PERFORMANCE OF CLOS MULTICAST NETWORKS

GHAZI M. AI SUKKAR & JAMIL N. AYOUB
EE Dept., University of Jordan, Amman, JORDAN

Abstract: A three-stage circuit-switched Clos network, under various routing control strategies is considered for multicast operation. It was found that, under certain assumptions, this network become nonblocking as the number of middle switches increases. Also, it was demonstrated that blocking probability can be reduced by certain modifications applied to its links.

Key-Words: - Multicast, simulation, blocking probability, switching Clos networks.

1 Introduction

In many communication environments, we often need to support one-to-many connections, where one source is sending the same message to multiple destinations. Such connections are also called multicast. Multicast communication is highly demanded in broad-band integrated services digital networks (BISDN), which is based on ATM technology, and in scalable parallel computers.

Some examples of multicast applications are video conference calls and video-on-demand services in BISDN networks, and carrier synchronization and write update/invalidate in directory-based cache coherence protocols in parallel computers.

An important feature of multicast in communication networks is bandwidth saving. A hot topic nowadays, mobile IP multicast, found its way as an architecture that promises to reduce the bandwidth and delays of such networks to less than half of their values in current mobile IP architecture [1 -5].

In order to support circuit-switched communication networks with multicast connections, the switches in these networks must have multicast capability. One of the proposed switch designs which can support multicast is Multistage Switching Networks. In this paper, we will consider the issue of supporting multicast in the well-known three-stage Clos network which is a special type of Multistage Switching Networks.

Clos networks have been extensively studied for both unicast (one-to-one) communication and multicast communication. Masson et al. [3], showed that a nonblocking multicast Clos network requires a higher network cost than a unicast Clos network. The blocking behavior of multicast Clos network with only a comparable network cost to a unicast Clos network is given in [1,2].

In this paper, a simulator is developed to study the blocking probability of a three stage Clos network under various routing control strategies. Also, we suggest certain modifications on the way the switches are connected, where we added links between the switches in the same stage for the input and middle stages, and suggest a certain connection routing algorithm through these new links. Simulation results show that this modification is efficient in lowering the blocking probability.

2 Multistage Switching Networks

Multistage switching networks are composed of crosspoints that are usually grouped together into building-block subnetworks called switch modules. In an \((N_1 \times N_2)\) multistage switching network with \(N_1\) input ports and \(N_2\) output ports, the switching modules used, might each have, \(n_1\) input ports and \(n_2\) output ports, where \(n_1 < N_1\) and \(n_2 < N_2\). Full availability is assumed for these modules. The \((n_1 \times n_2)\) switch modules in a multistage switching network are interconnected by means of links.

A multistage switching network with an odd number of stages is called Clos network [7]. A three-stage Clos network has \(r_1\) \((n_1 \times m)\) switch modules in the first stage, \(r_2\) \((r_1 \times r_2)\) switch modules in the middle stage, and \(r_3\) \((m \times n_2)\) switch modules in the third stage. The network has exactly one link between every two switches in its consecutive stages. Such network is denoted as \(v(m, n_1, r_1, n_2, r_2)\). Fig.1 illustrates a general schematic of this network. For the special symmetrical case where \(n_1 = n_2 = n\) and \(r_1 = r_2 = r\), the three-stage Clos is denoted as...
A multicast connection in a multistage network can connect a network input port simultaneously to more than one network output port, but an output port can be connected to at most one input port at a time. The number of output switches in a multicast connection is referred to as the fanout of the multicast connection [2].

A maximal set of multicast connections between the inputs and outputs of a multistage network is referred to as a multicast assignment.

When a \( \nu(m, n, r) \) network is considered for supporting multicast, it is reasonable to assume that every switch in the network has multicast capability. It is clear that if a multicast connection from an input port to at most one output port on each of the output stage switches can be realized, then a multicast connection from that input port to more than one output port on the same output stage switches can also be realized. From that we will assume that each multicast connection from an input port will connect that port to at most one output port in each of the output switches.

3 Blocking in Clos Network

Clos networks have been widely used in various Interconnection problems. Some recent applications include the NEC ATOM switch designed for BISDN [8], the IBM GF11 multiprocessor [9], and the ANSI Fiber Channel Standard for interconnection of processors to the input/output (I/O) system. More recently, Bruggencate et al. [10] showed that the network in the IBM SP2 [12], is functionally equivalent to a Clos network. Nonblocking Clos networks have been studied for both unicast communication (where a unicast network is sometimes referred to as a permutation network) and multicast communication. These studies showed that a nonblocking multicast network requires a much higher network cost than a permutation network.[12-17]

Recently Yang et al. [1] proposed an analytical model for the blocking probability of the Clos type multicast network. This model is concerned with that whether there exists a multicast tree from the source node to all the destination nodes of a new multicast connection. The blocking probability is calculated for the entire multicast tree (i.e., one-to-many blocking probability), and was shown that for a \( \nu(m, n, r) \) network:

\[
P_B = \frac{1}{r} \sum_{i=1}^{r} \sum_{k=0}^{\frac{m}{r} \cdot \frac{n}{i} \cdot \frac{m}{k}} q_i p_a^{m-k} \left(1 - \left(1 - p_b^k \right)^f \right) \text{.} \tag{1}
\]

