LABORATORY FOR
COMPUTER SCIENCE

MASSACHUSETTS
INSTITUTE OF
TECHNOLOGY

Hot Spots in Routing Networks

A Collection of Studies

Computation Structures Group Memo #267
1 October 1986

Andrew A. Chien

This report describes research done at the Laboratory for Computer Science of the Massachusetts Institute of Technology. Funding for this project is provided in part by the Advanced Research Projects Agency of the Department of Defense under the Office of Naval Research contract N00014-75-C-0661.

545 TECHNOLOGY SQUARE, CAMBRIDGE, MASSACHUSETTS 02139

Foreword

This document was produced in order to document peripheral research results concerning hot spots in routing networks. It is intended to present supplementary information for Andrew A. Chien's Master's Thesis – *Congestion Control in Routing Networks.* We encourage interested readers to first read *Congestion Control in Routing Networks* before perusing this document.

Contents

Chapter 1

The Effectiveness of Combining – an experimental study

1.1 Idealized Combining

The term "combining" refers to a local network operation that takes two or more network packets and combines them to produce a single output packet. Local state is maintained so that when a response is received, appropriate responses for all of the original input packets can be constructed and transmitted. This operation was originally proposed as means of approximating the PRAM model for parallel computation. The "combining" effect could conceivably allow all processors in a system to access the same memory location simultaneously. This type of computer was called "Ultracomputers".

To allow combining, systems such as RP3 relax the semantics of a series of requests to a single memory location. As the processors are operating asynchronously, their requests to a single location are interleaved in some arbitrary way. In fact, in the RP3 system, the requests from a single processor may become ordered arbitrarily. However, idealized combining has several benefits. It allows many processors to access a single location in the memory "simultaneously". This effectively increases the bandwidth of the memory. It may be effective in dissipating some forms of congestion (see Pfister and Norton).

1.2 Pragmatic Concerns

Combining has the potential benefit of reducing the amount of traffic to a popular memory location. If single memory locations in conjunction with "Fetch-and-Add" style operations for system synchronization, then this benefit will be substantial. In fact, many PRAM algorithms do use this sort of operation for rapid communication and synchronization.

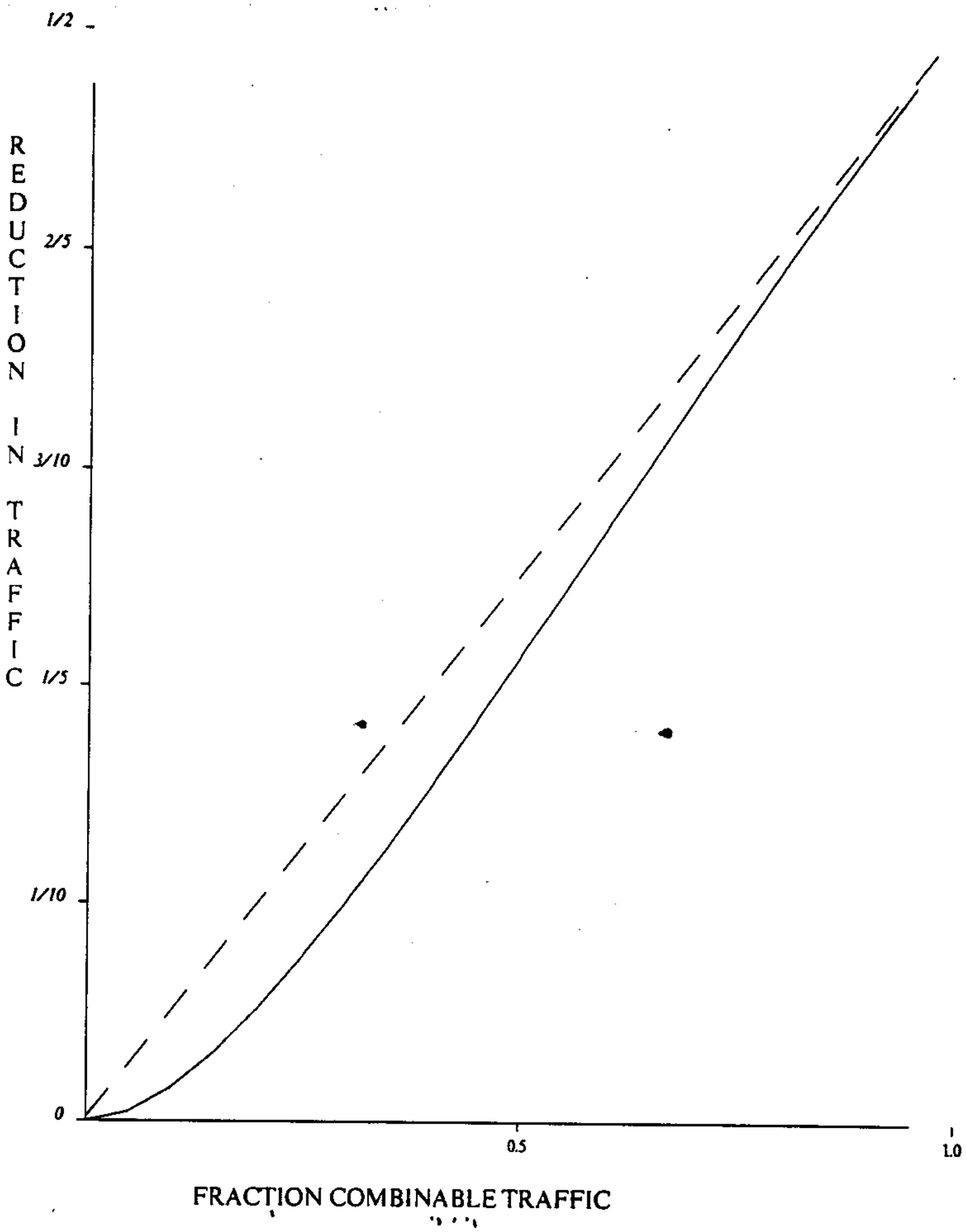
Unfortunately, the benefits of combining do not come without cost. In practice, idealized combining is very difficult to implement. The current proposal for the RP3 combining network allows only binary combining at each stage. This means that each packet may only combine once at each stage. In addition, the combining network is designed to carry only the synchronization references (they must be programmer or compiler specified). This means that the combining properties of the network will be enhanced – the combinable traffic will not be diluted by background data traffic. Also, the combining network can then be designed for a much lower speed (higher delay). The logic required for even binary combining necessitated implementation in a much denser, slower technology. The observation made by Pfister and Norton was that combining increased the hardware cost of the network by a factor between 6 and 30.

