Problem 1: Potpourri (55 pts)

A) [10 pts] Thread prioritization

Suppose we are running a **multithreaded** application where threads are part of the same application on a multicore processor. The memory controller is shared between the cores.

1) Provide one reason why prioritizing a memory non-intensive thread over a memory-intensive one in the memory controller would improve performance. If this is not possible, write N/A and explain why.

Prioritizing latency-sensitive (memory non-intensive) threads can increase system throughput

2) Provide one reason why doing the same would degrade performance. If this is not possible, write N/A and explain why.

Can delay the critical/bottleneck thread which may not be memory non-intensive

B) [4 pts] Memory bandwidth

Under what conditions would an application's performance increase linearly as memory bandwidth is increased?

If memory bandwidth is the performance bottleneck

C) [4 pts] Fat trees

What problem does the fat tree interconnect solve that is present in the tree interconnect?

High link contention between root and subnodes – a fat tree increases the bandwidth of these links

D) [10 pts] Interconnect

You are observing a system with many processing elements connected through a network. There is currently no activity on the network (no messages are being sent). On cycle 10, one of the cores generates a message destined for a cache bank somewhere else on the network. You observe the network on cycle 20 and see that this message has not departed the source location. Assume that all components are enabled (not powered off) and operating at full speed. There are no other messages present in the system at this time. Why could this be?

The system is using circuit switching, and there is a large delay to set up all links between source and destination.

E) [12 pts] Slack

As you recall, we have discussed the idea of slack based prioritization for on-chip interconnects in class. In fact, you reviewed a paper that introduced this concept. The key idea was to prioritize the packet that has the least slack over others in the router, where the slack of a packet (ideally) is defined as the number of cycles the packet can be delayed without hurting performance.

The concept of slack is actually more general. It can be applied to prioritization at any shared resource, assuming the slack of a "memory request" can be estimated well.

1) Suppose we have a mechanism that tries to estimate the exact slack of a memory request when the request is injected into the shared resources. Provide two reasons why estimating the exact slack of a packet might be difficult:

The exact latency of the request may not be known at the time of injection – the slack may change based on the state of the shared resources and the decisions made by them

How much the packet would affect performance may not be known at the time of injection – the overlap of latency of the packet may not be known at the time of injection

2) What performance issue can slack-based prioritization cause to other processors in the system? Why?

Can cause starvation to some threads

3) How can you solve this problem?

Batching

F) [5 pts] Dataflow

What is the purpose of token tagging in dynamic dataflow architectures?

Supporting re-entrant code. Ensuring that tokens come from same context.

G) [10 pts] Alpha 21264

The Alpha 21264 had a "Prefetch and evict next" instruction that "prefetched data into the L1 cache except that the block will be evicted from the L1 data cache on the next access to the same data cache set."

1) What access patterns could benefit from this instruction? Explain well.

Streaming or striding access pattern (no data reuse)

2) The Alpha 21264 processor employed a predictor that predicted whether a load would hit or miss in the cache before the load accessed the cache. What was the purpose of using this predictor? Explain concisely but with enough detail.

Allow speculative scheduling of consumers of the load