Module III – Lecture IV

Parallel Algorithm Models

Parallel Algorithm Models

1. Data parallel Model

- Each task performs similar operations on different data – **Data Parallelism**
- The decomposition of the problem into tasks is usually based on data partitioning
- Static mapping is used
- Can be implemented in shared-address-space and message passing paradigms
- simplest algorithm models
- Example - Dense Matrix multiplication

Parallel Algorithm Models

2. Task graph Model

- Use task dependency graph to promote locality or reduce interactions
- Here, the interrelationships among the tasks are utilized to promote locality or to reduce interaction costs
- This model is typically employed to solve problems in which the amount of data associated with the tasks is large relative to the amount of computation associated with them
- This type of parallelism that is naturally expressed by ***independent tasks*** in a task-dependency graph is called **task parallelism**.
- Examples: Parallel quicksort, sparse matrix factorization, and many parallel algorithms derived via divide-and conquer decomposition

Parallel Algorithm Models

3. Work-pool / Task pool Model

It is characterized by a dynamic mapping of tasks onto processes for load balancing

- Pool of task
- Allocate tasks to any process
- No pre-mapping
- Task generated statically or dynamically
- Pointers to the tasks may be stored in a distributed data structure
- In the message-passing paradigm, the work pool model is typically used when the amount of data associated with tasks is relatively small compared to the computation associated with the tasks.
- Example: Parallelization of loops by chunk scheduling, Parallel tree search.

Parallel Algorithm Models

4. Master-slave model (Manager-Worker)

- One or more master processes generating tasks
- Allocate tasks to slave processes
- Allocation may be static or dynamic, if dynamic, a piece of task (whenever it is generated) is allocated to avoid process waiting time
- Work can be allotted in phases
- Manager ensures Worker process synchronization at each phase
- No desired pre-mapping of work to processes
- This approach can leads to ***hierarchical or multi-level manager-worker model***
- It is suitable to implement in shared-address-space or message-passing paradigms since the interaction is naturally two-way
- ***Challenge:*** avoiding bottleneck at master

Parallel Algorithm Models

5. Pipeline/producer-consumer Model

In the pipeline model, a stream of data is passed on through a succession of processes, each of which perform some task on it. This simultaneous execution of different programs on a data stream is called ***stream parallelism***

- With the exception of the process initiating the pipeline, the arrival of new data triggers the execution of a new task by a process in the pipeline.
- A pipeline is a chain of producers and consumers.
- Each process in the pipeline can be viewed as a consumer of a sequence of data items for the process preceding it in the pipeline and as a producer of data for the process following it in the pipeline.
- The pipeline model usually involves a static mapping of tasks onto processes.

Parallel Algorithm Models

5. Hybrid Model

A hybrid model may be composed either of multiple models applied hierarchically or multiple models applied sequentially to different phases of a parallel algorithm.

For instance, data may flow in a pipelined manner in a pattern guided by a task-dependency graph

Example: Parallel quicksort