Given the large hardware overhead required for combining, one wonders whether that money could be better spent on other parts of the system or even other components of the network. For example, one might consider the replication of links (proposed by Kumar and Jump) as a means of enhancing network performance at a lower cost. In order to evaluate the effectiveness of the "realistic" form of combining that is practical, though expensive, we undertook a study of the rate of combining in a network switch. This study is described in the next section.

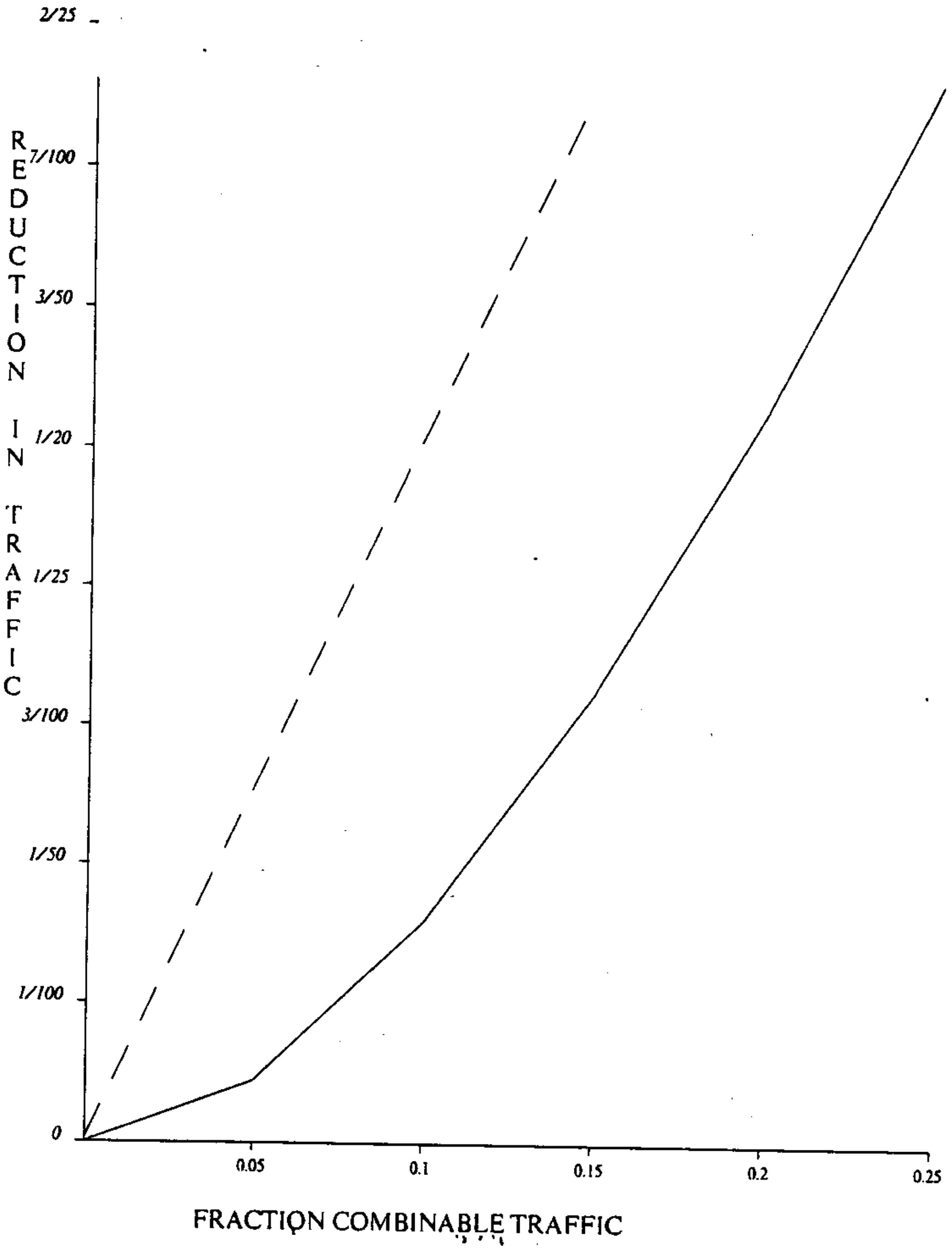
1.3 Results

We considered combining networks made from 2-by-2 routers. As in the RP3 combining network, we allow only binary combining. In this study, we simulated two different queue lengths, three packets and five packets. We assumed that all of the switch buffers were full at all times. During actual network operation, the buffers will not be this full. Thus, the results presented are somewhat optimistic for what would occur in an actual system.

