CACHE COHERENCE IN MULTIPROCESSOR SYSTEMS

Interconnects provide basic mechanisms for data transfer.

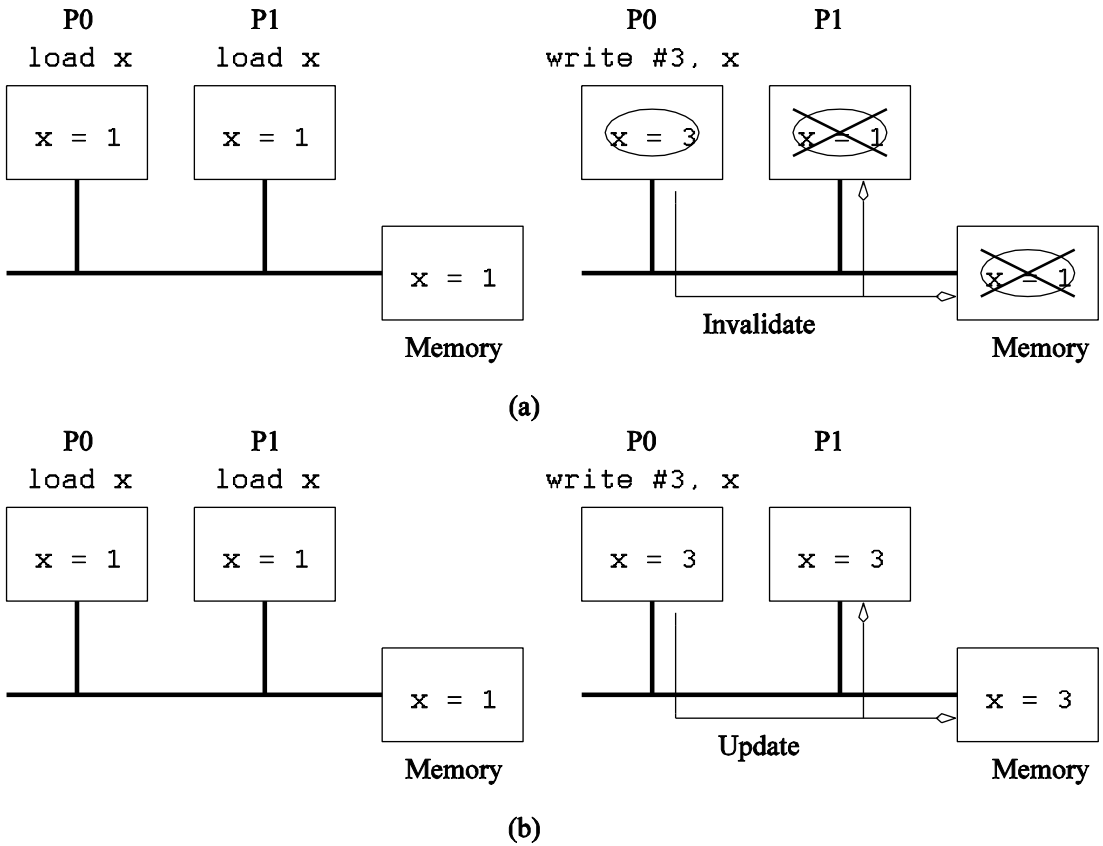
In the case of shared address space machines, additional hardware is required to coordinate access to data that might have multiple copies in the network.

The underlying technique must provide some guarantees on the semantics.

This guarantee is generally one of serializability, i.e., there exists some serial order of instruction execution that corresponds to the parallel schedule.

CACHE COHERENCE IN MULTIPROCESSOR SYSTEMS

When the value of a variable is changes, all its copies must either be invalidated or updated.



Cache coherence in multiprocessor systems: (a) Invalidate protocol; (b) Update protocol for shared variables.

PURDUE NExT

CACHE COHERENCE: UPDATE AND INVALIDATE PROTOCOLS

If a processor just reads a value once and does not need it again, an update protocol may generate significant overhead.

If two processors make interleaved test and updates to a variable, an update protocol is better.