Where \( p_b = \frac{an}{m} \) is the probability that a middle-output interstage link is busy, \( q_b = 1 - p_b \) is the probability that this link is idle, \( p_a = \alpha \cdot p_b \) is the probability that the input-middle interstage link is busy, \( q_a = 1 - p_a \) is the probability that this link is idle, and

\[
\alpha = \frac{1}{r} \sum_{i=1}^{r} \sum_{f=0}^{\frac{m}{r} \cdot \frac{n}{i} \cdot \frac{1}{m}} \left(1 - \left(1 - \frac{1}{m} \right)^f \right) \text{ which is less than 1.}
\]

Both the analytical and simulation results show that a network with a comparable cost to a permutation network is almost nonblocking for multicast connections and can provide cost effective support for multicast communication, and showed that routing control strategy can provide performance improvement.

4 Simulation of multicast networks

First we study the blocking behavior of a Clos multicast network \( \nu(m, n_1, r_1, n_2, r_2) \) under different routing control strategies. Then we study the blocking behavior of a modified Clos multicast network \( \omega(m, n_1, r_1, n_2, r_2) \) when certain links are added to the network switches.

Our model is blocked calls lost, and our simulator is a discrete event simulator, which means that events occur according to a clock that can be advanced by one time unit. The system is thus
updated every one time unit.

4.1 simulator Assumptions:
- The arrival process of multicast connection requests is Poissonian, and the holding time of a connection follows an exponential distribution.
- Selection of all simulation parameter variables for a multicast connection including the input port, the fanout, and the output switch set are determined randomly following a uniform distribution.
- Four routing control strategies $S_i$, $i = 1, 2, 3, 4$, for selecting a middle switch were defined and tested:
  - $S_1$ = Smallest relative cardinality strategy: choose the middle switch whose destination set has the smallest cardinality with respect to the connection request.
  - $S_2$ = Largest relative cardinality strategy: choose the middle switch whose destination set has the largest cardinality with respect to the connection request.
  - $S_3$ = Largest absolute cardinality strategy: choose the middle switch whose destination set has the smallest cardinality.
  - $S_4$ = Largest absolute cardinality strategy: choose the middle switch whose destination set has the largest cardinality.
- If $M_j$, $j \in \{1, 2, \ldots, m\}$ denote the subset of output switches to which middle switch $j$ is currently providing connection paths from the input ports, $M_j$ is referred to as the destination set of middle switch $j$. Clearly, the complement of this set $\bar{M}_j$ is the set $M_j$ that contains the output switches available for the middle switch $j$.
- During simulated network operation, a workload, $\mu$ is maintained defining network utilization, that is, the ratio of busy output ports to the total number of output ports.

4.2 Connection algorithm for $v(m,n,r)$
1. Choose randomly one of the idle input ports and decide to which input switch it belongs.
2. Choose the fanout and the set of output switches in this connection request, called the multicast set.
3. If there is no available middle switches for this connection request, then advance the blocking counter by one and terminate the connection process. Else go to the next step.
4. Scan the available middle switches excluding those with no idle output ports, that is those with $M_j$ empty. If there is still available middle switches and the multicast set is not empty then go to the next step, else go to step 8.
5. Choose a middle switch with at least one idle output from the available middle switches for the connection request according to some control strategy. If no such middle switch exists, then advance the blocking counter by one and terminate this connection process. Else go to the next step.
6. Realize the largest possible portion of the connection request in the middle switch chosen in the previous step, by intersecting the multicast set with the idle output ports set of this middle switch. The result represents the portion of the multicast set that this middle switch can realize. Then this portion is excluded from the multicast set. Also, this middle switch is excluded from the available set of middle switches.
7. Go back to step 4.
8. If the multicast set is not empty then advance the blocking counter by one and terminate this connection process. Else go to the next step.
9. Save the input-middle interstage links and the middle-output interstage links used in this connection request, and update the network state.
10. Continue the connection process (i.e., choose the output ports in each of the output switches in this connection request, and generate the holding time...).

4.3 Modified Clos Network
The high blocking probability and the high number of middle stage switches needed for a Clos network to be nonblocking for multicast connections, calls for a way to lower these two related variables. We propose a modification on the Clos network, which is represented by connecting the input switches to each other by using additional links, and also connecting the middle switches to each other. We refer to this network as the modified Clos network, denoted by $\omega(m, n_1, r_1, n_2, r_2)$. Fig. 2 shows a generalized modified multicast network. These added links are dedicated links, and are referred to as side links. One way of realizing these side links between the switches is by adding new input and output ports for each switch and connect these ports.