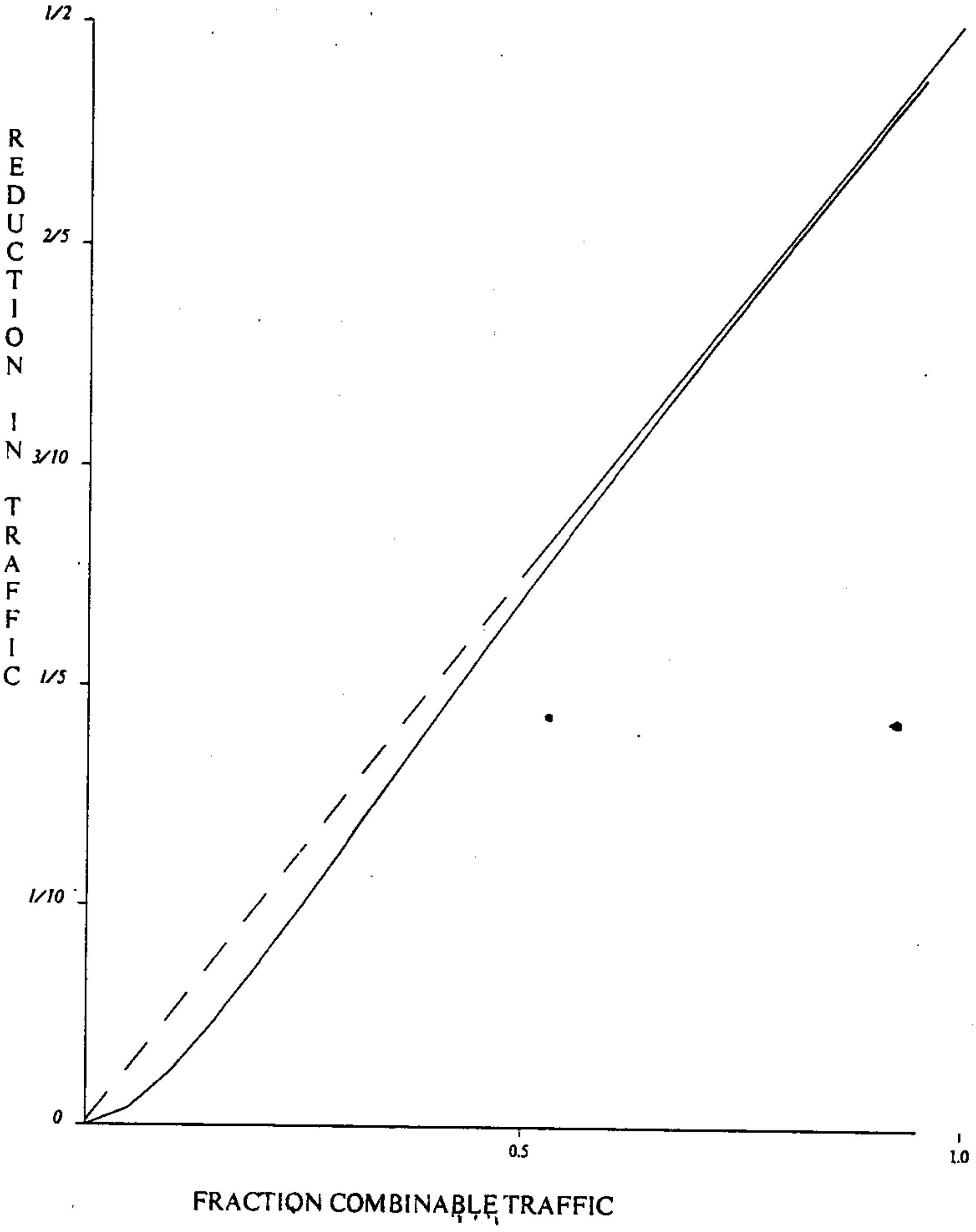
The results are attached. They show that for low fractions of combinable traffic, the binary combining behaves significantly worse than idealized combining. For the queue size of three packets, below 25% combinable traffic the effective reduction in traffic was less than half what would be possible with idealized combining. For lower percentages of combinable traffic, the relative performance of binary combining is even worse. For queue lengths of five, the results were much better. However, by the time that five packets of buffering have filled in several stages of a routing network, the delay performance of the network is severely degraded. These results lead to serious doubt concerning the effectiveness of binary combining schemes in situations where the combinable traffic is only a moderate fraction of the overall traffic. It appears that a very high percentage of the overall traffic must be combinable before binary combining will significantly reduce the amount of traffic.



