EE482B Project

Description

For the project, you must propose and perform an investigative study of some aspect of high-radix interconnection networks. As you will see from the suggested topics below, your study may be in any number of areas such as topology, flow control, router architecture, etc. Your first goal is to decide upon an idea you wish to investigate. Don't make this too broad or you may have difficulty finishing your study. Your idea, however, should be new and should be (depending on the results) worthy of conference submission.

First, you must decide upon an idea to investigate. Next, you need to establish a hypothesis and a methodology to prove or disprove that hypothesis. You must then perform the experiments or analysis to evaluate your hypothesis. Finally, you will be required to write a report including your original idea, hypothesis, methodology, and results.

Checkpoints

Checkpoint 1 (due May 14, 2003):
For checkpoint 1 you must turn in a project proposal that lists the members of your group and a description of your project including at least (1) the problem you are solving, (2) your solution to the problem, (3) your hypothesis about how your solution will work, and (4) your methodology for testing your hypothesis.

Checkpoint 2 (due May 21, 2003):
By checkpoint 2, you should be midway into your evaluation with any experimental software at least partially working and some preliminary results. You don't need to turn anything in for this checkpoint, but you must come to office hours or make an appointment to review your progress with Prof. Dally.

Final Report (due June 6, 2003):
You must submit a written report on your project. This report should be no longer than 10 pages (plus appendices if needed) and should include a description of your problem area, your solution, hypothesis, methodology, findings, and conclusions.

Presentation:
Each group must present their project and findings in class. These presentations will take place on May 28 and June 2, so you must have your work largely complete, except for writing the report, by your presentation date.

Topics in High-Radix Interconnection Networks

All projects this year will address a topic involving *high-radix* interconnection networks. That is, networks that use routers with a large number of channels (i.e., a high degree or radix). As technology has evolved, the *aspect ratio* of a router component, the ratio $B_r t_r / L$ has increased by two orders of magnitude. (B_r here is the pin bandwidth of a router component). Intuitively the aspect ratio is amount of data (in packets) that is in the router at any given point in time. As aspect ratio has increased, the degree or radix of an *optimal* topology has also increased. A good topology should balance the serialization latency of a packet $L\delta/B_r$ against the header latency count $Ht_r \geq t_r \log_{\delta/2} N$. As B_r has increased, serialization latency has decreased, requiring a balanced design to decrease hop count, by increasing δ. As a result, we expect routers with total degree (in+out) to be $\delta \geq 512$ in the near future and over 2048 by the end of the decade. This is in contrast to routers with degrees of $\delta = 8$ to 16 for most routers built over the last decade.

This trend toward building interconnection networks with a high degree (or radix) presents many new challenges in the design of networks and the routers to implement them. Many of the conventional approaches to building networks cease to work when the node degree is increased by an order of magnitude. This year our projects will each address one of the challenges posed by high-radix networks:

Router Architecture:

Internal switch architecture: to connect 256 input ports to 256 output ports, we could use a 256x256 crosspoint switch (possibly with speedup) or we could employ a multi-stage structure. Propose alternative internal switch configurations and compare them in terms of cost, delay, throughput, and ease of scheduling.

Allocator design: How do we design a 256 x 256 allocator to allocate virtual channels to packets and to allocate switch resources to flits? A good allocator needs to operate in a small number of cycles (preferably one) and be pipelined to generate a new allocation each cycle. Propose alternative allocator designs and compare them in terms of cost, latency, and quality of allocation. You may consider taking advantage of the fact that each input typically has only a small number X of packets to bid each cycle ($X << \delta_o$). Consider alternative ways of separating the allocator – into more than two parts. You may want to address both switch architecture and allocator design together since the two problems are coupled. Consider methods to pipeline an allocator. When pipelining consider the issue of re-bidding a request while it is still pending in later pipeline stages. How do you handle the case where the request wins an earlier allocation?

Topology

What topologies are best suited to take advantage of high-radix routers. An obvious choice, that you can use as a baseline, is a folded Clos network, possibly channel sliced, to balance T_s and T_h . Can you devise a more efficient topology that uses high-radix routers. You might consider a high-radix Cayley graph, a butterfly network with randomization, a variation on a torus network, or some topology of your own creation. For any new topology you devise, you should explain how to route packets on this topology, its throughput and latency under both benign and adversarial traffic patterns, and its fault tolerance. You may also consider the role of concentration in these networks.

You might also consider how the choice of topology is influenced by the use of electrical signaling for short links and more expensive (10x more expensive) optical signaling for long links.

Routing

Given a high-radix topology, e.g., a channel sliced (r,256,256) folded Clos, how does one route? A high-radix Clos presents many routing challenges (other high radix topologies present different sets of challenges). For example, when routing *upward* in the folded Clos, there are 256 possible choices at each stage. For an adaptive router, its hard to objectively choose among such a large set. With source routing, this results in a large routing header. Is it possible to restrict the output set, simplifying the routing decision, without degrading performance?

For a high-radix topology using routing tables, how are the tables to be filled? A good route set balances load well across a large number of traffic patterns and offers near optimum bandwidth and latency. In general one would like to use the smallest possible route set that meets a certain set of performance objectives. Propose and evaluate alternative algorithms for generating routing tables. Consider issues that arise when the number of table entries for each source-destination pair is limited to a small number (e.g., 4). Consider the tradeoff between source routing and hop-by-hop routing.

Adaptive routing in high-radix networks: How best can a routing algorithm take advantage of network state information in a high-radix network. Are there methods to effectively use more global information in the routing decision. How can a routing function be implemented in a clock cycle that considers a large number of alternative routes.

Flow control

What flow control methods are most appropriate for high-radix routers. Does the high radix impact the choice of packet vs. flit buffering? or of no buffering? Given a fixed amount of buffering per node, what is the most efficient way to distribute it? Deep buffers vs. many shallow buffers? Dedicated buffers per input vs. buffers shared across input sets?

Effect of credit delay: It has been observed that delaying the return of credits in a virtual-channel router reduces peak throughput. However, there is not analytical model of this phenomena. Develop such a model and validate it. Use your model to address tradeoffs in router design -- e.g., number of flits per virtual channel vs credit delay.