

Schedule 2/1/10

- **Review from last week**
 - **Circuits and trees**
- **Planning and scheduling**
 - **Digraphs**
 - **List processing**
 - **Bin packing – saved for next week**

Review

- **Graphs**
 - **Circuits**
 - Euler – Chinese postman problem
 - Hamiltonian – Traveling salesman problem
 - Brute force, nearest neighbor, sorted edges
 - **Trees**
 - Spanning tree
 - Kruskal's algorithm

A Last Graph

- **Digraphs**
 - **Graphs with a direction**
 - **Critical path – the longest path**
 - **Applications**
 - **Scheduling**

Planning and Scheduling

- **The machine scheduling problem**
 - **Make certain assumptions**
 - **Achieve certain goals**
 - **Examples: Digraphs**
 - **Scheduling emergency room patients**
 - **Scheduling the building of a house**

Assumptions

- 1. If a processor starts work on a task, the work on that task will continue without interruption until the task is completed**
- 2. No processor stays voluntarily idle. In other words, if there is a processor available and a task to be worked on, that processor will do that task.**
- 3. The requirements for ordering the tasks are given by an order-requirements digraph.**
- 4. The tasks are arranged in a priority list (order of importance) that is independent of the order requirements.**

Possible Goals

- 1. Minimizing the completing time of the job**
- 2. Minimizing the total time that processors are idle**
- 3. Finding the minimum number of processors necessary to finish the job by a specified time**

List Processing Algorithm

- **Definitions**
 - A task is *ready* if all its predecessors have been completed
- **Scan the list from left to right**
 - Skip tasks that are not ready
 - If no task can be assigned, keep processors idle until a task is ready
 - After a task is assigned, continue scanning the list for unassigned tasks

Factors that Affect Scheduling

- 1. The times to carry out the tasks**
- 2. Number of processors**
- 3. Order-requirements digraph**
- 4. Ordering of the tasks on the priority list**

Strategies to Reduce Time

- 1. Reduce times for individual tasks**
- 2. Use more processors**
- 3. Loosen constraints on the priority list**