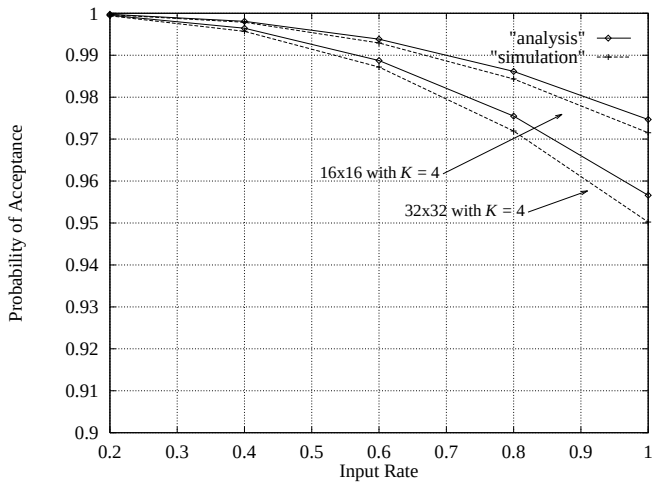

Figure 5: P_{A} of the proposed switch in re-routing mode (analysis and simulation results)

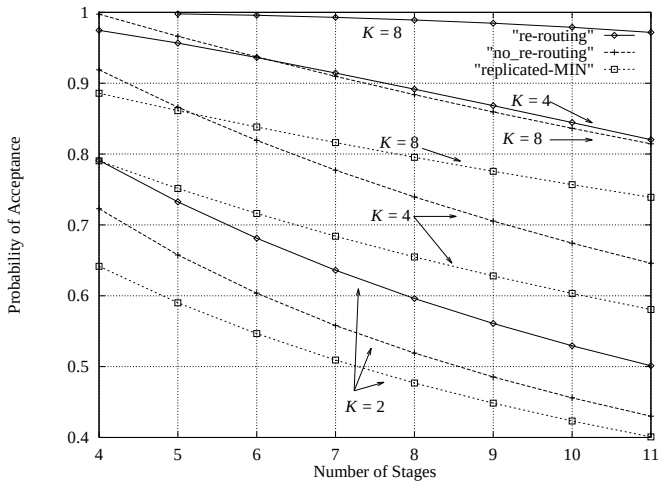

Figure 6: P_{A} of the proposed switch in re-routing and no re-routing modes and a replicated MIN (analytical results with Input Load = 1)