Flynn’s Hardware Taxonomy

MENTION FLYNN IEEE TRANSACTIONS ON COMPUTERS – “some computer organizations and their effectiveness”

SISD **-** single processor executes a single instruction on data stored in a single memory

- uniprocessors

SIMD **-** single machine instruction controls the simultaneous execution of a number of processing elements, each element has its own data memory and each instruction is executed on a different set of data by the different processors.

- vector and array processors

MISD **-** data is transmitted to a set of processors, each processor executes a different instruction sequence

- never been implemented

MIMD **-** a set of processors simultaneously execute different instruction sequences on different data sets

- SMPs, clusters, and NUMA systems

MIMDs can be divided by the way their processors communicate: shared memory or distributed memory

Shared **-** the processors communicate with each other through the shared memory

Distributed **-** computers communicate via fixed paths or a network

SMPs **-** multiple similar processors in one computer interconnected by a bus

- PROBLEM – cache coherence

NUMA **-** shared-memory multiprocessor

- access time from a processor to a memory word varies depending on the location of the word

Clusters **-** group of interconnected computers working together (illusion of one machine)

Taxonomy of Parallel Computing Paradigms

Class question: What is a paradigm?

Paradigm:

It is simply a model of the world that is used to formulate a computer solution to some problem. Paradigms are useful in parallel computer architecture as well as parallel programming because it controls the complexity of the details.

Asynch Vs. Synch :

Coordination is required in parallel programs when a certain task depends on others.

Synchronous or lockstep coordination is implemented in the hardware by enabling all operations at once in a way that removes the dependency of one operation on the other.

Asynchronous coordination relies on coordination mechanisms called *locks* to coordinate processors.

Vector/Array paradigm is another name for pipelining since numerical problems involving matrices and such require breaking down the problem into small stages. Pipelining for a parallel computer is similar to that of a processor, except in the sense of scale.

SIMD:

SIMD (Single Instruction, Multiple Data) means all the processors do the same thing at the same time or else they remain idle. SIMD uses two phases over and over to solve the problem of managing the data:

Phase 1: Partition and distribute the data.	Called data partitioning
Phase 2: Process the data in parallel.	Called data parallel processing

Now this might seem to be a very trivial at first, but this paradigm is extremely useful in solving problems that have lots of data to be updated on a wholesale basis. This is especially powerful in vector and matrix calculations.

Systolic:

Invented in the 80s by H.T. Kung at Carnegie Mellon University, a Systolic parallel computer is a multiprocessor which distributed data from memory to an array of processors before returning to memory.

This paradigm incorporates features of both SIMD and vector/array paradigms. In this mode, a parallel computer can exhibit very high speeds by avoiding input/output bottlenecks. This is done by churning the data among the processors as much as possible before returning to memory.

The most general form of parallelism is asynchronous parallelism, since the processors operate without regard to any global synch.

MIMD:

As stated earlier, an MIMD organization means that processors are doing different instructions to different data at the same time. This enables each processor to work on its data freely without relying on other processors for information.

However, certain mechanisms called locks are needed if lets say that two processors need access to one spot in memory. This is done by mutual exclusion, which only allows one processor to access info at a point in time.

Other locks such as read and write locks are used when the MIMD incorporates a shared memory.

MIMD is most useful in large-grained problems because of the overhead in passing data and control from task to task. When I say large-grained, I am speaking of grain size, which is equal to the number of serial instructions done by one processor.

Interconnection Networks (IN)

The processors themselves in a parallel system can range from simple 1-bit processors to some of the advance architectures we have seen in the previous case studies. While the speed and capacitance can vary, the most significant difference in these machines is how they are connected to each other.

In shared memory, every processor must be connected to the main memory. However, in distributed memory networks, each processor can be fully connected to each other processor or the processors are somewhat connected to each other and in some cases the data must 'hop' through other processors to reach its destination processor.

Distributed memory is set into two different classes: static and dynamic. The difference is in when a connection is needed between processors. Static IN's always have a connection whether it is direct (fully connected) or through another processor. Dynamic IN's make the connection on the fly.

IPC: (Inter Process Communication)

In computer science, inter-process communication or inter process communication (**IPC**) refers specifically to the mechanisms an operating system provides to allow the processes to manage shared data.