**Abstract**

Targeting Network-on-Chip (NoC) based multicore, we propose two network prioritization schemes that can cooperatively improve performance by reducing end-to-end off-chip memory access latencies:

- Scheme 1: Prioritizes memory response messages such that, in a given period of time, messages of an application that experience higher latencies than the average message latency for that application are expedited and a more uniform memory latency pattern is achieved.
- Scheme 2: Prioritizes the request messages that are destined for idle memory banks over others with the goal of improving bank utilization and preventing long queues from being built in front of the memory banks.

These two network prioritization-based optimizations together lead to uniform memory access latencies with a low average value. Our experiments with a 4x8 mesh network-based multicore show that, when applied together, our schemes can achieve 15%, 10% and 13% performance improvement on memory intensive, memory non-intensive, and mixed multi-programmed workloads, respectively.

**Motivations**

**Motivation 1:** Some memory accesses experience much higher delays than the others. For these messages, the contribution from the NoC is very significant.

**Motivation 2:** Bank loads are non-uniform. At any instant of time, while some banks are busy, other banks are underutilized.

A snapshot of the states of three banks controlled by one of the memory controllers. As can be seen, Banks 0 and 1 have requests to be serviced, whereas Bank-2 is idle.

**Implementation**

Scheme-1: At each router/memory-controller, once the message is ready to be sent out, the age field value is updated as follows:

\[ \text{age} = \text{age} + \left( \frac{\text{cycles}_{current} - \text{cycles}_{message_entry}}{\text{FREQ} \times \text{MULT}} \right) \times \text{local_frequency} \]

Scheme-2: Each node exploits “local information” to estimate the pressure imposed by the requests on the memory banks. Specifically, each router keeps and updates a table that records the number of off-chip memory requests sent from this router to each bank in the last T cycles.

**Summary**

- Identified
  - Some memory accesses suffer long delays and block the core
  - Banks mostly idle, bank utilization varies
- Proposed two schemes
  1. Network prioritization and pipeline bypassing of “late” memory response messages to expedite them
  2. Network prioritization of memory request messages to improve bank utilization
- Demonstrated
  - Scheme 1 achieves 9% average speedup
  - Scheme 1+2 achieves 13% average speedup

**Results**

32-core system

- Normalized Weighted Speedup
  - Scheme-1
  - Scheme-1 + Scheme-2

- Workloads (Mixed)
  - Normalized Weighted Speedup
  - (High Memory Intensity)
  - (Low Memory Intensity)

- Fraction of total accesses
  - Delay (cycles)

- Few accesses with high delays