CSCC73: Algorithm Design and Analysis Lecture Notes

Joshua Concon

University of Toronto Scarborough – Fall 2017

Pre-reqs are CSCB63 and STAB52. Instructor is Dr. Vassos Hadzilacos. He projects well so you can basically sit anywhere. If you find any problems in these notes, feel free to contact me at conconjoshua@gmail.com.

Contents

1 Wednesday, September 6, 2017

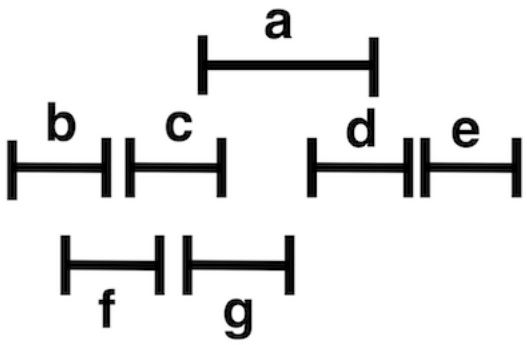
1.1 Greedy Algorithms

All Greedy Algorithms are Optimization problems: given input, compute output that

1. satisfies constraints
2. optimizes (min or max) certain criteria

Solutions that satisfies constraints are called **feasible**.

1.1.1 Interval Scheduling (KT 4.1)



Input: Set of jobs $1, 2, ..., n$ where job i has start time $s(i)$ and finish time $f(i) > s(i)$

> jobs i, j conflict if each starts before the other finishes. i.e. if $s(i) < f(j)$ and $s(j) < f(i)$

Feasible set of jobs: A set where no two jobs conflict.

Output: A max cardinality feasible set (a max non-conflicting set of jobs) ('a set' rather than 'the set' as there may be more than 1 optimal sets.)

e.g. $\{b, c, d, e\}, \{b, g, d, e\}$ (from figure 1)

Greedy "Schema":

```
sort jobs in some order
A := null
for each job i in sorted order do
        if i conflicts with no job in A then
                A := union of A and {i}
return A
```

We will make edits to this schema as we learn more about what sort order provides us with the most optimal set.

1.1.2 Possible sort orders

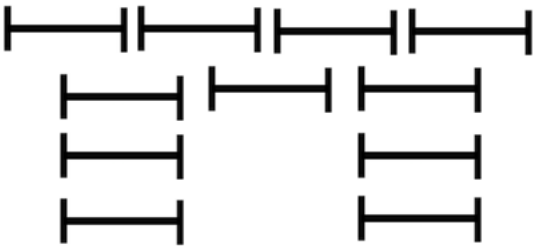
1. **sort by increasing start time**
   counterexample:

   As shown in the picture, the most optimal answer are the short intervals that come after the long interval has started, but the long interval is chosen first conflicting with the rest of the intervals.

2. **sort by increasing duration length**
   counterexample:

The optimal answer in this picture is the two long intervals, but since there shorter interval is chosen and it conflicts with both of the long intervals, this sort does not work.

3. **sort by increasing number of conflicts**
counterexample:



In this case, the most optimal answer is all 4 of the top intervals. However, since it starts with the interval with the least amount of conflicts, it chooses an interval that conflicts with 2 of the 4 top intervals, which is the middle interval.

4. **sort by increasing finish time**
this increments A using the smallest amount of time as possible, and this is also the correct sort for this algorithm.

Greedy "Schema" Revised:

```
sort jobs in increasing finish time
A := null
F := -infinity
for each job i in sorted order do
        if s(i) >= F then
                F := f(i)
                A := union of A and {i}
return A
```

Running Time: $O(nlogn) + O(n) + O(1) = O(nlogn)$

1.1.3 Proof of Correctness (optimality)

Let $j_1, j_2, ..., j_k$ be jobs added to A in order considered

Claim 1: A is feasible, proof trivial (just use sort by increasing finish time in the scheme, nothing in A should conflict by the algorithm).

Let $j_1^*, j_2^*, ..., j_m^*$ be jobs in some optimal A^* (in left to right order)

