Mid-term Preparation

Wei Wang

Basic Exam Information

- Time: Feb 21st , 2024
- Location: in-class
- Length: 75 min (designed to be 45min long)
- Open-book, open-computer, open-notes
- Math is simple, but you can bring a calculator, no cell phones allowed
- Only for Undergrad students

What materials do you need to prepare for the exam?

- The goal is to let you review and remember the necessary knowledge to master parallel programming
- ONLY class slides
- The answer to every question is in the slides
- Nothing outside class slides will be asked
  - Our slides cover most of the things you need to know about parallel programming
- I update my slides based on your feedback, so make sure you get the latest copies of the slides.

Types of Questions

- Short answers
- Fill in the blank
- True or false
- Problems
  - K-means
  - Concurrency
  - Decomposition (similar to the examples in slides)
  - Performance models

Things you need to know

- Introduction
  - What is Moore’s Law
  - What is Dennard Scaling?
    - Need to know the equation
  - What is the fundamental drive to adopt multi-core processors?

Things you need to know cont'd

- K-Means Clusters
  - Be able to manually do one iteration of K-Means algorithm
- Parallel Algorithm Design
  - Basic Concepts:
    - Task dependency graph
    - Degree of concurrency, granularity, critical path, limits on parallel performance (remember the quiz?)
    - Tasks, threads, processors and mapping
    - Metrics: speedup and parallel efficiency. Be able to compute speed up and efficiency

Things you need to know cont’d

- Parallel Algorithm Design
  - Characteristics of Tasks and Interactions
    - Characteristics: statically/dynamically generated, data size (data transport vs computation), computation size (uniform/non-uniform)
    - Interactions: static vs dynamic, regular vs irregular, read-only vs read-write, one-sided vs two-sided

Things you need to know cont’d

- Parallel Algorithm Design
  - Decomposition Techniques
    - Recursive
    - Data
    - Exploratory
    - Speculative
    - Hybrid
    - Decomposition techniques and their typical types of problems (i.e., speculative for simulation problems)

Things you need to know cont’d

- Parallel Algorithm Design
  - Mapping Techniques
    - The overhead of (bad) mappings
    - Static
      - What tasks can use static mapping?
    - Dynamic
      - What tasks should use dynamic mapping?
      - Centralized
        - Common techniques
        - disadvantages
      - Distributed
        - Common techniques
        - disadvantages

Things you need to know cont’d

- Parallel Algorithm Design
  - Minimizing Communication/Interaction Techniques
    - Remember the principles and techniques
  - Hardware considerations for task mapping
    - Remember the hardware constraints
- Shared-memory architecture
  - Shared-memory architecture vs Distribute-memory architecture
  - The basic synchronization techniques

Things you need to know cont’d

- Shared-memory architecture
  - Shared-memory architecture vs Distribute-memory architecture
  - The basic synchronization techniques
    - Mutex
    - Barrier
- OpenMP
  - Constructs and clauses
  - Be able to read and write simple OpenMP program

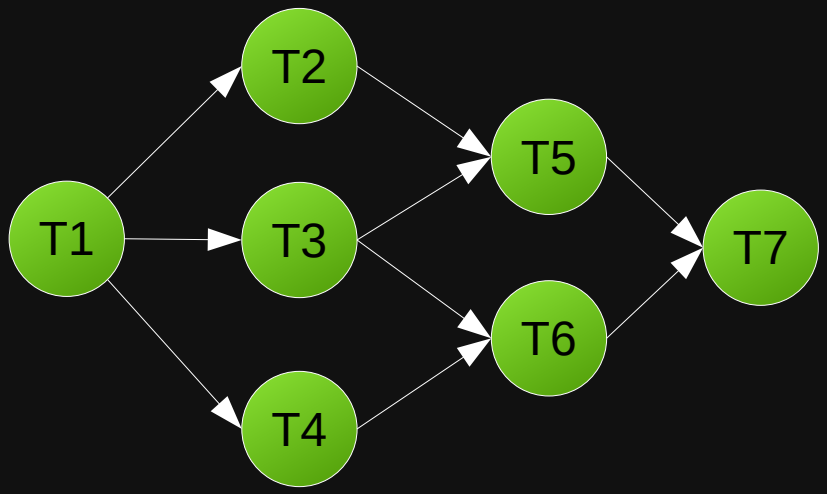
Parallel Application Application Performance

- Theoretical performance models
  - Amdahl's Law and Karp–Flatt metric
  - Gustafson's Law
  - Speedup model with communication cost
  - For each model,
    - remember its equation
    - Understand the intuition behind the model
  - Be able to tell the difference between Amdahl's Law, Gustafson's Law, and Speedup model w/ communication cost.

