Greedy Algorithms

Jonathan Backer
backer@cs.ubc.ca

Department of Computer Science
University of British Columbia

June 24, 2007

Introduction

Reading:

1. CLRS: "Greedy Algorithms" 16.1-16.2
2. GT: "The Greedy Method" 5.1

We have already discussed several algorithm design techniques:

- divide and conquer (sorting and integer multiplication)
- prune and search (select or randomized select)
- (now) greedy algorithms

Definition

A greedy algorithm solves an optimization problem:

- makes a sequence of choices
- picks the choice that seems "best" without explicit consideration for past or future choices

Greedy algorithm correctness

What property guarantees greedy solutions work?

Greedy choice property

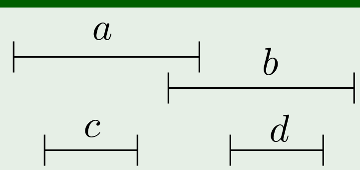
Some optimal solution can be obtained by combining

- a greedy choice with
- an optimal solution to remaining subproblem

Problem: Activity selection

Find a largest subset of non-overlapping intervals. Variants are greatest weight subset or largest subset taking the least time.

Example



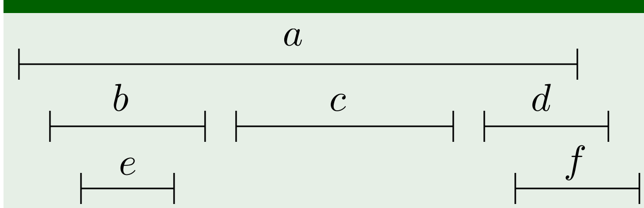
$\{a, d\}$ and $\{b, c\}$ are the only valid solutions.

Activity selection

Constuct a solution by

- selecting intervals one at a time
- but never chosing an interval that overlaps a previous selection

Example



Optimal is $\{b, c, d\}$, $\{b, c, f\}$, $\{e, c, d\}$, or $\{e, c, f\}$

What order to consider the activities?

- Sorted alphabetically? No, selects $\{a\}$.
- Sorted by left endpoints? No, selects $\{a\}$.
- Sorted by right endpoints? Maybe, selects $\{e, c, d\}$.

Sorting by right endpoints

- We choose the activity that ends the earliest.
- We leaves as much of the rest of the day available as possible.

```
Algorithm ActivitySelect(A)
   S ← ∅
   sort A by increasing right endpoints
   for j ← 0 to A.length-1
      if A[j].left ≥ maxRightEndPoint(S)
         S ← S ∪ A[j]
   return S
```

- The runtime is $\Theta(n \log n)$ because we can make each comparison take $\Theta(1)$ time.
- Why does it work?

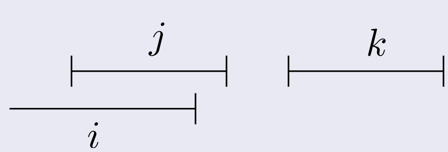
Greedy choice property

Lemma

If activity i has the smallest right endpoint, some optimal activity selection includes i.

Proof.

Consider an optimal activity selection S. If $i \in S$, then S is the desired selection. Otherwise, let $j \in S$ have the smallest right endpoint. Every $k \in S$ other than j must lie to right of j because j and k do not overlap and j's right endpoint is the smallest. So i and k do not overlap because i's right endpoint is at least as small at j's. Thus we can swap j for i to get an optimal selection.



Activity selection correctness

Theorem

`ActivitySelect` *returns an optimal activity selection.*

Proof.

Let i be the activity of A with the earliest right endpoint. By the previous Lemma, some optimal selection T of A contains i.

Let S be the selection S chosen by our algorithm. To prove optimality, we must show that $|S| \geq |T|$. We do this by induction on $|A|$.

Base case: $|A| \leq 1$. Trivially, $S = T = A$.

Activity selection correctness (cont'd)

Proof. (cont'd)

Induction step: Suppose our selection algorithm works for all sets of activities with less than $|A|$ activities (strong induction).

Let A' be the activities of A that do not conflict with i. Let S' be our algorithm's selection for A'.

By our greedy criteria, $S = \{i\} \cup S'$. By our inductive hypothesis, S' is an optimal selection of A'.

Let $T' = T \setminus \{i\}$. Then $T' \subseteq A'$. Therefore, $|T'| \leq |S'|$ by optimality of S'. Hence, $|T| \leq |S|$. Thus, S is an optimal selection of A.