Both protocols suffer from false sharing overheads (two words that are not shared, however, they lie on the same cache line).

Most current machines use invalidate protocols.

MAINTAINING COHERENCE USING INVALIDATE PROTOCOLS

Each copy of a data item is associated with a state.

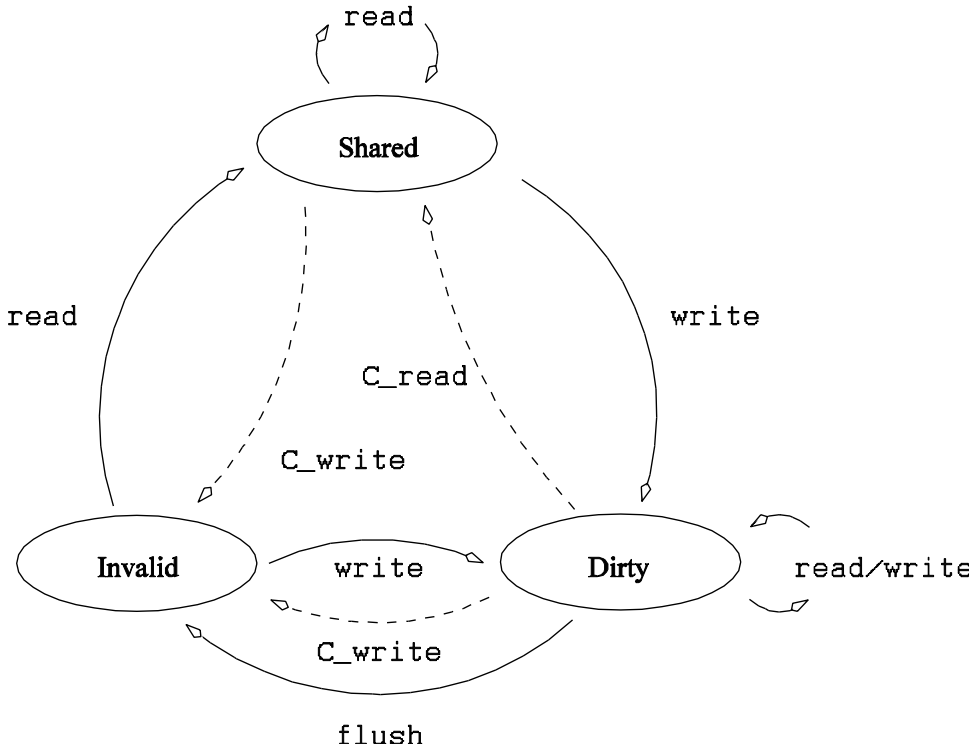
One example of such a set of states is, shared, invalid, or dirty.

In shared state, there are multiple valid copies of the data item (and therefore, an invalidate would have to be generated on an update).

In dirty state, only one copy exists and therefore, no invalidates need to be generated.

In invalid state, the data copy is invalid, therefore, a read generates a data request (and associated state changes).

MAINTAINING COHERENCE USING INVALIDATE PROTOCOLS



State diagram of a simple three-state coherence protocol.

PURDUE NExT

MAINTAINING COHERENCE USING INVALIDATE PROTOCOLS

Time ↓

Instruction at Processor 0	Instruction at Processor 1	Variables and their states at Processor 0	Variables and their states at Processor 1	Variables and their states in Global mem.
				x = 5, D y = 12, D
read x		x = 5, S		x = 5, S
	read y		y = 12, S	y = 12, S
x = x + 1		x = 6, D		x = 5, I
	y = y + 1		y = 13, D	y = 12, I
read y		y = 13, S	y = 13, S	y = 13, S
	read x	x = 6, S	x = 6, S	x = 6, S
x = x + y		x = 19, D	x = 6, I	x = 6, I
	y = x + y	y = 13, I	y = 19, D	y = 13, I
x = x + 1		x = 20, D		x = 6, I
	y = y + 1		y = 20, D	y = 13, I

Example of parallel program execution with the simple three-state coherence protocol.

PURDUE NExT