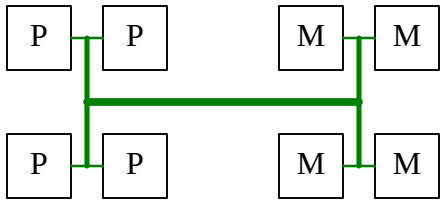
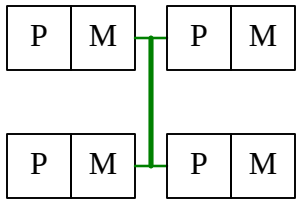
UMA vs. NUMA



UMA



NUMA

- "Dance Hall"
- SMP
- Latency = O(log(N))

- P/M nodes on network
- Modern Architectures
- Latency =
  - O(1) for local
  - O(log(N)) for remote

Managing Latency

- Caches
- Prefetching
- Multithreading
- Alternate consistency models
- NUMA only:
  - locality!

Types of Locality

- Data locality
  - static: try to place data intelligently
  - dynamic: migrate data to where it's needed
- Code locality
  - static: compiler assigns code to specific nodes
  - dynamic: remote procedure calls

Papers

- DASH, FLASH
  - serious caching
- Alewife
  - aggressive latency tolerance
- RAW
  - static everything: let the compiler do the work
- J-Machine
  - object oriented-style dynamic code locality
- M-Machine
  - multithreading, caching

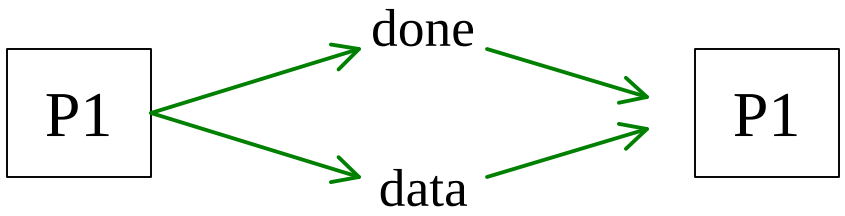
Memory Consistency Models

- Start with data = done = 0

Processor 1	Processor 2
`data = 27;`	`while (!done);`
`done = 1;`	`print data;`

- What output do you expect?

How Things Can Go Wrong



- Fire and forget leads to problems!
- For sequential consistency, need to wait for one write to complete before starting next one

Relaxed Consistency Models

- Start with data = 0

Processor 1	Processor 2
`data = 27;`	`sync;`
`sync;`	`print data;`

- Define special synchronization operations
- Reads/Writes ordered w.r.t. syncs

Papers

- Tutorial
- Release Consistency
- Highly opinionated paper advocating SC