Non-Uniform Cache Architectures
Designing Large Caches

- UMA vs. NUMA architectures
- Large cache latency dominated by wire delay
- Design Issues for large caches
  - Mapping: How many banks, and how lines map to banks
  - Search: How to find the set of possible locations for a line
  - Movement: Static (line always in same bank) vs. dynamic (line moves while in cache or across different lifetimes in the cache)
- Options for designing a 16MB cache: Kim paper Figure 1
Uniform Cache Architecture (UCA)

- Similar to a traditional cache
- Uses sub-banks, but limited by number of ports
- Moore’s law scaling leads to increasing wire delay
- Latency and performance: Kim Paper Table 2
Multi-Level Uniform Cache Architecture (ML-UCA)

- Cache split into two levels: L2 and L3
- Both levels aggressively banked to support parallel accesses
- Inclusion is enforced
  - L3 includes everything in L2
  - Simplifies design but consumes extra space due to data duplication
- Latency and performance: Kim Paper Table 8
Statically-Mapped Non-Uniform Cache Architecture (S-NUCA)

- Aggressively banked and supports non-uniform access
- No inclusion (avoid duplication)
- Mapping of data into banks is statically pre-determined based on the block index
- Could use private per-bank channel or 2D switched network (Kim Paper Figure 1c, 1d)
- Latency and performance: Kim Paper Table 3, 4
Dynamically-Mapped Non-Uniform Cache Architecture (D-NUCA)

- Mapping of data into banks is dynamic: data can be mapped to many banks within the cache
- Cache management attempts to have most requests served by faster banks
- Frequently used data promoted to faster banks
- Could be implemented by splitting cache sets across banks, one way per bank (Figure 4)
- Latency and performance: Kim Paper Table 5
- Comparing performance of all options: Figure 6
NUCA on CMPs

- New challenges with chip multiprocessing:
  - Private vs. shared caches
  - Data: private, shared read-only, shared read-write
  - How to allow replication

- Go to Chishti et al. Slides
Reading Assignment

- Arthur Veen, "Dataflow Machine Architecture," ACM Computing Surveys, 1986 (Read sections 1, 2, 3 and skim the rest of the paper)
- Gregory Papadopoulos and David Culler, "Monsoon: An Explicit Token-Store Architecture," ISCA, 1990 (Read)