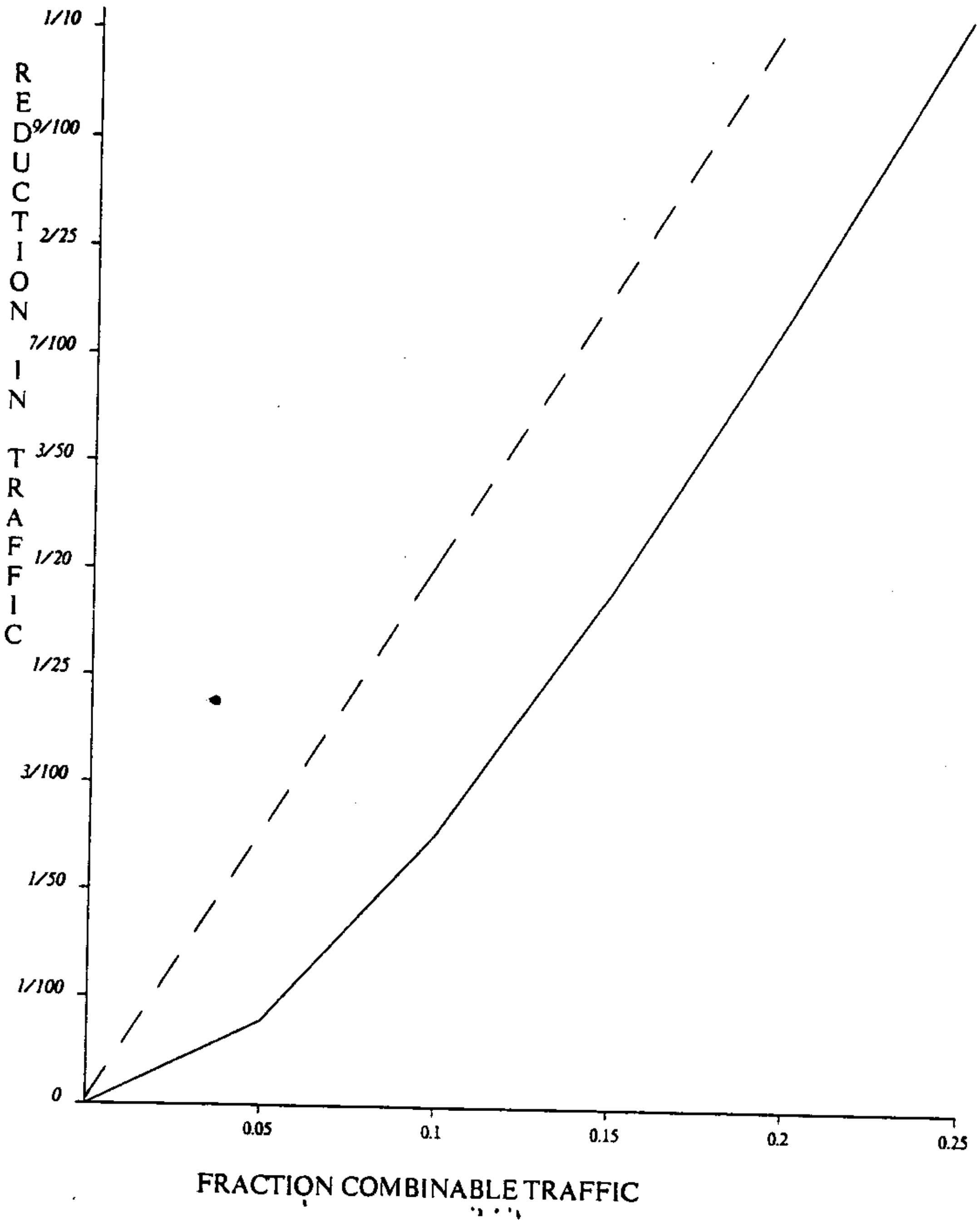
Queue Length = 3 Packets



Queue Length = 3 Packets



Queue Length = 5 Packets



Queue Length = 5 packets

Chapter 2

A Comparison of Hot Spot Detection Mechanisms

In this chapter we discuss different approaches to the detection of traffic imbalances in routing networks. Both centralized and distributed schemes are discussed. However, the discussion centers on distributed schemes because communication costs are quite significant in high speed routing networks. Two schemes are selected and simulated. Their relative performance in terms of sensitivity and robustness is presented and analyzed.

2.1 Hot Spot Detection Mechanisms

The basic purpose of a hot spot detection mechanism is to detect traffic imbalances, specifically hot spots. It is preferable to detect these hot spots as early as possible. This facilitates the effectiveness of the network control mechanism. It is also desirable for the mechanism to produce relatively few "false alarms". The importance of this requirement depends strongly on the control mechanism implemented. Less severe mechanisms will be able to tolerate more false alarms. Conversely, those mechanisms taking severe action or requiring large amounts of communication will be able to tolerate only a few false alarms. The cost of responding to more than a few such alarms may mitigate the benefits of the control system.

2.1.1 Centralized Hot Spot Detection

There are many possible centralized schemes for detection of traffic imbalances. The basic idea is that information concerning the network state is communicated to a central controller. This controller then digests the information and comes to conclusions about the condition of the network. Such schemes can communicate two basic kinds of information. They can either communicate information about the actual traffic, source and destination distribution, or they can communicate the value of some network performance parameter. Below, we discuss these two approaches and their primary drawbacks.

Traffic Distribution

In a scheme that communicates the traffic distribution, there are several possibilities. The source nodes can collect statistics for recent history and communicate that information to the central controller. The switches embedded in the network can keep statistics and communicate them to the central controller. Or, the sink nodes can collect information about the packets they receive (this may require the addition of some sort of return addresses) and send that information to the central controller. Various combinations of these information collection schemes are also possible.

At the central controller, the information is tabulated and processed to produce a profile of the loads in the system. Then on the basis of this information, decisions to declare hot spots or other congestion can be made. This scheme is attractive because it seems capable of detecting hot spots that arise from very small source imbalances. The cumulative effect of these imbalances is to cause a very extreme hot spot. A central controller, would have global knowledge of the network traffic load and hence be able to detect such a hot spot. In fact, the communication of traffic to a central controller has the potential to be the most sensitive and robust scheme. If the controller has all the information, it can detect all imbalances as soon as the distributions from all the sources are received.

The primary criticism of this scheme is that it requires excessive amounts of communication. If each source is to periodically transmit its traffic distribution to a central controller, this communication traffic is very large. Furthermore, if each source must, for example, specify the amount of traffic it sent to each of the destinations, the amount of information that we must communicate is growing at rate $O(n^2)$. This is a prohibitive amount of communication. This communication requirement is growing at a rate faster than the bandwidth of the network. Of course, we need not transmit the traffic distribution to the central controller very often, but the less often we transmit this information, the more sluggish the system response will become. This is because longer transmission cycles to the global controller will defer the time at which it can determine that a traffic imbalance exists. The primary benefit of this scheme is its potentially very high sensitivity and accuracy. This will have to be traded away in order to keep the communication requirements within reason. Thus, this sort of approach is impractical.

Network Performance Parameter

In this scheme, the elements of the network communicate the value of some network performance parameter to a global controller. This controller then interprets the information and reaches some conclusions about the network state. Some common choices for parameters might include average delay, average switch throughput, or link utilization. In practice, one can only maintain a small number of these without unreasonably increasing the amount of overhead. One could even conceive a scenario in which many parameters were measured and the information needed by the global controller was explicitly requested when it was applicable. Again, this genre of systems has the potential to be very sensitive to traffic imbalances. In the presence of very mild traffic imbalances, a cleverly designed network controller might be able to "focus" on the area experiencing an imbalance and tune the traffic through that region very precisely.

This type of system has several significant drawbacks. Although the communication requirement is less than in the centralized traffic distribution scheme, the presence of a global controller means that we will still need O(n) traffic. In order for the controller to make accurate and timely decisions, it will need updates fairly often. This will require the presence of another network, or significant bandwidth from the network being controlled. In addition, this system exhibits the property that as the system becomes congested, the increased activity of the global controller (to dissipate the congestion) is likely to further aggravate the congestion problem.

Summary

In summary, schemes employing global controllers for detection of traffic imbalances have the potential to be very sensitive and robust detectors of traffic imbalances. However, because of their excessive communication requirements, it is doubtful that they will be a practical solution to the congestion problem. Further, the idea of using a centralized controller is perhaps inconsistent with the main motivation for using multi-stage routing networks. In such networks, modularity, simple scalability, and fixed size building blocks were emphasized as aids to building a fast and reliable system. Using a global controller for detection seems to negate many of these advantages.

2.1.2 Distributed Detection Mechanisms

There are various approaches to distributed detection of traffic imbalances. However, the underlying premise to all of these schemes is that communication control information is expensive. Hence, such communication should be minimized. To this end, network state information is collected locally – at the sinks, sources, or switches – and decisions concerning the seriousness of traffic imbalances are also made locally. This allows for a very low overhead, modular, scalable detection mechanism. However, due to the limited amount of communication and the lack of a central network state, the sensitivity and robustness of this type of detection scheme is likely to be lower than that in centralized schemes.

Source or Sink Based Mechanisms

These mechanisms measure the balance of traffic at either the sources or the sinks of the network. This measurement is relatively cheap because it is a local operation. However, at the network size grows, if a source or sink is to maintain a comprehensive distribution of recently processed traffic, the amount of state required will grow proportionally to the size of the network. This is discouraging, but the approach still seems feasible.

The benefit of measuring traffic imbalances at the sources to the network is of questionable utility. If multiprocessors, like uniprocessors exhibit locality of access, it seems that the traffic distribution from a single processor will often be extremely skewed. If such large variances in the traffic distribution are typical, then it will be difficult to construct a robust system based on the local measurement of traffic imbalance. Any scheme is likely to produce a relatively large number of false alarms. Furthermore, it is a characteristic of the congestion problem in which we are primarily interested that the traffic at each source is quite balanced. Any scheme based solely on the source traffic distribution would probably not detect many of these hot spots.

Measuring the traffic rate and mix at the network sinks is a promising approach to detection of traffic imbalances. Clearly, in the case of hot spots, a sink based distributed detection scheme would be able to detect a hot spot by simply measuring the load at the sink. However, detection based solely on sink statistics does not give us a great deal of information about the internal state of the network. For example, contention for an internal network link might never be detected by this class of schemes. This approach was deemed promising, but not quite as attractive as the switch based detection approach. Thus, we have pursued two switch based techniques described below.

Switch Based Detection

Switch based detection schemes measure some network parameters at each of the switches. Information collected in this way can tell us a great deal about the network. As the network size increases, so does the amount of information we are collecting. This means that have a modular, in some sense scalable data collection scheme. Unfortunately, we are still operating under the assumption that communication is expensive and must be minimized. This means that the collected information must be processed by each switch. Switch based schemes are more attractive than source or sink based schemes because switch base schemes can approximate these other schemes (by using the switches near the network inputs and outputs). In addition, switch based schemes are able to detect internal network state – something source and sink based schemes cannot easily do.

Two switch based detection schemes were developed. They are based on slightly different premises and are described below. Each schemes theoretical advantages and disadvantages are discussed. Finally, simulation results for both schemes are presented. These results show that the traffic imbalance scheme performs significantly better under the random traffic loads used for simulation.

2.2 Two Switch Based Distributed Detection Mechanisms

We consider two seperate triggering techniques for traffic imbalance detection. The first technique is based on an exponentially decaying average of link utilization. For this technique, a switch keeps a decaying average for each of it's output links. When the switch sends a packet on that link, the average is updated by adding a constant factor and multiplying by the decay constant. When a switch does not send a packet on the link, it updates the state by simply multiplying the average by the delay constant. When the average exceeds some threshold, we declare that the link to be in a state of overload. This is equivalent to detecting a hot spot for that link. We will refer to this detection scheme as the Link Utilization or simply Utilization scheme.

The second technique is based on a key observation about hot spot congestion. When a hot spot occurs, not only does the traffic on the links to the hot spot increase in intensity, there also is a commensurate decrease in traffic on the links not leading to the hot spot.

Thus, at the boundary of the saturating tree, an imbalance in the amount of traffic being routed onto various links is clearly detectable. This imbalance is larger for the stages near the hot spot (as the routing function of the network "concentrates" the hot spot traffic).

Thus, by keeping track of the proportion of packets we have routed in every direction over a window of time, we can detect such imbalances. We consider the implementation for 2-by-2 routers. Implementation for larger switches is possible, but the 2-by-2 case is much simpler. This technique is implemented by having each switch remember where it routed the last **m** packets. When more than a certain percentage of these, have gone to the same output link, the warning is triggered. Some compensation has been added to account for low traffic situations. In early simulation runs it was discovered that traffic levels which posed no threat of causing hot spots were causing switches to go into warning. So to address this problem, when a switch is idle for a cycle, it removes the oldest value from its memory and inserts a nonsense value (note that it only inserts one, not two) into its memory. This seems to effectively prevent the false triggers at low traffic levels. In practice, experimental results have shown that values of **m** around 25 to 30 are quite effective. We implement this scheme with a ternary (three valued) shift register. This shift register supports population count operations very efficiently. We will refer to this detection as the Traffic Imbalance or simply Imbalance Scheme.

2.2.1 Motivation for the Two Mechanisms

Link Utilization Trigger

The idea behind this mechanism is that links in the network will typically be utilized somewhere around the steady state throughput of the network (in this topology, there are n links through each graph cut seperating the sources and sinks). When we have a hot spot developing, the utilization of those links very near the hot spot will increase. In fact, the link feeding into the hot memory (the "weakest link" in Pfister and Norton's terminology) will saturate almost immediately. The other links in the tree will exhibit higher utilization in the transient phase, but as the throughput of the network degrades, their utilization will also decrease below homogeneous traffic steady state values.

It seems plausible that two things could serve as a trigger: the "weakest link" reaching a very high utilization, and other links' utilizations exceeding the norm in the transient phase. Let us first consider triggering on the "weakest link" utilization. One might doubt the validity of this measure because it only detects the hot spot after it has started backing up the network. Further, the hot spot is detected at the "hot memory", which is farthest away from the sources in the network. What worsens the situations is that we cannot simply go into warning when we have high utilization for a brief period. The high utilization may be due to statistical fluctuation. If we were to take action, severe degradation of the throughput of this memory might result.

Detecting the hot spot during the transient phase is rather tricky. The primary criticism of this scheme comes from a control "stability" point of view. If this is the only trigger to the system, then if we miss the triggering during the transient phase, then the system will become stable in the degraded configuration. In order to prevent this situation, it is clear that this scheme must be paired with another scheme which will detect the system degradation. This other scheme need not have as rapid a response time, and the triggering described in the previous paragraph might be acceptable.

Because of the short duration of the "possible detection" window in the transient phase, and the relatively slow response at the hot link, we are not sure that this scheme will be practical. However, it is easily parameterizable, and we may be able to find a class of settings for which it is quite effective. But it is clear that there are many factors which cast into doubt the validity of this triggering technique to respond rapidly to a hot spot.

Traffic Imbalance Trigger

This mechanism is based on the observation that at the entry side of a multistage routing network, on any given link the hot spot intensity is very low. [1] That's why it is difficult to detect the hot spot at the source. As we pass through each stage of routers, the amount of hot spot traffic increases exponentially (by a factor of the switch arity at each level). At this rate, in a few stages it begins to cause a rather significant imbalance in the amount of traffic a given router sends to its output links. So not only does the traffic on the links to the hot spot increase in intensity, as the system backs up, but there is a commensurate decrease in traffic on the links not leading to the hot spot. Thus, we are able to detect the hot spot 1) while it is beginning to congest the network and 2) in steady state when it has degraded the network throughput.

This scheme has several significant drawbacks. Each switch must keep a significant amount of state and be able to do limited computation on it very rapidly. Detection of congestion is still sensitive to statistical variation in background traffic, and detection is still done several stages into the network, some distance from the sources where any flow control must be exercised. These concerns are mitigated by the following practical considerations. Specialized hardware can be built to keep the state, update it, and do the necessary computation. This hardware is far less complex than the "combining switch" hardware solution proposed for the RP3 machine. All detection schemes will be sensitive to statistical variations in traffic. However, if we are able to make our scheme sufficiently sensitive, we can afford to have each alarm only result in a small correction. Thus, we may be able to tolerate some false alarms. Hot spots are detected some distance from the sources but since the percentage of hot spot traffic increases exponentially as we move into the network, this number of stages may be very small. Futhermore, if we have hot spots with more limited host participation, we may be able to detect them even closer to the sources.

[1] This statement make the typical assumption that the hot spot is spread over a large number of processors.

2.3 Simulation of Two Triggering Mechanisms

The two triggering mechanisms described above were implemented. A study was undertaken to determine their sensitivity and robustness. The objective here is to determine how early can we detect the onset of congestion and at that level of sensitivity, how often are we going to have "false alarms". This tradeoff will have major effect in how we can respond to these alarms while maintaining system stability. It will also effect how sensitive we can afford to make the triggering mechanism because we clearly do not want to degrade the system throughput in the balanced traffic case.

An ideal detection scheme would detect all of the hot spots at the sources (where any flow control must be exercised) and would not be susceptible to false alarms due to statistical flucuations in the traffic. It would also detect the hot spots very early (just as they were starting and before they congested the network. The two proposed detection schemes are evaluated in this framework. Both of the schemes presented suffer from serious limitations present in any distributed detection scheme. These limitations make some false alarms inevitable. However, the effects of these limitations can be mitigated by trading some response time performance of the detection scheme for enhanced resistance to transients.

One advantage of a distributed detection scheme is as follows. In a multi-stage routing network, one can make the observation that localized detection gives us some information as to who the sources of the hot spot are. Any switch's position in the network uniquely determines which sources supply traffic and which sinks receive traffic via this switch. This observation is relied upon heavily when implementing throttling mechanisms.

2.3.1 Simulation Results for Two Triggering Mechanisms

We present two kinds of simulation results. First, we present the number of warnings (false alarms) for the two detection schemes at various threshold settings. In the case of the Link Utilization scheme, we also note the decay constant values. This tells us what level we will

Threshold	Warnings
0.60	158.4
0.65	74.1
0.70	42.2
0.75	5.6
0.80	1.6
0.85	0.2
0.90	0.0

Table 2.1: Number of False Alarms with Traffic Imbalance Detection

have to set the appropriate parameters to in order to have a reasonably low number of false alarms. After that, we present the time to detect a hot spot with a variety of settings of the threshold. This allows us to see which scheme is in fact more sensitive at a tolerable level of robustness.