As we see for each switch, we have to add two input ports: one port for the link that comes from the upper switch and the other for the link that comes from the lower switch. Similarly, two output ports need to be added.
4.4 Connection algorithm for $\omega(m, n, r)$

1. The first step after generating the multicast connection request (i.e., choose the input port, and the multicast set), is to make a test on the input-middle interstage links that connect the input switch associated with that connection request with the middle switches. If there is no idle input-middle interstage links available for this input switch at the current network state, then we try to route this connection request through the side links to one of the two input switches located at the two sides of this switch. Else if there is available input-middle interstage links for this input switch we try to realize the connection request through them with no need to route this connection request through the side links, and we continue the connection process by going to step 3 directly without entering the next step.

2. Since the input-middle interstage links associated with the input switch from which the connection request is originated are busy, we scan the side links, to see if we can route this connection request through one of the two switches near this input switch. We face one of the following cases:
   - Both of the side links are busy; in this case a blocking occur and the connection process is terminated.
   - Just one of the two side links is idle; in the current state, then we make a new test in the input-middle interstage links of the switch which the side link is connected to. If the input-middle interstage links are all busy, then a blocking occur, else we try to use these links for connection request.
   - If the two side links are idle, then we choose one of the two input switches, which these links are connected to. We choose the one that has an idle input-middle interstage links, if both have idle input-middle interstage links, then we choose the one which has the largest number of these links, if they have the same number we choose one of them randomly. If none of them has idle input-middle interstage links then a blocking occur.

3. Now the available middle switches for this connection request become these middle switches which are directly connected to the input switch through the idle input-middle interstage links, and also the middle switches which can be reached by the directly connected middle switches through the side links.

4. We continue with the same steps as in the connection process algorithm for the Clos network, including Steps 4, 5, 6, 7, 8, and 9. But in step 9 we must make some test before realizing the connection request. This is because some of the middle switches that are selected to realize the connection request may be one of that switches which are not directly connected to the input switch, and the side links that are providing connection to these links must be saved i.e. considered busy during this connection request.

A routing algorithm must be applied in the case that the middle switch that is chosen to realize the connection is not directly connected to the input switch, and we can reach this switch through two side links (two middle switches). We apply this routing algorithm in order to find the route through which we realize the connection. So we apply the following algorithm:
   - We choose the route through the side switch that has the largest number of idle middle-output interstage links (if this side switch is chosen for realizing the connection, then we calculate the number of idle middle-output interstage links after realizing the connection).
   - If both of the side switches have the same number of idle middle-output interstage

Fig. 2: A general schematic of the Modified Clos network.
links, then we choose the one that is already chosen to realize the connection.
- If both of them are already chosen to realize the connection, then we choose one of them randomly.
- If none of them is chosen to realize the connection then we also choose one of them randomly.

5 Simulation Results
To test our simulator, the symmetric Clos network \( \nu(32, 32, 32) \) was considered first. Fig. 3 shows the blocking probability result of this simulation along with the analytical model result as given in Eq. (1). Obviously, the simulation result with the \( S_2 \) strategy is in good agreement with the analytical model. Also, when a \( \nu(64, 64, 64) \) network was simulated, the results were again in agreement with the results given in [1]. This result shows that \( S_1 \) is the best strategy, and that the multicast network tends to be nonblocking with increasing \( m \), the number of middle switches.

Now we present the performance results of the simulated modified Clos multicast network \( \omega(m, n, r) \). Two configurations: \( \omega(m, 32, 32) \) and \( \omega(m, 64, 64) \) were studied using as the number of simulated connection requests to be 5000 and 10000 respectively. Results are shown in Figs. (4 – 8 ). Fig. 6 shows a comparison between the Clos network \( \nu(m, 32, 32) \) and the modified Clos network \( \omega(m, 32, 32) \) for the case of \( S_1 \) routing control strategy. From these results we can observe that the modification we added to the Clos network results in a lower blocking probability. The effect of loading on blocking is shown in Fig. 5.

6 Conclusions
Simulation results showed that for a Clos multicast network \( \nu(m, n, r) \), as the number of middle switches \( m \) increases, the network tends to be nonblocking. Also, results show that routing control strategies are effective for reducing the blocking probability, and that the Smallest relative routing strategy leads to the lowest blocking probability. These observations also apply to the modified Clos network \( \omega(m, n_1, r_1, n_2, r_2) \). In addition, results show that our modification on the Clos network is efficient in lowering the blocking probability, despite the fact that it results in more complex routing process. Such networks with a comparable cost to a permutation network can provide cost-effective support for multicast communication.
Fig. 6: \(\nu(m, 32, 32)\) and \(\omega(m, 32, 32)\) networks compared under \(S_1\) control strategy, with \(F=32\), \(\mu=75\%\), and \(h=120\).

Fig. 7: Blocking probability vs. \(m\) for \(\omega(m, 64, 64)\), with \(\mu=75\%\), \(F=64\), and \(h=120\).

Fig. 8: \(\nu(m, 64, 64)\) and \(\omega(m, 64, 64)\) networks compared under \(S_1\) control strategy, with \(F=64\), \(\mu=75\%\), and \(h=120\).

References:


