June 1991

On Call Migration

Ming Chit Tam
University of Pennsylvania

On Call Migration

Abstract

In an environment where network resources are reserved e.g, telephone networks, the path with smallest number of hops is preferred and other alternate paths are used only when there the shortest path is full. However if the alternate path is longer more network resources are devoted to the circuit and this in turn could worsen the situation. *Circuit migration* is a solution to reduce the amount of resources inefficiently used due to alternate routing in connection oriented networks. By rerouting a circuit when its shortest path becomes available, one can smooth out the congestion and increases the utilization of the network. The overhead of circuit migration is comparable to call set up and the tradeoff of circuit migration is improvement in performance vs. some additional call processing capacity. In this report we will focus on the above tradeoff, evaluating it analytically and by simulation on a completely connected topology. Our initial results indicate that migration could improve the performance of the network at high load but it has to be done very often. Such a large amount of overhead could be expensive enough to offset the gain in performance. On further investigation, we discover that threshing can also occur in circuit migration. We proposed two solutions to the problem. The first solution is to migrate only when the shortest path is no longer highly utilized. The second solution migrates a circuit only if its path is congested. A hybrid solution using the two above is also examined. We will also address the reordering problem that could occur when a circuit is transferred to a new path.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-91-51.

On Call Migration

MS-CIS-91-51
DISTRIBUTED SYSTEMS LAB 6

Ming Chit Tam

Department of Computer and Information Science
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA 19104-6389

June 1991

On Call Migration [*]

Ming Chit Tam

Distributed Systems Lab.

University of Pennsylvania

Philadelphia, PA 19104.

[*]This research is funded by the CNRI gigabit effort and a grant from AT&T Bell Laboratories

Abstract

In an environment where network resources are reserved e.g, telephone networks, the path with smallest number of hops is prefered and other alternate paths are used only when there the shortest path is full. However if the alternate path is longer more network resources are devoted to the circuit and this in turn could worsen the situation. **Circuit migration** is a solution to reduce the amount of resources inefficiently used due to alternate routing in connection oriented networks. By rerouting a circuit when its shortest path becomes available, one can smooth out the congestion and increases the utilization of the network.

The overhead of circuit migration is comparable to call set up and the tradeoff of circuit migration is improvement in performance vs. some additional call processing capacity. In this report we will focus on the above tradeoff, evaluating it analytically and by simulation on a completely connected topology.

Our initial results indicate that migration could improve the performance of the network at high load but it has to be done very often. Such a large amount of overhead could be expensive enough to offset the gain in performance. On further investigation, we discover that threshing can also occur in circuit migration. We proposed two solutions to the problem. The first solution is to migrate only when the shortest path is no longer highly utilized. The second solution migrates a circuit only if its path is congested. A hybrid solution using the two above is also examined. We will also address the reordering problem that could occur when a circuit is transferred to a new path.

1 Introduction

1.1 Circuit Migration

In connection oriented networks, routing decisions are often made dynamically according to the network status in order to fully utilize the resources and improves service quality. Since network status changes all the time, what might be a good path at some point may no longer be good later. Conventional wisdom said that once a new connection has chosen a path it uses the path until the session finishes. This approach makes it difficult to optimize the usage of network resources all the time.

For example, in networks where resources are not reserved, the selection of a route amount all the possible paths for a connection is influenced by the utilization of the links along these paths. The aim is to balance the load of the networks so that no resource is particularly congested. Congestion in this case could lead to extensive queuing delay and packet loss. Since there is no admission control, the QOS(Quality Of Service) is not guaranteed. As the status of the network changes, a chosen path may no longer offer the smallest delay or least level of packet lost. If a connection must follow the same path throughout a session, it would not be possible for the connection to take advantage of a smaller delay or smaller level of packet lost on the other paths. Worse yet, if the QOS offered by the current path deteriorates then the delay of the circuit will increase and so as the level of packet loss. If a connection can be migrated from one path to the other, this situation could be avoided. In section 2, we will survey some of the work that has been done with this environment in mind.

In very high speed networks where many applications are real time and require the QOS to be guaranteed, resource reservation is important. A path is said to be unavailable if it cannot accommodate the service requirement of the incoming call while maintaining the QOS of the existing calls [Decina 90]. With the notion of call blocking and equivalent bandwidth, the situation is very similar to circuit switching in telephone networks. Indeed many researchers feel that the very same set of analytical techniques can be reused. In these networks, one of the popular routing metrics used is hop count as opposed link utilization. Shortest hop routing is adapted so as to minimize the use of network resources. However shortest paths may not always be available and other paths are selected in these case. In the rest of the paper, this routing policy is referred to as **alternate routing**. In this paper **primary path** is used interchangebly with shortest path and **rerouting** is also used interchangebly with migration. Alternately routed circuits are circuits that use paths other than the shortest paths. Routing calls on longer paths means that more resources are used and other calls may be blocked. The situation can deteriorate and trunk reservation [Ash 81] is often used to limit the extend of alternate routing. In trunk reservation the last m trunks of a link is reserved for calls that use the link as part of the shortest path. Thus a call that requests use of the link as part of an alternate route is rejected if the number of free trunks on the link is less than m. However, once a circuit is assigned a path it will stay there until the session finishes. Migrating a circuit from a longer path to a shorter path would free up the extra resources used and hence reduces the call blocking rate.

In fact whether we use link utilization or hop count as a routing metrics they can all be abstracted as path costs. The structure of the algorithm in the routers is basically the same, i.e when new network status information comes the router would decide on whether to reroute some of the existing circuits. If there are more than one candidate, the decision could be made based on improvement of path costs. Under this general framework, one can select the circuit using the most congested path or the circuit using the longest path in hops. By migrating circuits from high cost path to low cost path, circuit migration may either improve the load balancing of the network or reduce the overall resource requirement of circuits. This could lead to smaller delay, lower level of packet loss rate or larger number of calls taken.

This paper investigates the potential tradeoff of circuit migration in an environment where resources must be reserved at the call set up time and a call is blocked if the network does not have enough resources. We would like to know how effective is circuit migration in increasing the number of calls accepted and what is the price of doing so. The tradeoff is the increase in the number of calls taken vs. the number of connections needed to be migrated. We looked at the variation in performance when circuit migration is used together with trunk reservation. Our analysis was indicative on the performance but the tradeoff was not clear until the simulation was done. From the initial set of results we found that it was very important to put some constraint on circuit migration in order to achieve at a good tradeoff. We proposed two ways to constrain migration and their effectiveness were investigated through simulation. Other issues investigated were the problem of packet reordering during the migration and migration mechanisms. The problem of circuits competing for migration would be addressed briefly.

In the rest of this section, we will describe what is involved in a migration and how circuit recovery can be easily accommodated by circuit migration. We will also describe the network model used in this paper. Section 2 surveys previous work in circuit migration, more background on circuit migration is also provided. We perform an approximate analysis on the performance of circuit migration in section 3. Section 4 describes the simulations and their conclusions. In section 5, we will discuss in more detail how to migrate a circuit and investigate the amount of reordering buffered needed. Section 6 concludes the paper.

1.2 Our Network Model

In this section we will describe our network model in general and give the reason for choosing a completely connected topology. Details of the simulation is given in section 4. The network model that we had in mind is shown in figure 1. The routing network consists of a number of switches connected by links. Apart from switching packets, a switch is also responsible for routing calls for users that are attached to the switch. We call this part of the switch the router. We assume that routing decisions are made at source routers but it is up to the switches to accept or reject a connection attempt. The network may use source routing or table routing by setting up virtual circuit tables at each switch. All the calls are homogeneous with a bandwidth requirement of one channel and an exponential holding time. Thus resource reservation is done by decrementing the free channel count of the links. Both static and dynamic routing are used. In static routing a list of paths is sorted in ascending order according to their hop count. In dynamic routing the shortest path is tried first with the alternate paths forming the rest of the list. When the shortest path is not available the cost of each alternate path is evaluated and the one with the lowest cost is chosen. The cost function is given as follow :

$$cost = \sum_{j=1}^{LengthOfPath} \frac{1}{ResidualCapacityOfj} + 1$$

Since dynamic routing performs better than static routing, we will only discuss the results of dynamic routing.

In addition to routing new calls, a router is also responsible for rerouting calls that it has set up. When a router receives some information about the status of the network, it examines the set of circuits that is alternately routed and decide whether migration is possible. Typically, migration occur when the router finds that the shortest path of one or more of its alternately routed circuit is available. If migration is possible then it will select one of the circuits to reroute. In practice it is possible for a router to select a number of circuits to be rerouted onto the same path and perform the rerouting in bulk (bulk reroute). The resources released by a migrated circuit could be used to further migrate other circuits, i.e, **cascade migration**.

In our environment since any alternately routed circuit is a candidate for rerouting, the list of candidate circuits to be rerouted does not have to be recomputed every time the status information comes in. However, the selection from the list may be based on some dynamic information such as the load of the paths that the circuits are using or the age of the circuit etc.. As mentioned below if the status information reflects a link failure then recomputation must be triggered and failed circuits are added onto the list of circuits to be rerouted.

The effectiveness of a routing strategy is dependent on the topology of the network. Since there is no benchmark topology for testing routing strategies, we had chosen a completely connected topology as it is easy to analyze. There is also another good reason for doing so, i.e, it allow us to isolate the effect of routers competing to migrate their own circuits. Competition for migration occurs because in practice, when each router receives the network status information it has to decide whether it should reroute any of its circuits. Just as routers may all favor a certain set of links in dynamic routing and direct the newly arrived traffic there, the same could also happen in circuit migration. A number of routers may receive the new network status and decide to reroute their circuits to use the idle capacity. They may start the rerouting process only to find that the idle capacity has been grabbed by some other routers and abort the processes. This sort of competition could mean unnecessary network traffic and processing load on the network. In our studies, only routes of one and two hops were used and hence if a call cannot use the primary path it will try to use one of the three two hops path. The circuit may then be migrated to the one hop path later. By considering only a completely connected topology where each node pair has a disjoint shortest path, we have eliminated router competition. The amount of overhead due to routers competition is topological dependent and we plan to address this problem in the second stage of our research. We note that telephone networks are configured to be completely connected. In end of section 4 we will suggest some possible solutions to reduce the competition when an aribitary topology is used without sacrificing the performance improvement of circuit migration.

1.3 Circuit Migration And Circuit Recovery on Link/Nodes Failure

It is possible to use the circuit migration mechanism to recover virtual circuits when there are link or node failures. Suppose a link in the network went down, when the link status information arrives at the routers, the recomputation of routes for each node pair would be triggered. The circuit would now be using a path with cost of infinity, i.e, any feasible path with the required capacity would have a smaller cost. The router put these circuits onto the list of alternately routed circuits and the migration mechanism will migrate these circuits onto new paths. These failed circuits will be selected whenever possible since their path costs must be the highest.

2 Previous Works

[Girard 83] considered a circuit switching environment in which the effect of circuit migration on the number of calls accepted was investigated. Call migration improves the throughput of the network in each case, on average there is a 1 to 2% improvement. However none of the examined routing schemes uses trunk reservation. The tradeoff of circuit migration was not investigated. The results are conservative since cascade migration was not considered. In this paper, we will investigate all of these and perform an approximate analysis.

[Humblet 86] suggested migration in the CODEX network where the length of a path is measured in terms of the switching delays rather than by hops. A circuit is rerouted if the cost that it adds onto the current path (assuming that circuit is taken off) is bigger than the cost that it adds to some other path. The rerouting is initiated at the receiving end of the circuit. Unfortunately no figures was given and we do not know to what extend does the scheme improves the delay. The potential problem of path oscillation was only briefly addressed. It is not clear when to reroute and when not to.

[Hwang 91] looked at the circuit migration in an environment similar [Humblet 86], i.e, resources are not reserved and the aim of rerouting is to reduce the packet delay of connections. Each node pair is assumed to have a number of equal length paths that are completely disjoint. The cross traffic on each link in these paths are assumed to be the same. Only homogeneous virtual circuits were considered. The paper then focused on the case of a single node pair with a certain number of paths and try to balance the load between these paths through rerouting. Two metrics are used to measure the network status, namely the number of virtual circuits on each path and the number of unacknowledged packets on each path. The simulation was done by using the two metrics separately. A circuit is initially routed using on the lowest cost path and can be rerouted later. Notice that when the number of virtual circuit is chosen as the metrics, the number of virtual circuit on each path belonging to the same node pair would never be differ by more than one. It is found that circuit migration reduces the delay only by a very small percentage in both cases. In general it is better to use the virtual circuit count as a routing metrics. [Hwang 91] investigated circuit migration under a very different set of assumptions from what we are interested in.

3 Analysis

In this section we will modify Krupp's model and provide an analysis on the performance gain of circuit migration used together with trunk reservation as a joint scheme. [Krupp 82] suggested an analytical approach to model the effect of alternate routing and trunk reservation. It is possible to modify the analysis to predict the performance of migration bearing in mind that had the actual modeling been used the underlying markov chain would be very different and could involve solving a large number of state equations. In the rest of this section we will follow the derivation from [Schwartz 87].

3.1 Krupp's Model

Consider the completely connected topology in figure 1. Let A be the point to point load and a be the total traffic offered on the link. A is defined by $\frac{\lambda}{\mu}$ where λ is the arrival rate of call with a poisson distribution and μ is the average duration of a call with exponential distribution. Thus

a includes the point to point load A and also those offered as a result of using the link as part of the alternate route. A is related to a by $A = a(1 - r)$. Assume the last N-m channels of a trunk is reserved for primary traffic then by modifying the Erlang-B formula we have :

$$p_j = \frac{a^j}{j!} p_0 \quad 0 \leq j \leq m \tag{1}$$

$$p_j = \frac{a^j (1-r)^{j-m}}{j!} p_0 \quad m < j \leq N \tag{2}$$

where p_j is the probability that the occupancy of the link is j and N is the capacity of the link. Thus when less than m trunks are used, the load on the link is a and above that only calls using the link as primary path can attempt. Since

$$\sum_{j=0}^{N} p_j = 1$$

therefore

$$\frac{1}{p_0} = \sum_{j=0}^{m} \frac{a^j}{j!} + \sum_{j=m+1}^{N} \frac{a^j (1-r)^{j-m}}{j!}$$

The throughput K of the link is then given by :

$$K = \sum_{j=0}^{N} j p_j \tag{3}$$

Assuming that q is the probability that a link will accept a call that use it as part of an alternate route, then q is :

$$q = \sum_{j=0}^{m-1} \frac{a^j}{j!} p_0 \tag{4}$$

Finally by substituting (2) and (4) into (3) and after some manipulation we get :

$$K = A(1 - p_N) + arq \tag{5}$$

Here q and p_N are all functions of r and a hence we still need one more constraint to solve for a for a given A. This is supplied by the observation that K is also equal to :

$$K = A(1 - p_N) + 2Ap_N[1 - (1 - q^2)^M] \tag{6}$$

Equation (6) comes from the observation that the probability of a call being accepted by the primary route is $(1 - p_N)$ and the probability that it is rejected by the primary route and accepted by one of the secondary route is

$$p_N[1 - (1 - q^2)^M]$$

where M is the number of alternate routes. The 2 comes from the fact that each alternate path consists of two hops and hence if an alternate path is taken the load applied onto the network is doubled. Thus by equating (5) and (6) we arrive at the following result :

$$A = \frac{arq}{2(p_N - \hat{z})} \tag{7}$$

where $\hat{z}$ is the blocking probability of a call.

$$\hat{z} = p_N(1 - q^2)^M$$

By using the secant method to solve A for a given a, it is possible to find $\hat{z}$. Finally, the amount of offered load accepted is given by :

$$C = A(1 - \hat{z}) \tag{8}$$

It is very difficult to construct a markov chain similar to the one above that incorporates circuit migration. The reason is that the departure rate of circuit on the link would depend on how many of these circuits are primarily routed and how many of them are alternately routed. Those that are primary routed have an individual departure rate of μ and those that are alternately routed has a sum departure rate of $N\mu$ per node pair. Thus the total departure rate of the circuits on a link cannot be expressed as a simple function of its load. Moreover, the arrival rate of circuits at state N-1 is certainly not simply λ since alternately routed circuit is rerouted to the primary path whenever the primary path is not full.

3.2 An Approximation

Instead of constructing a new model and solve it, we would try to arrive at an approximation by modeling the amount of time T that an alternately routed circuit would spend on the alternate route before rerouting to the primary path. T includes the time when an alternately routed circuit starts on the alternate path till a rerouting decision is made. It does not include the additional amount of time spend during the migration. When normalized with the average service time, the value represent the portion of time when two links are used. This value can then be used to modify equation (6) by replacing the constant 2.

$$K = A(1-p_N) + 2Ap_N[1-(1-q^2)^M]$$

In additional to being simplier, this approach also allows us to model the effect of migration delay on the overall performance. When migration occurs, the circuit is kept on the old path until the new path is set up. Therefore a circuit may have to spend additional time on its old path before it can be migrated. By adding this amount of delay to T during our computation, we can analysis the delaying effect of the migration mechanism. The value T can be found by modeling the migration process. Recall that a call is alternately routed only when the primary path is unavailable. Our assumption is that whenever a channel of the primary path becomes available, one of the alternately routed circuit would be rerouted. Neglect the effect of rerouting calls from our primary path, the rate in which circuits leave the primary path is $N\mu$. Hence the alternately routed circuits are rerouted at a rate of $N\mu$ to the primary path. Therefore the alternately routed circuit form a queue with the primary path acting as the server with service rate of $N\mu$. For simplicity, assume the queue is served FIFO, i.e the oldest alternately routed circuit is always rerouted first. This assumption does not invalidate our results for other queuing discipline eg., LCFS, ie., the most recently set up candidate is rerouted first. The reason is that we are only interested in T the average waiting time and this value is the same for many queuing disciplines. The arrival rate to the queue is just the arrival rate of calls after the primary path is blocked and is simply λ. Thus we have an M/M/1/N queue as shown in figure 2. The queue is finite since the number of circuits that can be alternately routed is bounded and is dependent on the load of the other links. Its size can be approximated by assuming that the size of the queue to be an unknown n and equate the lost probability of the queue with the probability that a call would not be accepted by any alternate route. The state equation for this $M/M/1/N$ system is as follow.

$$p_i = (\frac{A}{N})^i p_0$$

The value n can be found by equating

$$\frac{(\frac{A}{N})^n}{1+\sum_{i=1}^{n}(\frac{A}{N})^i} = (1-q^2)^M$$

to the nearest integer. We solve for an integer n such that the L.H.S and R.H.S are closest. Now that all the properties of this queue is known, we can use it to find the average system time T which is given by the Little's formula :

$$rT = \bar{n}$$

where $\bar{n}$ is the average number of alternately routed circuits and τ is the throughput of the rerouting process:

$$\bar{n} = \sum_{i=1}^{N} i p_i$$

$$\tau = \sum_{i=1}^{N} N p_i \mu$$

The average waiting time for an alternately routed circuit to be rerouted is :

$$S = T - \frac{1}{N\mu}$$

thus by normalizing S and substituting this into the equation (6) we have :

$$K = A(1 - p_N) + p_N(1 - (1 - q^2)^M)(1 + S\mu) \tag{9}$$

Since T depends on q which is in turn a function of A at the beginning of each round we increment a by a fixed amount and assume the value of A to be the one obtained last round. Then we compute q using a and A, with q we can find T. We then proceed to find the fix point A for this a. When A converges a new q is found hence a better value of T. This process can be iterated until T and A converges.

Figure 3 is a plot of the number of calls accepted vs. the offered load and both values are normalized. We assume a link capacity of 100 trunks and a holding time of 10 units on average. The load is varied by slowly increasing the arrival rate of calls. We can see that migration does indeed give a 2 to 3% increases in number of calls taken. The major performance gain occur at the region of load 0.9 to 1.1. This is understandable since it is only at this point when alternate routes are intensively used. Moreover, we find that the gain when only 2 trunks are reserved is higher than when 4 trunks are reserved. This is a very interesting observation since when there is no circuit migration, it is better to reserve more trunks at high load. We will explain this behavior when we discuss our simulation results in the next section.

It is possible to add the migration delay to the equation above to model its effect on the overall performance. Figure 4 depicts the effect of migration delay on the number of calls accepted. The migration delay injected is 0.01, 0.05 and 0.1 of the average holding time. When only two trunks are reserved the effect of delay on the performance is quite pronounced. However, as long as the delay is less than 0.01 of the holding time, the degradation in performance is minimal. When four trunks are reserved, the effect on the performance is minimal as long as the migration delay is less than 0.05 of the average holding time. Based on these results we felt that the delaying effect of the migration mechanism itself is small as long as the virtual circuit has a reasonable holding time. For example if the migration process takes 0.5 second then the circuits should have an average holding time of at least 50 seconds.

Although the analysis offered an estimation on the amount of improvements to expected. It does not give us any information about the tradeoff. In particular we do not know how many circuits must be migrated. In the next section, we will report our simulation results and discuss the tradeoff.

4 Simulation

All of the simulation results presented are done by keeping the average holding time constant and slowly increasing the arrival rate. Throughout the simulation, uniform traffic is assumed ,ie., the traffic pattern between every node pairs are the same. The simulation is divided into a number of periods with the average circuit holding time kept at 10 and the length of each period kept at 1000. The link capacity is 100 channels and the five nodes are completely connected. All the circuits are duplex and use the same path in both directions. Network status information is broadcasted to every node periodically. Migration involves rearranging a two hops connection to a single hop one. When a new call arrives the router selects a path based on the routing algorithm. The channels on the path are then reserved. A call is considered to be set up successfully only after this step. Since a router is just part of the switch, it is assumed to have the current status of the direct links. Therefore, once it has decided to choose a single hop route rejection should not come from the switching module. However if the path is two hops long then the router may not have up-to-date information of the second hop and hence rejection is still possible. For exactly the same reason, consideration for migration can occur as soon as the primary path has free capacity. When one of the channels of a fully utilized link is released, the source node of the link will search for an alternately routed calls that can be rerouted. If there is more than one candidate circuits, the router selects the one most recently set which is equivalent to LCFS in our previous analysis. The motivation here is to minimize the number of calls to be migrated. In the initial experiment, migration is triggered immediately when there is an alternately routed call and the primary path of the call is free. During the migration process, the new path is first set up by reserving channel from the primary path. Channels on the alternate path are then released. When a call finishes the channel reserved is released. Since our previous analysis showed that the duration of call set up, tear down and migration does not have significant effect on the performance as long as the circuits have a reasonable holding time, we will not model these values in our simulation.

Figure 5 shows the number of calls accepted vs. load at different trunk reservation factor.

With no trunk reserved the number of calls accepted decreases dramatically at load bigger than 0.9. It reflects that congestion is spread all over the network due to unrestricted alternate routing. When shortest path only routing is used, the throughput is lower at low and medium load but the problem with alternate routing at high load is eliminated. By reserving four trunks, we improve the throughput at medium load but at high load its performance is still subsumed by shorted path only routing. The above results match [Schwartz 87] very well and served as a verification of our simulations.

4.1 Trunk Reservation And Migration

Figure 6 compared the number of call accepted under trunk reservation and circuit migration with trunk reservation. At low to medium load, trunk reservations work very well and the ratio of call accepted and load is almost 1. When the load exceeds 0.9 the strength of circuit migration begins to show. Circuit migration releases the resource that was committed when alternate routing is used and therefore the network can use these resources to accept more calls. If there is no circuit migration, these resources would have been held until the call finishes. As we can see in Figure 6, without circuit migration, alternates routing becomes less and less beneficial as the load increases. Figure 7 depicts the call blocking probability resulted from using these schemes at various load. Since call blocking probability is inversely proportional to the number of calls accepted, we would consider the number of calls accepted as a performance indicator in the rest of the paper. Circuit migration allows us to use alternate routing at a minimal cost and hence improves the performance by 1.5 to 2 %. In addition to the improvement we found that the actual gain is dependent on the number of trunks reserved. The biggest improvement is obtained when no trunk is reserved, when four trunks are reserved the improvement is less. Under trunk reservation, a call is rejected by a link if it uses the link as part of an alternate route and that the utilization of the link is above a certain threshold. Although the link may have some unused capacity, it is reserved for calls that use it as the primary route. Under circuit migration, capacity devoted to an alternate route is only used in a temporary basis. The call is rerouted when capacity is available from its primary path. Hence there is less incentive to be conservative and turn down an alternately routed call when the link does indeed have the capacity to take the call. Thus by reserving less trunks, we allow more calls to be accepted knowing that they will release these resources very soon and would not block other calls from using it. This explains why the improvement is *higher* when *less* trunks are reserved. Interestingly, this was predicted by our analysis.

4.2 The Cost Of Improvement

Although circuit migration improves the throughput, calls are rerouted and as a result there will be more set up and tear down activities in the network. Rather than looking at the cost of migration perse, we would like to know how many circuits must be migrated in order to achieve a certain improvement. We measure the intensity of migration by the number of circuits migrated and normalize it with the average number of circuits that the network could have accommodated during that period. The latter value is dependent solely on the absolute capacity of the network and the holding time of calls. In our case since there are 10 links and each with a capacity of 100 trunks. Given an average holding time of 10 units, the average number of the calls that could be accomodated 100,000 in a time period of 1000 units. We use this as the normalizing factor since we feel that the call processing ability of the network should be a function of this magnitude. Thus the intensity of migration is an indication of how much call processing is needed. Figure 8 shows that the intensity of the migration activity decreases as more and more trunks are reserved. When no trunk is reserved the migration activity increases almost linearly with the load. On the other hand, once we introduce trunk reservation the amount of migration activities decreases. When more trunks are reserved, the number of calls taking alternate paths decrease and hence less migration occurs. Moreover when trunk reservation is introduced, the migration activities peaks at a load of 1 and decreases gradually. The reason is that as the load increases, the utilization of the trunk would become so high that it will frequently exceed the reservation threshold and hence only primary calls are accepted. From figure 8, we can see that the curve representing higher trunk reservation factor lower off earlier and faster.

Thus the tradeoff here is obvious, reserving less trunk improves the performance by being more flexible in resource allocation but requires more migration. On the other hand, reserving more trunks does not give as big an improvement but less migration has to be done. Unfortunately the cost of migration is still quite substantial even when four trunks are reserved, nearly 12% of the calls have to be migration. In fact if we look at the number of circuits that use alternate routes when migration is activated, we found that migration induces more alternate routing!

4.3 Threshing in Migration

The excessive amount of alternate routing led us to investigate in detail what was actually happening. Upon a closer look we find that when one performs migration, in addition to the load offered by primary calls (calls that uses the link as part of its primary route) and alternately routed calls, a link is also subjected to calls generated by the migration process. When one of the channel in a fully utilized link is released, migration takes place immediately and fills up the link again, very soon a new call will arrive and has to be routed on a longer path. However since it will not be long before another one of the 100 circuits in the primary path finishes, migration will take place and thus we enter a loop of migration and alternate routing. This phenomenon is shown in figure 9. Figure 10 depicts the migration profile at a load of 0.96, here the amount of time that a circuit spend on its alternate route before it is rerouted is investigated. For most of the circuits this duration is rather short hence further indicating the existence of threshing in migration. We propose two solutions to this problem. The first one restrains migration by allowing it to happen only when the primary path is relatively free, i.e when its utilization is below a certain threshold. The second solution migrates a circuit only if there is free capacity on the primary path and the circuit to be migrated is using at least one of the highly congested links. We call the first one **MIF** (Migrate If Free) and the second one **MIC** (Migrate If Congested). In the rest of this section we will look at the effectiveness of these schemes and also a hybrid approach where the two are combined. Unless specified, we reserved four trunks for primary calls.

4.3.1 MIF (Migrate If Free)

MIF is a direct solution that reduces the frequency of migration by doing so only when the primary link is relatively free, e.g, when there are three or four empty channels on the link. We can define the MIF threshold t_f such that if less than t_f channels are free no migration is done. When there are more than one alternately routed circuit that can be migrated, we can either select the one that is most recently set up (like what we did above) or the one using the most congested path by using a cost function similar to the one in dynamic routing. Our results indicate that both approaches give very similar performance. We choose to select the most recently set up circuit as the latter approach is subsumed by the hybrid approach described later. Since migration is not done as often the tradeoff here is a possible decrease in performance for less migration. Figure 11 shows the effect of MIF at various t_f. It can been seen that migration activities reduces as we become more cautious about migration. At $t_f = 4$ the amount of circuits migrated is reduced to a mere 3.6%. Figure 12 shows that there are some decreases in performance as we become more cautious about migration. However, the degradation is small and the tradeoff is well worthy. Figure 13 corresponds to figure 10 and shows the amount of time that a circuit spends on its alternate route before it is rerouted. By being more cautious , we have removed threshing and reduces the amount of circuits migrated.

4.3.2 MIC(Migrate If Congested)

The idea of MIC is to migrate only when a circuit is qualified. A circuit is qualified to be migrated if it is on a highly congested path. A path is congested if the utilization of any of its links is above a certain threshold. Let t_c be the threshold such that a link is said to be congested if the number of channels used is above t_c.

Unlike MIF where the process of migration could not happen until the utilization of the primary path is under a certain threshold, the MIF approach allows migration as long as there is free capacity in the target links. However, if none of the alternately routed circuits are qualified no migration will occur even if the primary path has free channels. If there are more than one qualified circuits, the tie is broken by using the same formula in dynamic routing and select the most congested one.

We simulated MIC for t_c = 98,99 , i.e, migration can occur only when at least one of the links is has 98 or 99 channels out of the 100 occupied. MIC seems to be more effective than delayed migration. As shown in figure 14, MIC reduces the amount of calls migrated by a rather substantial amount. Only a maximum of 2.5% of the calls have to be migrated in compare to delayed migration where 3.6% of the calls most be migrated. Figure 15 confirms that the effect on performance is minimal.

4.3.3 Hybrid Approach

It is possible to combine the two schemes together by not considering migration until the primary path occupation is under a certain threshold and even then migrate only if there is a qualified circuit. The hybrid scheme can be described by three parameters (t_f, t_c, t_r) where t_r is the number of trunks reserved. When t_c is set to 0 we would have MIF with most congested path first. Similarly, if t_f is set to 1 then we will have MIC. Figure 16 summarizes the degree of freedom that we have in circuit migration. In order to test the approach, we keep t_c at 99 and t_r at 4 and vary the MIF t_f across a spectrum of values. We hope that under the hybrid scheme, we would obtain the best tradeoff between the improvement in call acceptance and reduces the amount of circuits to be migrated. Figure 17 shows the number of call accepted. Figure 18 shows that the hybrid approach is rather effective , the amount of migration at load 1.00 is reduced to a mere 1.5% when $t_f = 4$.

In conclusion with the hybrid scheme we can reduce the amount of migration to a mere 1.5% with an increase in 1.3% of performance. Considering that the alternate paths are all twice as long as the primary path. Each migration could at most allow one more new call to be accepted. This ratio between performance and migration is very near the upper bound of 1 already. Further improvement is probably difficult. We had also done similar simulations when 2 trunks are reserved, in this case 4% of circuits have to be migrated for a performance gain of 2%. While the schemes experimented in here bring major improvement in reducing the migration intensity they are also very useful in resolving migration competition. Recall that migration competition occurs because when migration is done in a distributed fashion, a number of routers may observe changes in network status and jump into the chance with their alternately routed circuits. In a completely connected topology this does not happen since a link can only be on the shortest path of one node pair. By delaying migration until the utilization of the links in the shortest path drops, conflict can be reduced since the link may indeed have enough capacity to meet all these concurrent migration attempts. Finally, MIC helps to further reduce the number of contenders since a circuit is not qualified for migration unless its path is congested.

5 Issues In Circuit Migration

5.1 Setting Up And Tearing Down Connections

Call migration is by no means free. Migrating a circuit from one path to another involves partial call set up and tear down. The process must be done so that the interrupt to the service is minimal. This can be done by performing the set up on the new paths while packets are being transmitted on the old circuit. For switches that are on both paths no resource reservation is needed. If source routing is used then when the acknowledgement of successful call set up comes back, the source can switch to a new header and tear down the old path. If table routing is used a separate virtual circuit number should be used in each hop. Otherwise if the new path overlap the old path at some point, the packets may start to follow the new path even when it has not been fully established. As in the case of source routing, no extra resources should be reserved from the links that are used in both paths. When the source router receives the set up acknowledgement it can switch to the new path and tear down the old path. During the tearing down process, routing tables of the switches along the old path must be cleaned up. But we cannot reclaim the resources on the links that are used in both paths. As we can see, the overhead of the migration process is comparable to a connection set up and tear down. It is possible that the resource to be used for rerouting has been grabbed by the others before the set up message arrives, in this case the rerouting attempt failed and the circuit remains on the old path.

5.2 Packet Reordering

It is possible for packet reordering to occur during circuit migration. Reordering could occur because the new path has a smaller delay than the old path and hence packets traveling on the new path may arrive at the destination before some of the packets traveling on the old path. In ATM networks, cell reordering within a packet can be avoided by switching to the new path only when a new packet is to be transmitted. The sequencing of packets may be combined with the selective repeat where packets arriving out of order are buffered while waiting for missing packets. In applications where discarded packets are not retransmitted, then there would not be any selective repeat buffering at the receiving side and packet resequencing may have to be introduced. It is possible to use existing resequencing algorithms if the packets are sequence numbered. In ATM environment where the cell header does not include sequence number, some more information may be needed.

In principle all the packets that arrives at the destination from the new path before the last packet on the old path must be buffered. Thus if we can distinguish packets on the new path from packets from the old path, then we can start buffering the packets on the new path as soon as they come in. We would release the packets in the buffer only after the last packets on the old path has arrived. It is possible to separate packets on the new path from the old one since they may have different virtual circuit identifier. Another way is to set a certain bit for the packets going to the new path. This would continue for a while until the receiving end has released the buffer. We can devote another bit on the header so that it would be set if the packet is the last packet on the old path. Such a packet act as the release buffer message. Since it is possible for this packet to be discarded, the buffer of packets coming from the new path may have to be held for longer than necessary and overflow may occur. This situation can be avoided by releasing the buffer after a certain time period. This period starts when the first packet from the new path is received and its length can be set to be slightly bigger than the delay difference between the old and the new path. Once the buffer is released, subsequent packets coming from the old path are discarded. The amount of buffer needed is closely related how much faster is the new path in compare to the old path and the peak rate of the connection. [Tam 91] investigates in detail the amount of buffer needed when we only have an estimate on the total delay on both paths. As an example, consider a 9 Mbps circuit with constant bit rate and the new path is faster than the old path by 10ms. Then the amount of buffer needed is approximately 11.25 Kb. If the router has a total buffer of 2Mb, then 177 circuits can be migrated in parallel.

There are a number of advantages to perform resequencing at the router level. Firstly circuit migration should be transparent to the applications. The end user should not have to modify the transport layer or above. Secondly, since reordering would only occur temporarily the resequencing buffer can be shared and putting it in the router allow it to be shared. Moreover, if there is any additional hardware that could help the resequencing it could be shared in this way too. Since buffering is necessary, buffer at the routers must be reserved like other resources before rerouting can occur.

In fact, after sending the last packet onto the old path the packets, the source can delay sending out packets onto the new path for a short period of time by buffering them. This would reduce the amount of buffer needed at the receiving side and hence allow more circuits going into a destination router to be migrated in parallel. By sharing the buffering requirement between the source and the destination, rerouting are less likely to fail because buffers are unavailable.

6 Conclusions

The work presented in this paper is our first step towards the area of circuit migration. Routing on alternate paths consumes extra resources and usually these resources are kept until the circuit is released. Migration allows these resources to be released much earlier by rearranging the virtual circuit to use the shortest path when it becomes available. Since resources used as part of an alternate route would be released shortly by rerouting, there is less incentive to be too cautious in reserving resources for forthcoming calls. This phenomenon is shown in our analysis and simulations where we concluded that rerouting performs better when less trunks are reserved. The analysis in this paper is intuitively based. Nevertheless, the approximation was accurate in reflecting the relationship of trunk reservation and call migration. While migration improves the throughput at very high load by 2%, a large number of circuits must be migrated in a simple minded scheme, e.g, migrate whenever the shortest path has the required resource. Our simulation indicates that the number of circuits migrated is also inversely proportional to the number of trunks reserved. Thus together with the above phenomenon forms an interesting tradeoff. Further investigation revealed that threshing can also occur in migration. Two solutions are proposed, namely migrate only when the primary path utilization is below a certain threshold or migrate only when the alternately routed circuits are causing congestions. While both of these schemes traded performance for less migration, the latter method seems to be more effective than the first one. The hybrid scheme provides further improvement and acheives a near upper bound tradeoff.

Migrating a circuit includes setting up and tearing down a circuit. During the migration process, packets from the new path could arrive earlier than those on the old path causing packets to be misordered. By exploring the difference in the delay of the two paths, it is possible to reduce the size of the reordering buffer at the other end substantially. Hence we believe provided that buffering is done appropriately, interruption of services during the migration process is minimal.

In this paper we have only considered completely connected network since it allows us to eliminate the problem of migration conflict. Using arbitrary topology opens the door to some more possibilities, for example, we can give priorities to circuits that is using very long alternate paths or to those where the primary path is short. Moreover, since alternate routes of different lengths is possible, we can migrate a circuit from a long alternate path to a shorter alternate path and finally to the primary path.

Further research on circuit migration should be directed toward using it as an integrated scheme to improve network utilization and at the same time performs circuits recoveries in case of link or node failures.

References

[Schwartz 87] M. Schwartz. "Telecommunication Networks, Protocols, Modeling and Analysis". *Addison Wesley 1987.*

[Decina 90] M. Decina, T. Toniatti, P. Vaccari, L. Verri. "Bandwidth Assignment And Virtual Call Blocking In ATM Networks" *IEEE Globecom 1990, page 844 - 851*

[Girard 83] A. Girard, S. Hurtubise. "Dynamic Routing and Call Migration Circuit-Switched Networks". *IEEE.Transactions on Communications VOL. COM.-31 NO. 12,* December 1983.

[Ash 81] G. R. Ash, R. H. Cardwell, R.P. Murray. "Design and Optimization of Networks with Dynamic Routing,". *BSTJ*, vol. 60, no. 8 OCT. 1981, 1787-1820.

[Bertsekets 87] Bertsekets D., Gallager R. "Data Networks", *Prentice Hall 1987, Englwood Cliffs, NJ 07632*

[Krupp 82] R. S.Krupp. "Stabilization of Alternate Routing Networks", *ICC '82, Philadelphia, June 1982.*

[Humblet 86] Pierre A. Humblet, Stuart R. Soloway, "Algorithms For Data Communication Networks - Part 2" CODEX Corporation.

[Tam 91] Ming-Chit Tam, David Farber. "On Call Repacking" DSL Technical Report, University of Pennsylvania 1991.

[Hwang 9] Ren-Hung Hwang, Jam F. Kurose. "On Virtual Circuit Routing and Re-Routing in Packet -Switched Networks". *IEEE. ICC' 91, June 23, Session 41.4*

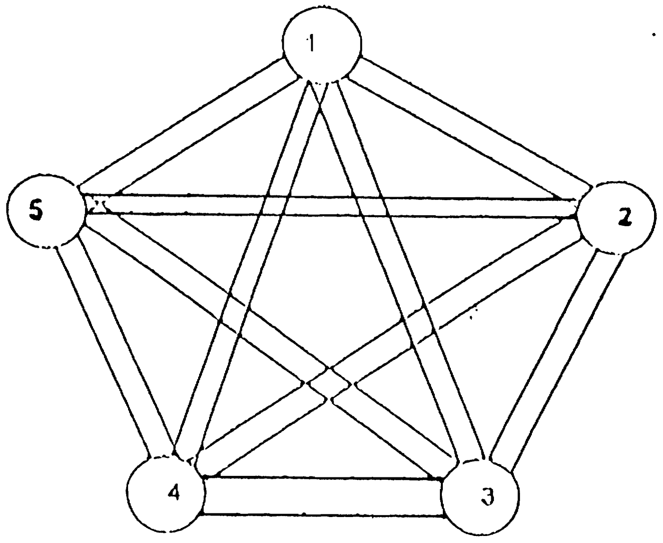

Figure 1: Our Model - A Completely Connected Network

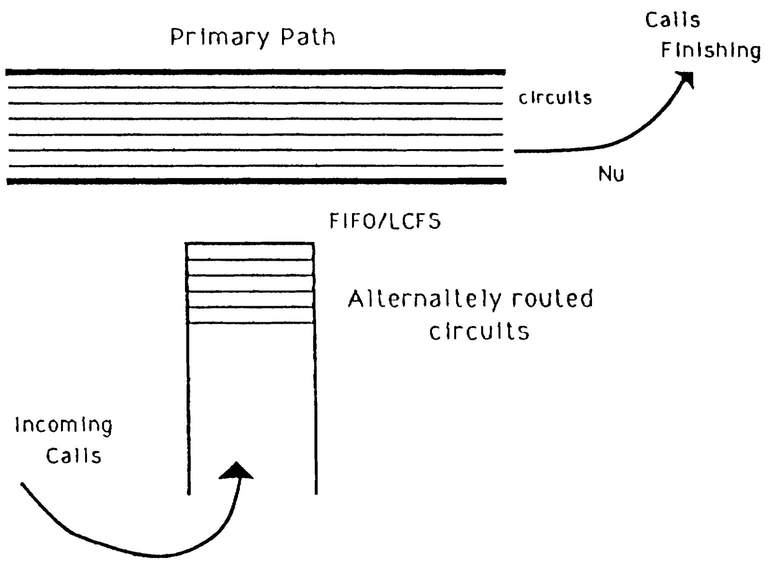

Figure 2: M/M/1/N queue modeling the migration process

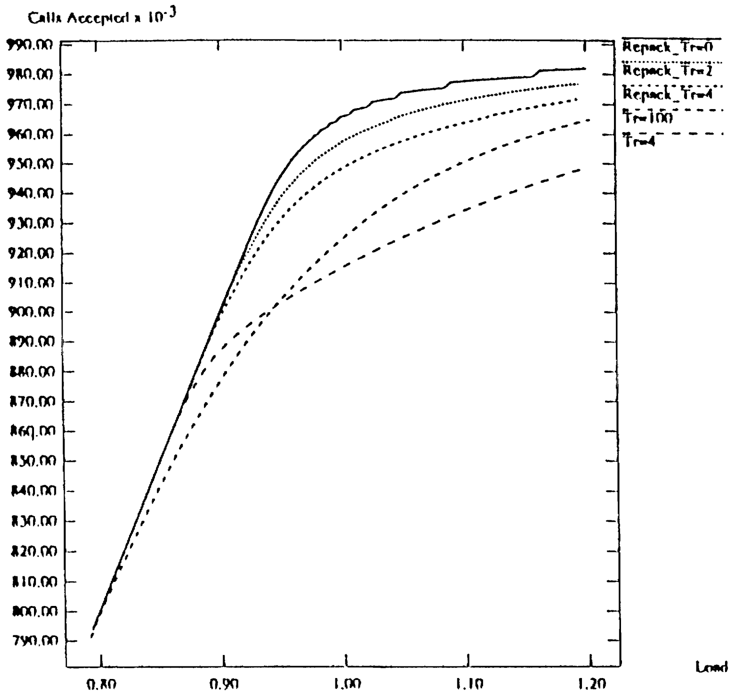

Figure 3: Number of calls accepted vs. load

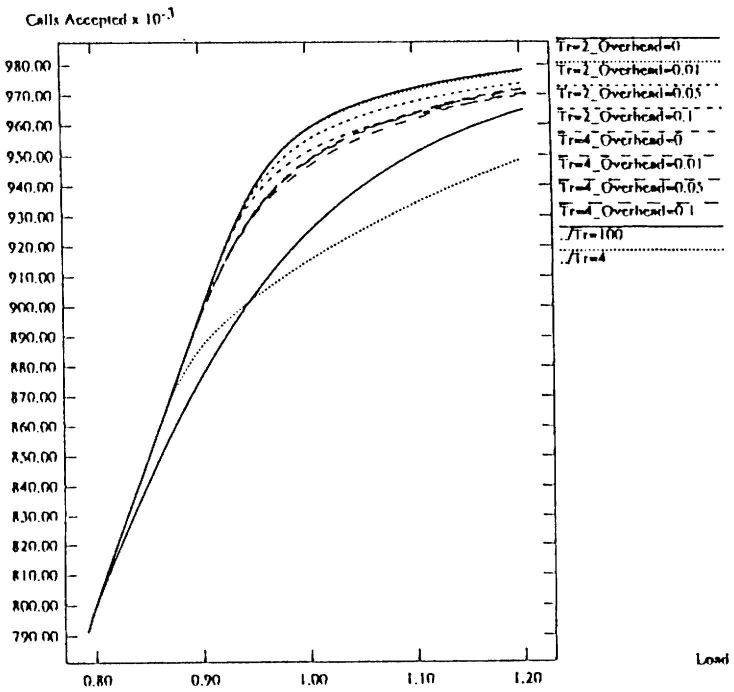

Figure 4: Effect of migration delay on performance

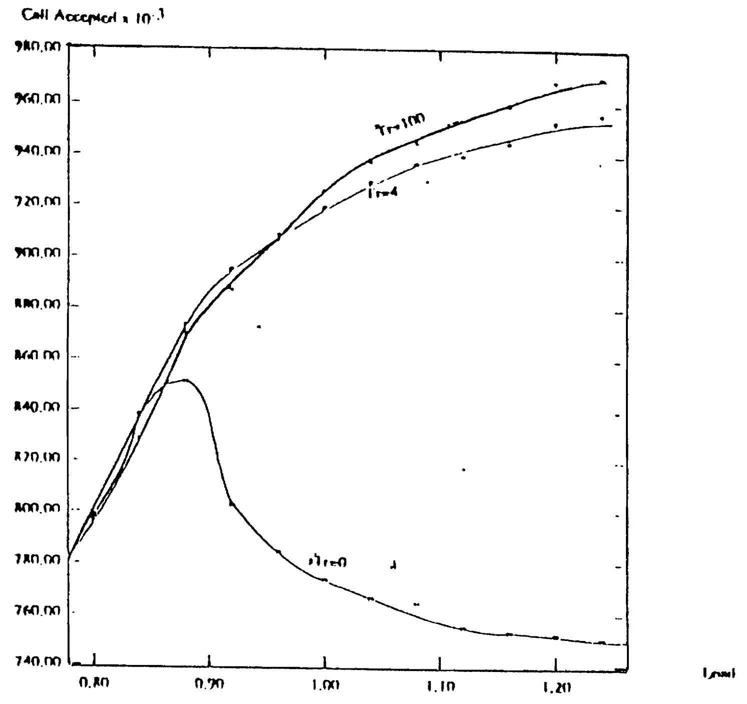

Figure 5: Number of calls accepted vs. load

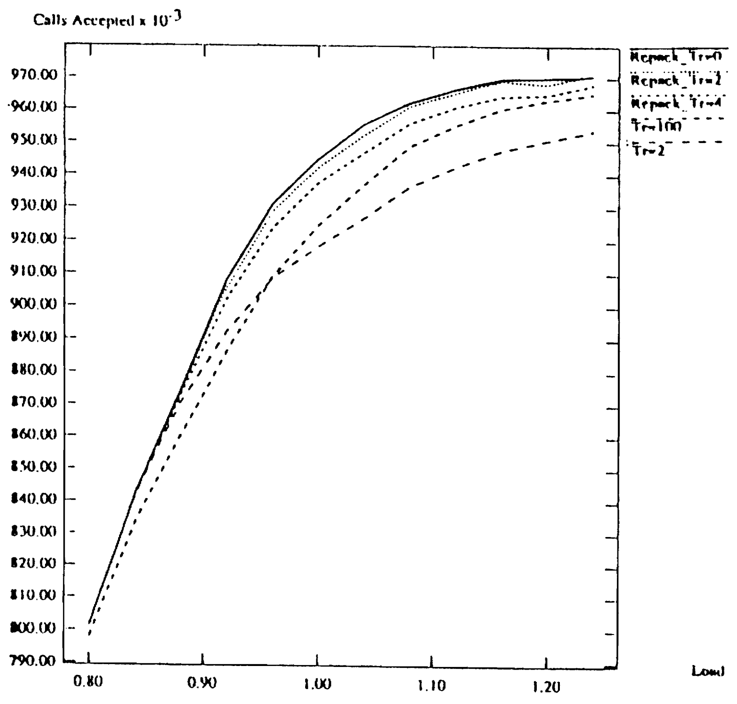

Figure 6: Number of calls accepted vs. load

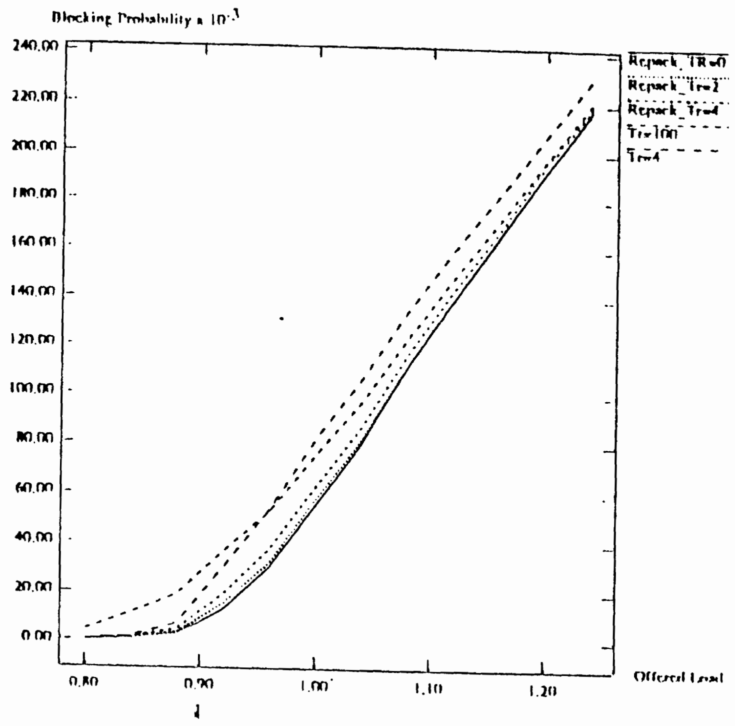

Figure 7: Call blocking probability

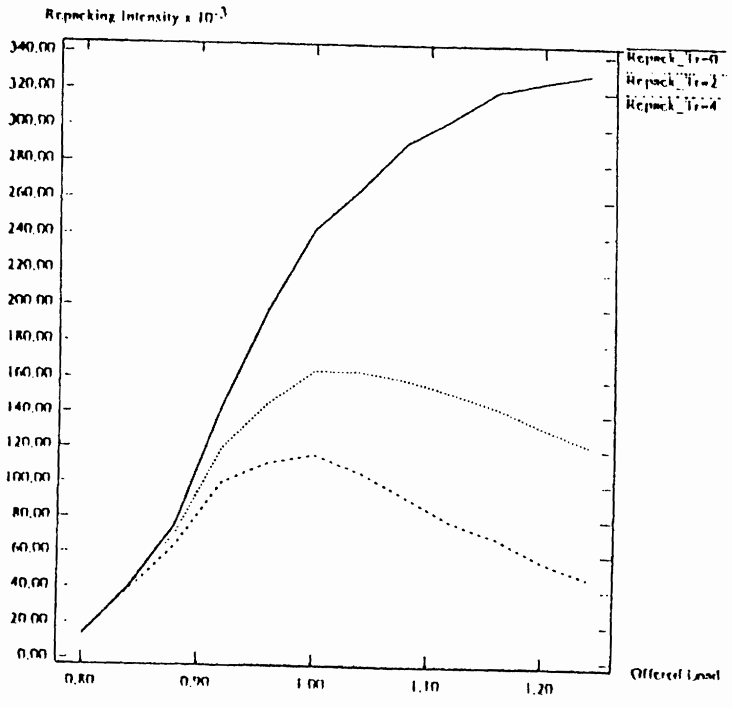

Figure 8: Migration Intensity

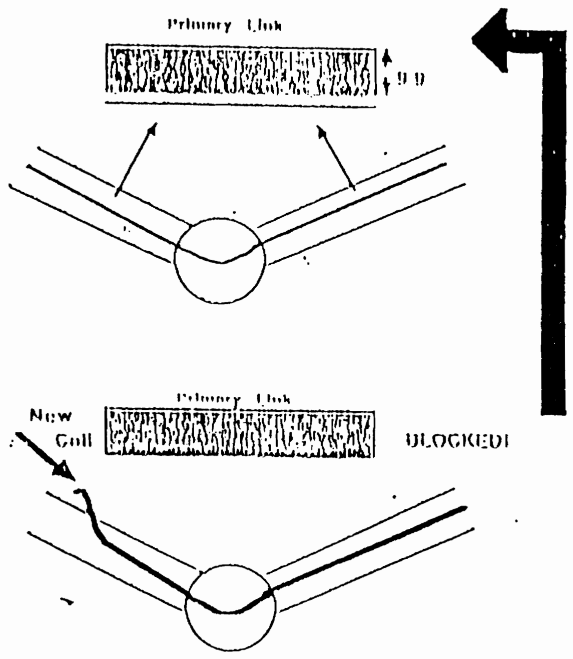

Figure 9: The phenomenon of threshing in circuit migration

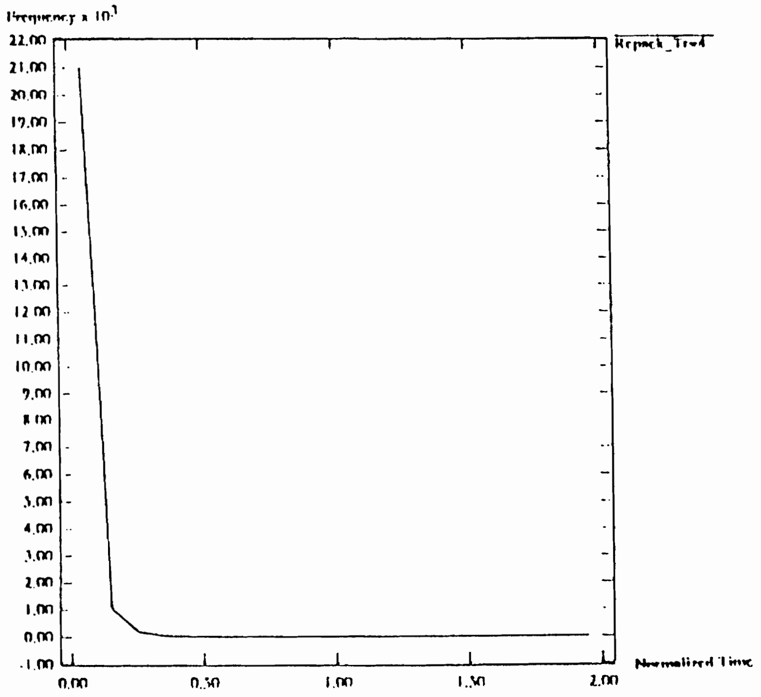

Figure 10: Circuit migration profile

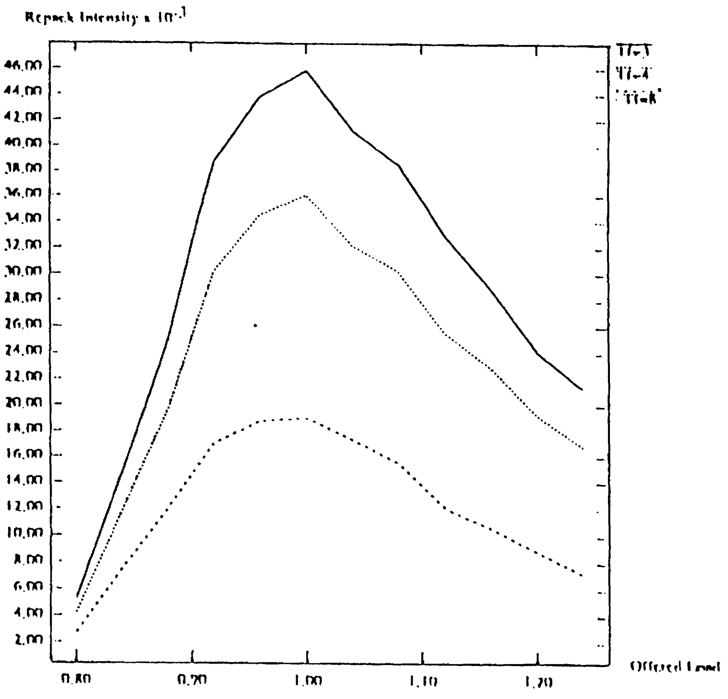

Figure 11: Migration Intensity at various T_f

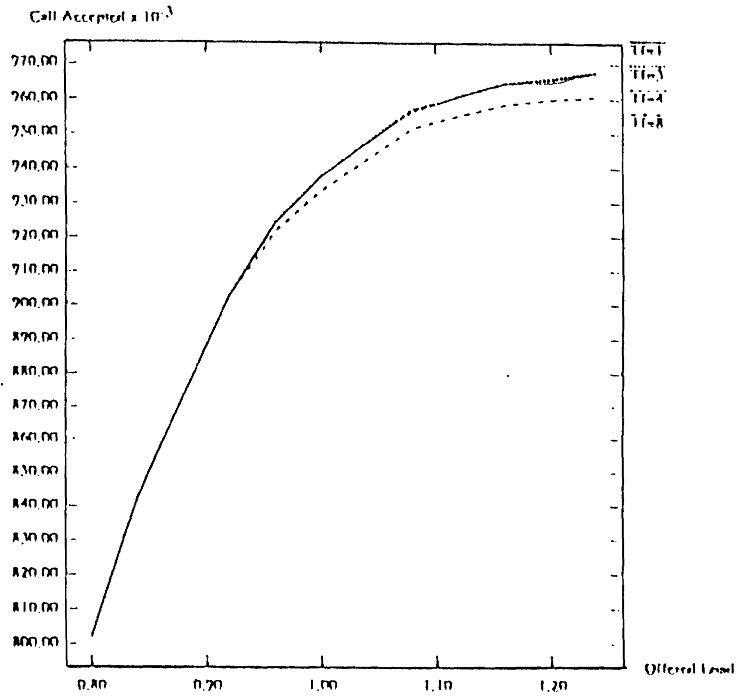

Figure 12: Small degration of performance with MIF

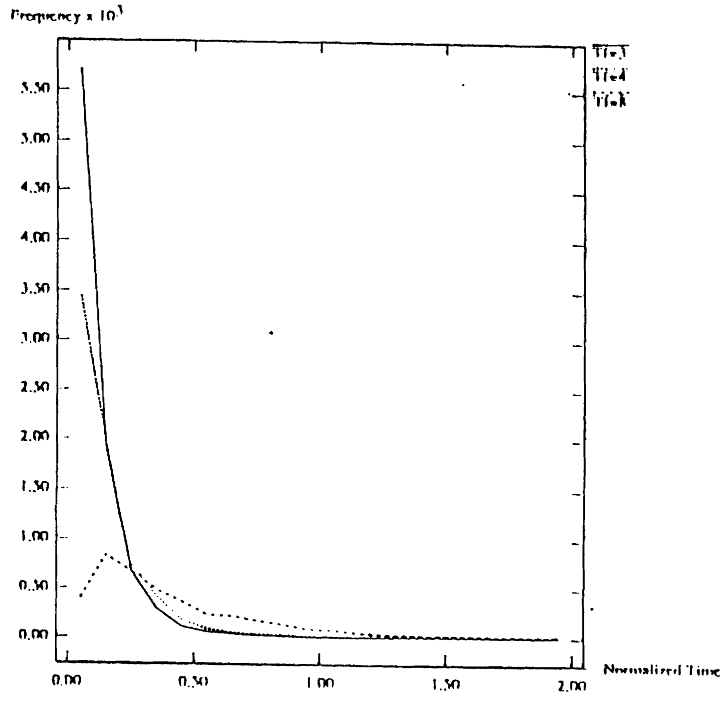

Figure 13: Migration profile of MIF

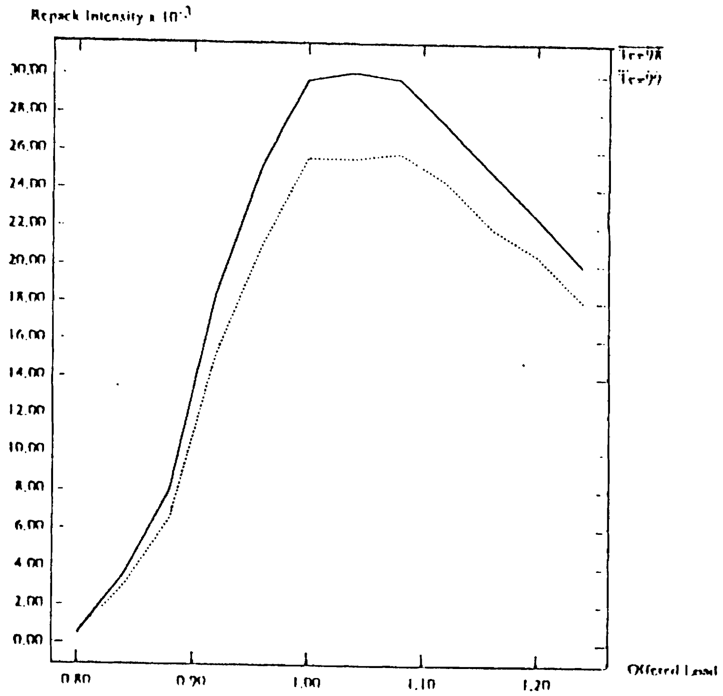

Figure 14: Migration Intensity at various T_c

