IE 495 Lecture 13

October 12, 2000
Reading for This Lecture

- **Primary**
  - Horowitz and Sahni, Chapter 4
Course Recap
Our View of the World

Computer Systems

Hardware
- Memory
- Processors
- Interconnections
- Storage

Software
- Operating Environment
- Shared Libraries
- Application Code

OS
- Programming Language
- Compiler
- Data Structures
- Algorithms
- Implementation
Classifying and Modeling Architectures

Classifying Architectures

Flynn's Taxonomy

- SISD
- SIMD
- MISD
- MIMD

Serial Computing → RAM Model

Parallel Computing → PRAM Model(s)

Modeling Architectures

Shared Memory

- Shared Memory PRAM Model

Distributed Memory

- Distributed PRAM Model

Interconnection Networks

- CRCW
- CREW
- ERCW
- EREW
Analyzing Architectures and Algorithms

Lecture 2

Analyzing Architectures

Interconnection Networks

Performance Measures

Graph Properties
  Degree
  Bisection Width
  Communication Diameter
  Connectivity Matrix
  Adjacency Matrix

Time to Perform Operations
  Semigroup operations
  Sorting operations

Lecture 3 and 4

Analyzing Algorithms

Asymptotic Analysis

Modeling Assumptions
Classifying Algorithms

Orders of Magnitude
Polynomial/Exponential
Time Complexity
Space Complexity

Induction and Recursion

Master Theorem
Design, and Analysis of Parallel Algorithms

- Scalability
- Performance Measures
- Design Issues
- Implementation
  - OpenMP
  - PVM
Basic Data Structures

- Stacks, Lists, and Queues
- Heaps
- Hashing
- Graphs
- Analysis
- Implementation
Second Half of the Course

- Greedy Algorithms and Matroids
- Graph Algorithms
- Search Algorithms/Divide-and-Conquer
  - Branch and Bound
  - IP
- Matrix Algorithms/Numerical Algorithms
  - Numerical Analysis
Greedy Algorithms
Basic Algorithm

A is an array of the inputs

S = Ø;

for (i = 0; i < n; i++){
    x = SELECT(A);
    if (feasible(UNION(S, x))){
        S = UNION(S, x);
    }
}
Basic Data Structures

- SELECT
- UNION
Fractional Knapsack Problem

- We are given $n$ objects.
- Each object has a weight $w_i$ and a profit $p_i$.
- We also have a knapsack with capacity $M$.
- **Objective:** Fill the knapsack as profitably as possible.
- We allow fractional objects.
- Algorithm
- Analysis
Job Sequencing with Deadlines

- We are given a set of $n$ jobs.
- Each job takes one unit of time.
- Each job has a deadline $d_i$ and a profit $p_i$.
- **Objective**: A feasible schedule that maximizes profit.
- Algorithm
- Analysis
Spanning Trees

• We are given a graph $G = (V, E)$.

• A spanning tree of $E$ is a maximal acyclic subgraph $(V, T)$ of $G$.

• A spanning tree always has $|V| - 1$ edges (why?).
Minimum Spanning Tree

- We associate a weight $w_e$ with each edge $e$.
- **Objective**: Find a spanning tree of minimum weight.
- Applications
Prim's Algorithm

* S is the set of nodes in the tree

S = \{0\}
for (i = 0; i < n; i++){
    SELECT i \notin S nearest to S;
    S = UNION(S, i);
}
Analysis of Prim's Algorithm

- Correctness
- Optimality
- Implementation
- Complexity
Kruskal's Algorithm

$T$ is the set of edges in the tree

$T = \emptyset$

for ($i = 0; i < m; i++$)
  SELECT the cheapest edge $e$
  if (feasible(UNION($T$, $e$))
    UNION($T$, $e$);
Analysis of Kruskal's Algorithm

• Correctness

• Optimality

• Implementation

• Complexity