Both detection schemes were simulated. The length of the simulation runs were 1000 packet times, run for 5 iterations. The number of alarms in that period was averaged over the number of iterations. The background intensity of the traffic was 0.6 (meaning for a given source, on the average six packets were sent in every ten time steps). The minimum recurrence time (for a switch to give two warnings) was 50 packet times. The simulated network had eight ports. This meant that the absolute maximum number of false alarms possible was 12 $*$ 20 or 240 warnings.

The number of false alarms were measured under the traffic imbalance detection scheme. The results of these simulations are presented in Table 2.1. It is clear that for thresholds greater than around 0.75 minimal numbers of false alarms detected.

The number of false alarms were measured under the link utilization detection scheme. The results of these simulations are presented in Table 2.2. Here we can see that for a wide range of settings, the maximum number of false alarms was achieved. We note that if the decay constant is greater than or equal to the threshold, the threshold will be exceeded at a switch every time that switch transmits a packet. Thus, the maximum number of false

Threshold	Decay Constant	Warnings
0.70	0.7	240.0
0.70	0.6	240.0
0.80	0.8	240.0
0.80	0.7	240.0
0.80	0.6	240.0
0.90	0.7	240.0
0.90	0.6	240.0
0.90	0.5	233.0
0.95	0.6	240.0
0.95	0.5	223.2
0.975	0.60	240.0
0.975	0.55	239.5
0.975	0.50	212.0

Table 2.2: Number of False Alarms with Traffic Imbalance Detection

alarms will be achieved. For a small range of thresholds and decay times, it was possible to keep the number of false alarms to a reasonable levels.

We could not further decrease the decay constants because doing so would make the thresholds unrealizable. Further reduction of the thresholds, as one might expect, failed to decrease the number of false alarms. There were no settings at which the Link Utilization scheme yielded acceptable performance.

2.3.2 Summary

From the simulation results, we are able to draw several conclusions. We see that the traffic imbalance detection scheme will allow us to set the threshold anywhere above ≈ 0.8 and result in only a minimal number of false alarms. Conversely, we see that the link utilization scheme requires very high thresholds, and sharp decay factors in order to prevent false alarms. In fact, there were no settings that yielded acceptable performance. It is possible that the there are such settings. But such settings are also likely to retard rapid response to traffic imbalance.

It is important to note that the simulation study was performed in high load conditions. These conditions would tend to degrade the performance of the link utilization scheme. However, these conditions are the most important conditions for the detection scheme to work well because so much throughput is at stake. We also note that the high traffic levels did not appear to impair the performance of the traffic imbalance scheme. We conclude from this study that the traffic imbalance scheme exhibits more tolerance to high traffic loads. Therefore, if the traffic imbalance scheme exhibits acceptable sensitivity, our experimentation will it as the detection method.

Chapter 3

The Impact of Buffer Policies on Network Behavior in the Presence of Hot Spots

3.1 Queue Policies

In a typical routing network, all queues in the system are assumed to behave in a first-in-first-out (FIFO) manner. Fast FIFO queues are very easy to build because of their simplicity. Furthermore, using this policy guarantees that all packets will eventually be delivered (assuming some sort of fair arbitration when there is contention for links).

However, FIFO policies have significant drawbacks. Such policies will not result in the most efficient use of network bandwidth. That is, sendable packets[1] will sometimes be forced to wait. We will refer to this phenomenon as "blocking." In a 2-by-2 router, this might happen if the two packets at the heads of the queues both need to be transmitted on output 0. Only one of them can be transmitted in the current time step. Packets in either queue that could be sent on output 1 will not be sent in the current time step. Thus we are wasting network bandwidth because we require the queues to behave in a FIFO manner.

[1] A packet that could be transmitted over an idle link in the current time step.

The networks we consider are built from 2-by-2 routers. Each router contains two queues on its input links. In this study, we consider three types of queue policies: FIFO, Random, and Modified FIFO. The FIFO scheme is the ordinary first-come-first-serve policy. The Random policy selects a packet at random from each of the queues and attempts to transmit them. If there is contention, only one of the packets is sent. The packets "behind" the transmitted packet in the queue all move up one place. In the Modified FIFO scheme, the queues are ordered randomly by the arbiter. Then, for each queue in succession, we send the first sendable packet in queue. This does not guarantee that a maximal number of packets is sent in each cycle because we are examining the queues sequentially. However, it does eliminate the "blocking" behavior in the queues. This scheme makes the behavior of the routers similar to that of routers with "inside buffers". However, the scheme we propose is more flexible because more buffer sharing is allowed.

3.2 Performance Under Balanced Traffic

Several simulations were performed in an attempt to determine how the queue policies affected network performance under balanced loads. We simulated the network under a balanced load, while varying the queueing policy. For each policies, we examined the normalized network throughput and the average delay. The results of these simulations are shown in Table 3.1.

We found, not surprisingly, that the Modified FIFO policy outperformed both of the other policies. For all three sizes of networks, the Modified FIFO scheme exhibited superior throughput and delay characteristics. This is not surprising, because for any given switch state, the Modified FIFO policy will always result in the transmission of an equal or greater number of packets than either the FIFO or Random schemes. We also note that implementation of the Modified FIFO requires a significant amount of computation in each switch. This was reflected in the much greater amount of time required to perform simulations of this policy. The simulations with the Modified FIFO policies typically required twice the computation time as the other simulations.