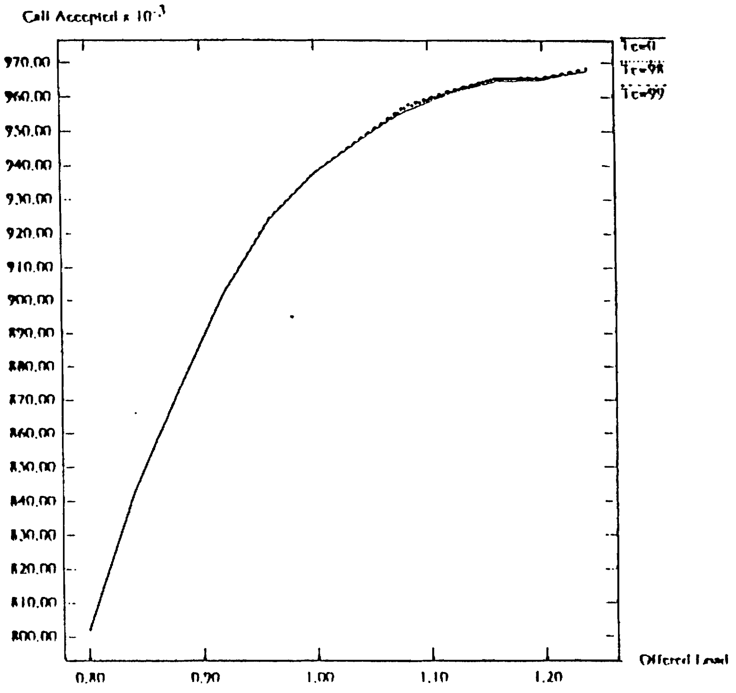

Figure 15: Small degration of performance with MIC

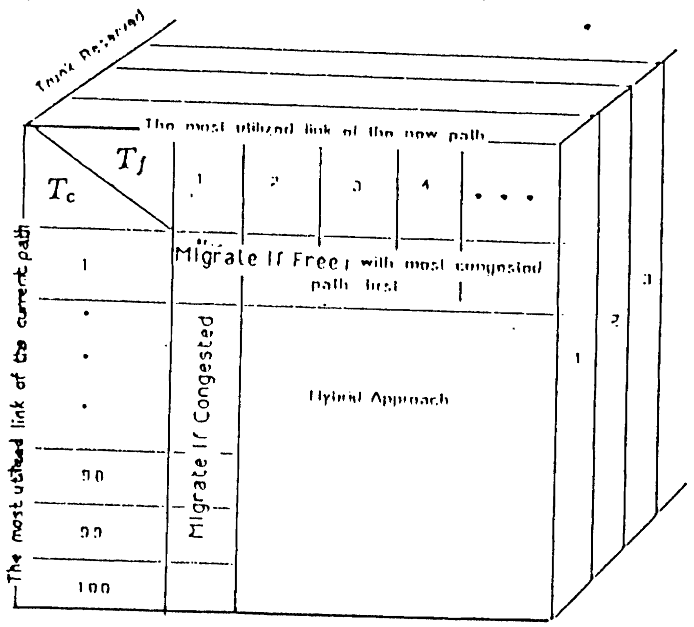
Trunk Reserved
The most utilized link of the new path
T_f
T_c
1
2
3
4
. . .
The most utilized link of the current path
1
.
.
.
98
99
100
Migrate If Free, with most congested path first
Migrate If Congested
Hybrid Approach
1
2
3

Figure 16: The three parameters investigated

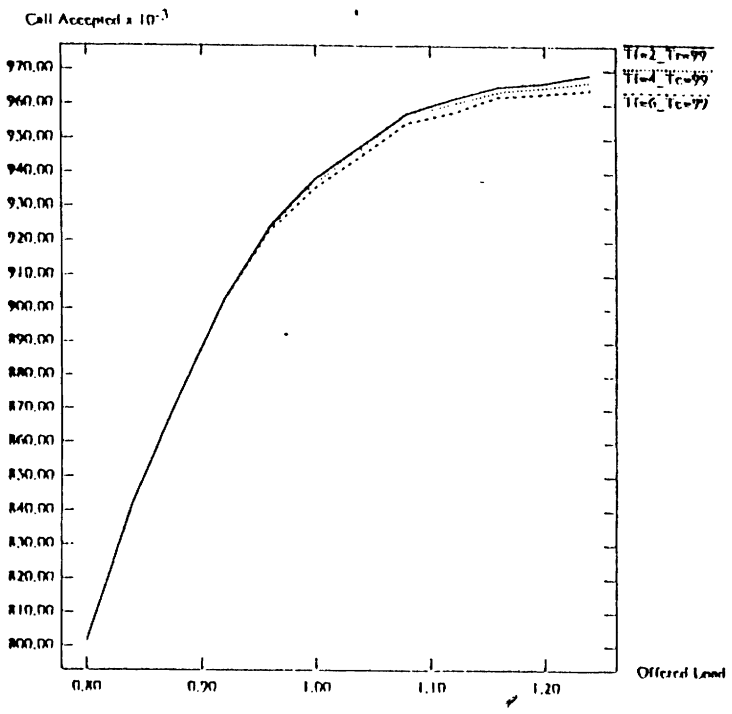

Figure 17: Small degradation with hybrid scheme

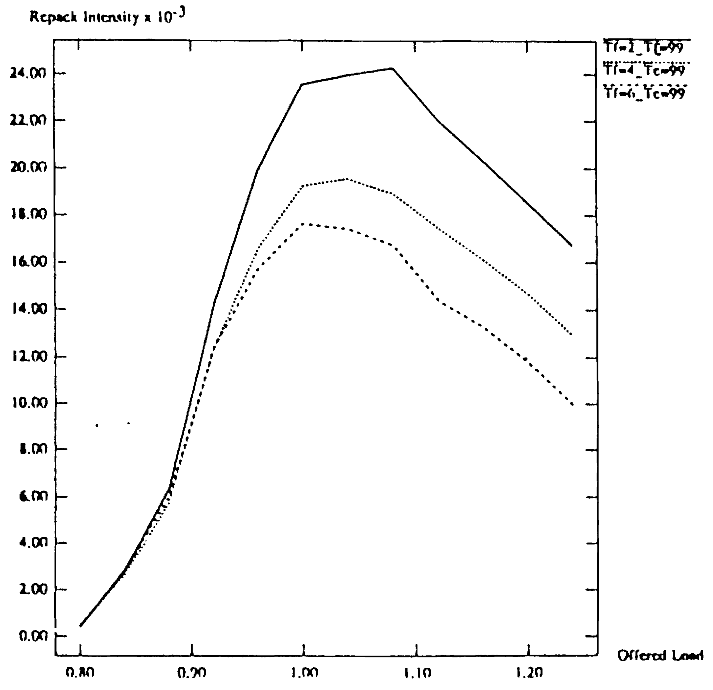

Figure 18: Migration Intensity of hybrid scheme