CSE 421
Algorithms

Richard Anderson

Lecture 7

Greedy Algorithms

Greedy Algorithms

- Solve problems with the simplest possible algorithm
- The hard part: showing that something simple actually works
- Pseudo-definition
  - An algorithm is Greedy if it builds its solution by adding elements one at a time using a simple rule

Scheduling Theory

- Tasks
  - Processing requirements, release times, deadlines
- Processors
- Precedence constraints
- Objective function
  - Jobs scheduled, lateness, total execution time

Interval Scheduling

- Tasks occur at fixed time
- Single processor
- Maximize number of tasks completed

- Tasks {1, 2, . . . N}
- Start and finish times, s(i), f(i)

Simple heuristics

Schedule earliest available task

Instructor note counter examples

Schedule shortest available task

Schedule task with fewest conflicts

Schedule available task with the earliest deadline

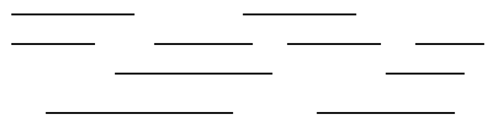
- Let A be the set of tasks computed by this algorithm, and let O be an optimal set of tasks. We want to show that |A| = |O|
  - Let A = $\{i_1, \ldots, i_k\}$, O = $\{j_1, \ldots, j_m\}$, both in increasing order of finish times

Correctness Proof

- A always stays ahead of O, $f(i_r) <= f(j_r)$
- Induction argument
  - $f(i_1) <= f(j_1)$
  - If $f(i_{r-1}) <= f(j_{r-1})$ then $f(i_r) <= f(j_r)$

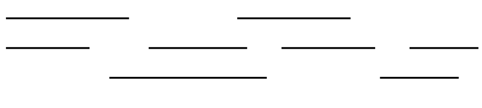
Scheduling all intervals

- Minimize number of processors to schedule all intervals



Lower bound

- In any instance of the interval partitioning problem, the number of processors is at least the depth of the set of intervals



Algorithm

- Sort by start times
- Suppose maximum depth is d, create d slots
- Schedule items in increasing order, assign each item to an open slot

- Correctness proof: When we reach an item, we always have an open slot

Scheduling tasks

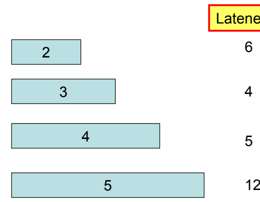
- Each task has a length t_i and a deadline d_i
- All tasks are available at the start
- One task may be worked on at a time
- All tasks must be completed

- Goal minimize maximum lateness
  - Lateness = $f_i – d_i$ if $f_i >= d_i$

Example

Show the schedule 2, 3, 4, 5 first and compute lateness

Lateness



Greedy Algorithm

- Earliest deadline first
- Order jobs by deadline

- This algorithm is optimal

This result may be surprising, since it ignores the job lengths

Analysis

- Suppose the jobs are ordered by deadlines, $d_1 <= d_2 <= . . . <= d_n$
- A schedule has an *inversion* if job j is scheduled before i where j > i

- The schedule A computed by the greedy algorithm has no inversions.
- Let O be the optimal schedule, we want to show that A has the same maximum lateness as O

Proof

- Lemma: There is an optimal schedule with no idle time.

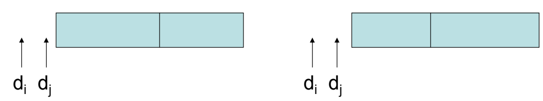
- Lemma: There is an optimal schedule with no inversions and no idle time.

- Let O be an optimal schedule k inversions, we construct a new optimal schedule with k-1 inversions

If there is an inversion, there is an inversion of adjacent jobs

Interchange argument

- Suppose there is a pair of jobs i and j, with i < j, and j scheduled immediately before i. Interchanging i and j does not increase the maximum lateness. Recall, $d_i <= d_j$



Summary

- Simple algorithms for scheduling problems
- Correctness proofs
  - Method 1: Identify an invariant and establish by induction that it holds
  - Method 2: Show that the algorithm's solution is as good as an optimal one by converting the optimal solution to the algorithm's solution while preserving value