The networks we consider are built from 2-by-2 routers. Each router contains two queues on its input links. In this study, we consider three types of queue policies: FIFO, Random, and Modified FIFO. The FIFO scheme is the ordinary first-come-first-serve policy. The Random policy selects a packet at random from each of the queues and attempts to transmit them. If there is contention, only one of the packets is sent. The packets "behind" the transmitted packet in the queue all move up one place. In the Modified FIFO scheme, the queues are ordered randomly by the arbiter. Then, for each queue in succession, we send the first sendable packet in queue. This does not guarantee that a maximal number of packets is sent in each cycle because we are examining the queues sequentially. However, it does eliminate the "blocking" behavior in the queues. This scheme makes the behavior of the routers similar to that of routers with "inside buffers". However, the scheme we propose is more flexible because more buffer sharing is allowed.

3.2 Performance Under Balanced Traffic

Several simulations were performed in an attempt to determine how the queue policies affected network performance under balanced loads. We simulated the network under a balanced load, while varying the queueing policy. For each policies, we examined the normalized network throughput and the average delay. The results of these simulations are shown in Table 3.1.

We found, not surprisingly, that the Modified FIFO policy outperformed both of the other policies. For all three sizes of networks, the Modified FIFO scheme exhibited superior throughput and delay characteristics. This is not surprising, because for any given switch state, the Modified FIFO policy will always result in the transmission of an equal or greater number of packets than either the FIFO or Random schemes. We also note that implementation of the Modified FIFO requires a significant amount of computation in each switch. This was reflected in the much greater amount of time required to perform simulations of this policy. The simulations with the Modified FIFO policies typically required twice the computation time as the other simulations.

Queue Policy	Network Size	System Throughput	Average Delay
FIFO	8	0.65	16.5
	32	0.61	23.2
	128	0.58	28.9
Random	8	0.67	16.8
	32	0.63	23.6
	128	0.61	29.9
Mod FIFO	8	0.84	15.5
	32	0.79	23.0
	128	0.76	29.3

Table 3.1: Balanced Load Network Performance.

3.3 Performance in the Presence of Hot Spots

The results presented above show that the Modified FIFO policy outperforms the others under a balanced traffic load. Of course, we were primarily interested in measuring the network performance in the presence of hot spots. Not only did we want to see how the queue policy would affect steady state throughput and delay in the presence of hot spots, we also wanted to see how it would affect congestion and dissipation of congestion. Two separate studies were performed, and the results are summarized in Tables 3.2 and 3.3.

3.3.1 Steady State Performance

In the first study, we sought to compare the steady state performance of the different queue policies in the presence of a hot spot. For the various network sizes, a hot spot intensity of 16% was used. The source at each network input sent 16% of its traffic to hot spot. The remainder of its traffic was distributed evenly amongst all of the outputs, including the hot port. The homogeneous background traffic had an intensity of 0.4. Thus, on the average, each port transmitted four packets in every ten cycles. The results are shown in Table 3.2.

Queue Policy	Network Size	System Throughput	Average Delay
FIFO	16	0.293	30
	64	0.092	119
Random	16	0.288	33
	64	0.089	129
Mod FIFO	16	0.296	30
	64	0.091	123

Table 3.2: Hot Spot Load Network Performance.

3.3.2 Transient Performance

In the second study, we attempted to characterize the time to congest and decongest the network for the various queue policies. We know that any hot spot detection scheme will take some time to be triggered. Therefore, any queue policy that extends the congestion time is of interest. Furthermore, any queue policy that shortens the dissipation time will also be of interest, because that will aid any control mechanism in dealing with the congestion. To these ends, we measured the time to congest and decongest the network under the three queueing policies. The results are presented in Table 3.3. The times to congest are measured from the time the hot spot was introduced to the time the last stage of switches' buffers become 80% full. The time to decongest is measured fromt the time the hot spot is removed to the time the first stage of switches (in the saturating tree) is less than 80% full.

From these results, it seems clear that neither of the more complicated queue policies offers an appreciable advantage over the FIFO policy in slowing the onset of congestion. The Random policy generally required longer periods to congest, yet also required longer periods to dissipate congestion. The Modified FIFO policy performed in a similar fashion. With all three network sizes, the Modified FIFO scheme delay full congestion significantly. However, at the end of the hot spot, this time was paid back in terms of much longer dissipation times.

Queue Policy	Network Size	Congestion Time	Decongestion Time
FIFO	16	265.0	48.2
	32	86.9	107.9
	64	65.9	222.5
Random	16	261.3	65.8
	32	107.9	155.7
	64	79.0	301.4
Mod FIFO	16	381.4	75.8
	32	118.4	173.3
	64	87.5	330.0

Table 3.3: Congestion and Decongestion Times.

These results were very similar to what we expected. For example, we expected the Modified FIFO to perform significantly better (in terms of congestion time) than the other two policies as it allowed for more full use of the network buffering by hot spot packets. The bad effects of using all of the network buffering for hot spots is that dissipation takes longer - we must drain an entire network full of hot spot packets. In the case of decongesting the network, it seems that the simple FIFO policy performs the best of all. In all the network configurations simulated, decongestion time was significantly shorter with the FIFO queueing disipline. Thus in terms of maximizing the effectiveness of a network control algorithm, FIFO queue policy is attractive because of its simplicity.

3.4 Summary

Three queueing disciplines were examined: FIFO, Random, and Modified FIFO. The Modified FIFO scheme did allow significantly higher traffic rates to be sustained in the balanced traffic case. We note that the Random and Modified FIFO schemes were considered primarily because they might result in better network performance in the presence of hot spots. These schemes performed better in terms of congestion time. However, both performed poorly in terms of dissipation time. Since they offer no significant performance advantage in terms of hot spot congestion control, the FIFO scheme is comparatively more attractive because of its simplicity. Therefore, we conclude that the Random and Modified FIFO schemes do not enhance network performance in the presence of hot spots. The FIFO scheme performed comparably in all the studies undertaken. Thus, the FIFO policy is the most attractive queueing policy in the presence of hot spots.