Example Problem 1

Given the following Task Dependency Graph

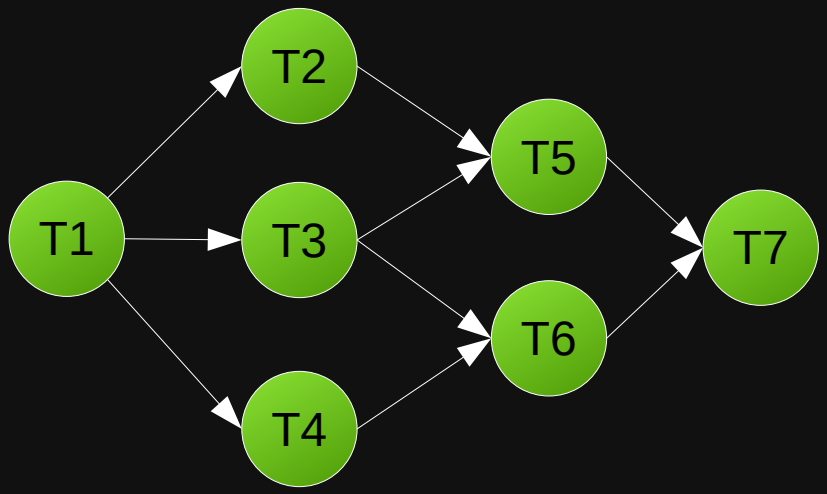
- What is the maximum concurrency?
- What is the average concurrency?
- What is the length of the critical path (# of tasks on critical path)?



Example Problem 1 cont'd

Given the following Task Dependency Graph

- What is the maximum concurrency? A: 3
- What is the average concurrency? A: 1.75
- What is the length of the critical path (# of tasks on critical path)? A: 4



Example Problem 2

- Given the following data points:
  (1,1,) (2,2) (3,3) (-1, -1), (-2, -2), (-3,-3)
- Given the following centers:
  (4,6), (-6,-4)
- Please partition the data points into the two clusters based on the above centers

Example Problem 2 cont'd

- Answer:
  - Cluster at (4,6) has points: (1,1), (2,2), (3,3)
  - Cluster at (-6,-4): (-1,-1), (-2,-2), (-3,-3)
- Note:
  - The actual problem is more difficult to compute than this example
  - Although a calculator is unnecessary, you are encouraged to bring a calculator. NO CELL PHONES!

Example Problem 3: OpenMP Programming

- The following code tries to produce some data using multiple threads. After producing the data, each thread should update a global counter to indicate it has finished. However, there is a bug in the following code preventing from executing correctly. Please fix the following code.

```
int counter = 0;

#pragma omp parallel
{
    produce_data();

    counter++;

}
printf("%d copies of data generated\n", counter);
```

Example Problem 3: OpenMP Programming Extended

- Solution:

```
int counter = 0;

#pragma omp parallel
{
    produce_data();

    #pragma omp critical
    {counter++;}

}
printf("%d copies of data generated\n", counter);
```

Amdahl's Law

- Consider a parallel application with 60% parallelizable part, whose execution time can be improved by 6 times, what is the overall speed up of this application?
- Answer:

$$Spd = \frac{1}{40\% + \frac{60\%}{6}} = \frac{1}{40\% + 10\%} = 2$$

Parallel Efficiency

- Given the above application, if it is executed on a 6-core processor, what its parallel efficiency?
- Answer

$$Eff = \frac{T_1}{p * T_p} = \frac{1}{6 * 50\%} = 0.33$$

Gustafson's Law

- Consider a parallel application with 60% parallelizable part, whose execution time can be improved by 6 times. Additionally, we can also increase the parallelizable part by 6 times, what is the overall speed up of this application?
- Answer

$$Spd = 1 - p\% + s \times p\% = 40\% + 6 \times 60\% = 4$$

Speedup w. Communication

- Consider a parallel application with 60% parallelizable part, whose execution time can be improved by 6 times. Additionally, the parallelization add 10% extra commutation time. what is the overall speed up of this application?
- Answer:

$$Spd = \frac{1}{40\% + \frac{60\%}{6} + 10\%} = \frac{1}{40\% + 10\% + 10\%} = 1.67$$

Example Conceptual Questions:

- Please list the three characteristics of a task:
  - Answer: generation, work size, data size
- True or false: Dennard scaling is the fundamental drive of the adoption of multi-core processor
  - Answer: True