Lecture 9

Hardware and software parallelism

Hardware parallelism is defined by machine architecture and hardware multiplicity i.e., functional parallelism times the processor parallelism .It can be characterized by the number of instructions that can be issued per machine cycle. If a processor issues *k* instructions per machine cycle, it is called a *k-issue* processor. Conventional processors are *one-issue* machines. This provide the user the information about **peak attainable performance**. Examples. Intel i960CA is a three-issue processor (arithmetic, memory access, branch). IBM RS -6000 is a four-issue processor (arithmetic, floating-point, memory access, branch).A machine with *n k*-issue processors should be able to handle a maximum of *nk* threads simultaneously.

Software Parallelism

Software parallelism is defined by the control and data dependence of programs, and is revealed in the program's flow graph i.e., it is defined by dependencies within the code and is a function of algorithm, programming style, and compiler optimization.

The Role of Compilers

Compilers used to exploit hardware features to improve performance. Interaction between compiler and architecture design is a necessity in modern computer development. It is not necessarily the case that more software parallelism will improve performance in conventional scalar processors. The hardware and compiler should be designed at the same time.

Program Partitioning & Scheduling

Grain size and latency

The size of the parts or pieces of a program that can be considered for parallel execution can vary. The sizes are roughly classified using the term "granule size," or simply "granularity." The simplest measure, for example, is the number of instructions in a program part. Grain sizes are usually described as fine, medium or coarse, depending on the level of parallelism involved.

Latency

Latency is the time required for communication between different subsystems in a computer. Memory latency, for example, is the time required by a processor to access memory. Synchronization latency is the time required for two processes to synchronize their execution. Computational granularity and communication latency are closely related. Latency and grain size are interrelated and some general observation are

- As grain size decreases, potential parallelism increases, and overhead also increases.
- Overhead is the cost of parallelizing a task. The principle overhead is communication latency.
- As grain size is reduced, there are fewer operations between communication, and hence the impact of latency increases.
- Surface to volume: inter to intra-node comm.

Levels of Parallelism

Instruction Level Parallelism

This fine-grained, or smallest granularity level typically involves less than 20 instructions per grain. The number of candidates for parallel execution varies from 2 to thousands, with about five instructions or statements (on the average) being the average level of parallelism.

Advantages:

There are usually many candidates for parallel execution

Compilers can usually do a reasonable job of finding this parallelism

Loop-level Parallelism

Typical loop has less than 500 instructions. If a loop operation is independent between iterations, it can be handled by a pipeline, or by a SIMD machine. Most optimized program construct to execute on a parallel or vector machine. Some loops (e.g. recursive) are difficult to handle. Loop-level parallelism is still considered fine grain computation.

Procedure-level Parallelism
Medium-sized grain; usually less than 2000 instructions. Detection of parallelism is more difficult than with smaller grains; interprocedural dependence analysis is difficult and history-sensitive. Communication requirement less than instruction level SPMD (single procedure multiple data) is a special case Multitasking belongs to this level.

Subprogram-level Parallelism
Job step level; grain typically has thousands of instructions; medium- or coarse-grain level. Job steps can overlap across different jobs. Multiprograming conducted at this level No compilers available to exploit medium- or coarse-grain parallelism at present.

Job or Program-Level Parallelism
Corresponds to execution of essentially independent jobs or programs on a parallel computer. This is practical for a machine with a small number of powerful processors, but impractical for a machine with a large number of simple processors (since each processor would take too long to process a single job).

Communication Latency
Balancing granularity and latency can yield better performance. Various latencies attributed to machine architecture, technology, and communication patterns used. Latency imposes a limiting factor on machine scalability. Ex. Memory latency increases as memory capacity increases, limiting the amount of memory that can be used with a given tolerance for communication latency.

Interprocessor Communication Latency

- Needs to be minimized by system designer
- Affected by signal delays and communication patterns Ex. n communicating tasks may require n (n - 1)/2 communication links, and the complexity grows quadratically, effectively limiting the number of processors in the system.

Communication Patterns

- Determined by algorithms used and architectural support provided
- Patterns include permutations broadcast multicast conference
- Tradeoffs often exist between granularity of parallelism and communication demand.