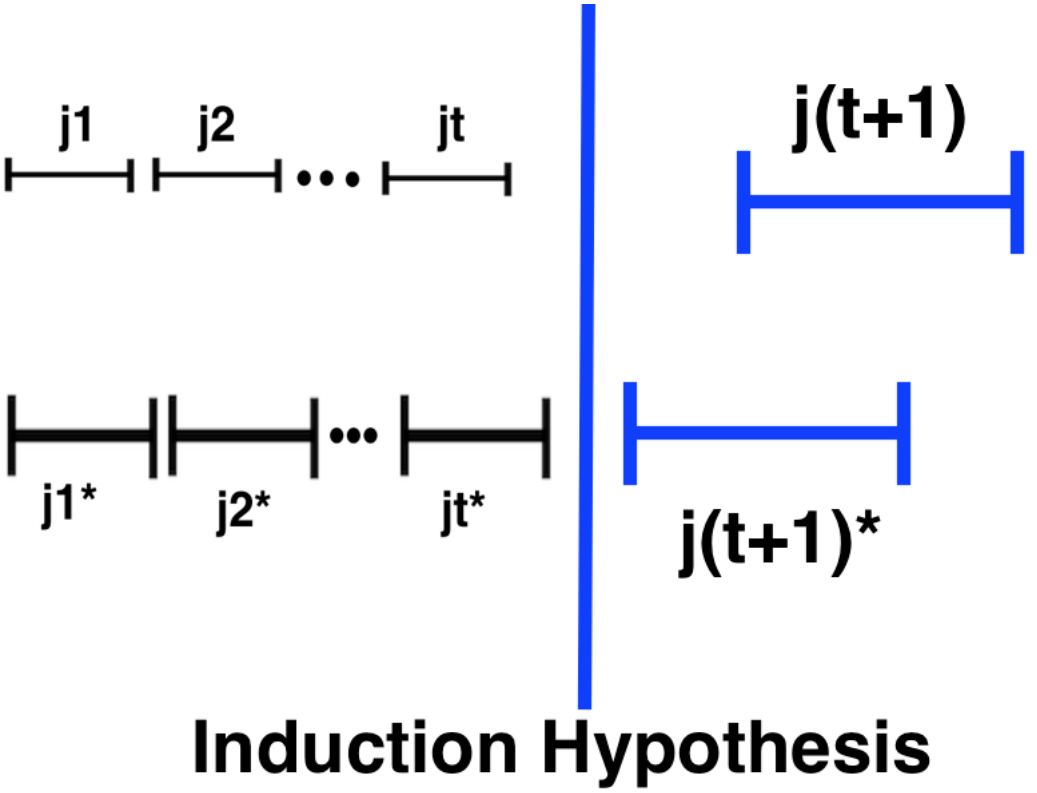
Claim 2: $f(j_t) \leq f(j_t^*), \forall t, 1 \leq t \leq k$ (greedy algorithm stays ahead)

Proof. (proof of Claim 2)
Use induction.

Basis: $t = 1$, algorithm adds interval with earliest finish time to A, so this holds.

Induction Step: For contradiction, assume $f(j_{t+1}) > f(j_{t+1}^*)$. This is impossible because $f(j_t) \leq f(j_t^*)$ and the Induction Hypothesis Algorithm has not considered j_{t+1}^* yet and will consider j_{t+1}^* before j_{t+1} and will add j_{t+1}^* to A instead of j_{t+1}.

Induction Hypothesis: $f(j_t) \leq f(j_t^*)$



Assume for contradiction: $f(j_{t+1}) > f(j^*_{t+1})$.

We have...

$$\begin{aligned} f(j_t) &\leq f(j^*_t) \quad \text{by Induction Hypothesis} \\ &\leq s(j^*_{t+1}) \quad \text{Because } A^* \text{ is feasible and jobs are labelled left to right} \\ &< f(j^*_{t+1}) \quad \text{Finish time is strictly greater than start time} \end{aligned}$$

Immediately after the algorithm adds job j_t to A, job j^*_{t+1}:

- has not been considered ($f(j_t) < f(j^*_{t+1})$
- j^*_{t+1} does not conflict with jobs in A as $f(j_t) \leq f(j^*_t) \leq s(j^*_{t+1})$
- j^*_{t+1} has priority over j_{t+1} (by assumption, $f(j_{t+1}) > f(j^*_{t+1})$)

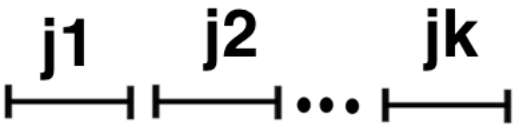
Therefore j_{t+1} is not the next job added to A by the algorithm, therefore there is a contradiction. So Claim 2 must be true.

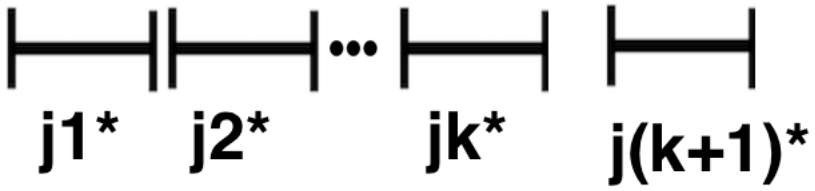
■

Claim 3: $k = m$

Proof. (Proof of Claim 3)

Clearly $k \leq m$, since m is optimal.
Assume for contradiction that $k < m$





But then j^*_{k+1} should also be in A, but it's not in A. Therefore it doesn't exist.

Therefore $k = m$ ■

Alternative Approach ("promising set"):
For each iteration i in a Greedy Algorithm, there exists an optimal set A^* such that $A_i \subseteq A^*$

Generalization of Interval Scheduling:
Interval Scheduling can be generalized to cover different problems, such as

Weighted Intervals, finding the minimum amount of concurrent machine to perform all intervals.