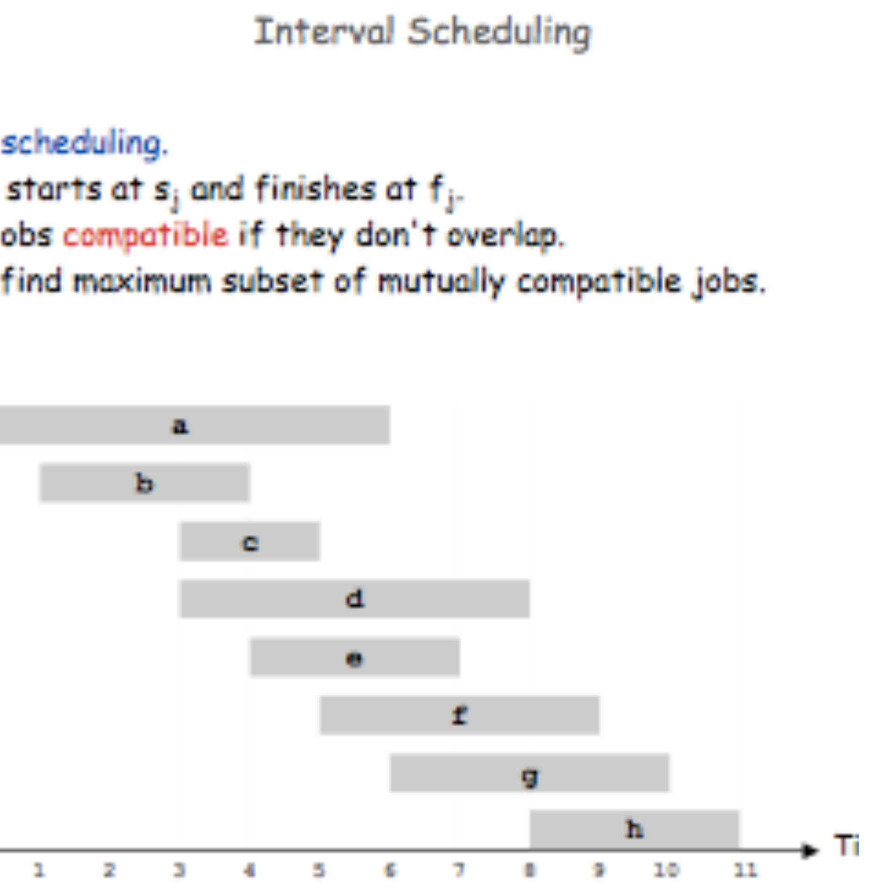
CSC373: Lecture 3

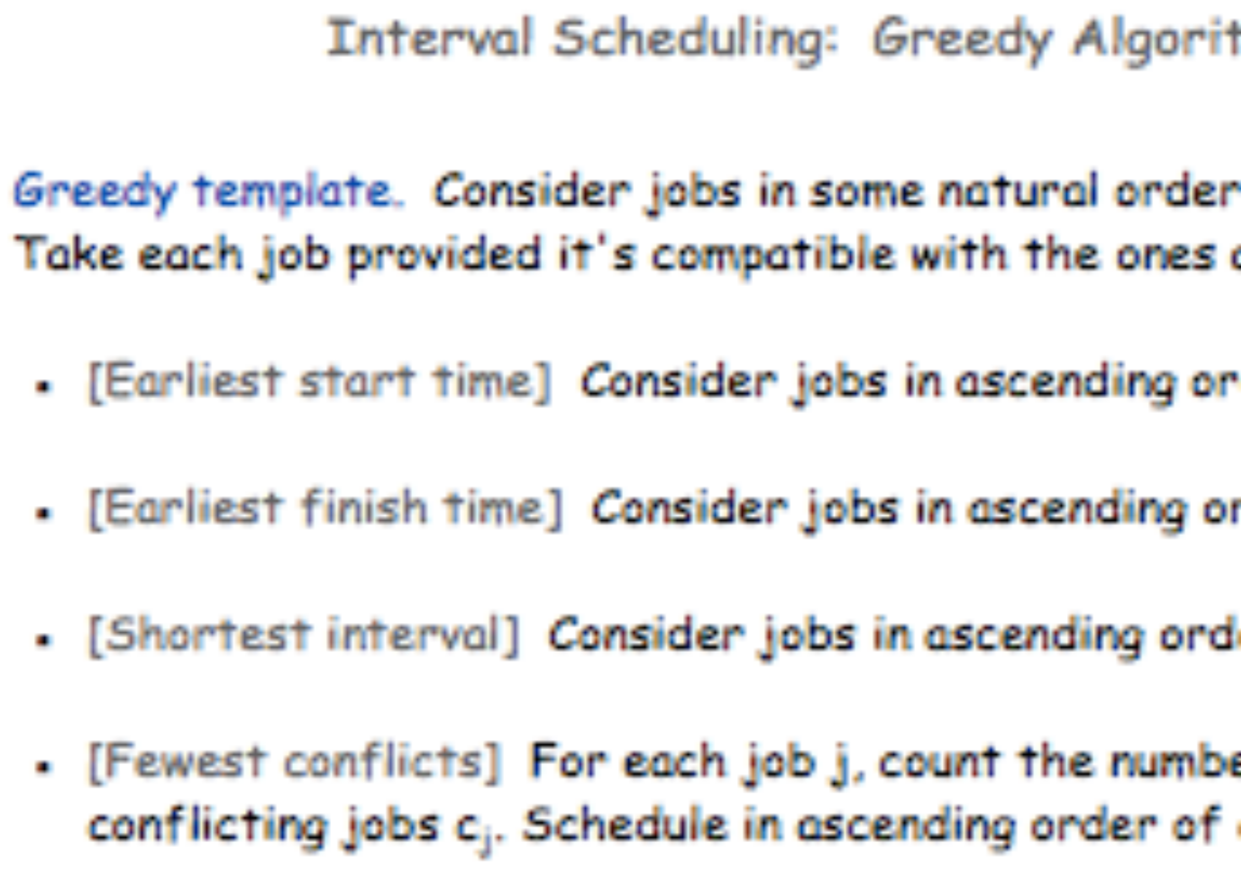
Continuation of Greedy Algorithm Discussion

The EFT Greedy Algorithm



Interval Scheduling

scheduling.
starts at s_j and finishes at f_j.
obs compatible if they don't overlap.
find maximum subset of mutually compatible jobs.



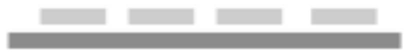
Interval Scheduling: Greedy Algorit

Greedy template. Consider jobs in some natural order
Take each job provided it's compatible with the ones

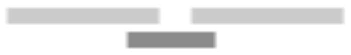
- [Earliest start time] Consider jobs in ascending or
- [Earliest finish time] Consider jobs in ascending or
- [Shortest interval] Consider jobs in ascending ord
- [Fewest conflicts] For each job j, count the numbe
conflicting jobs c_j. Schedule in ascending order of

Interval Scheduling: Greedy Algorithms

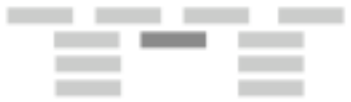
Greedy template. Consider jobs in some natural order.
Take each job provided it's compatible with the ones already taken.



counterexample for earliest start time



counterexample for shortest interval



counterexample for fewest conflicts

Comments on the optimality of *EFT*

- Last class Yuli Ye gave a proof that the *EFT* greedy algorithm for the interval selection problem (ISP). The proof given was to show that the partial solution *S(i)* at the end of the *i* th iteration is ***promising*** in that it can be extended to an optimal solution (using intervals not yet considered).
- This is not the only possible proof of this result. But before giving another type of proof, you might rightfully ask “why bother proving this”?

Why bother proving facts about a particular algorithm?

- As we have seen, other reasonable (greedy) algorithms for ISP fail to obtain an optimal solution (for all input instances). So while in hindsight we can motivate *EFT* and convince ourselves that *EFT* is optimal, we need a convincing argument (i.e. a proof at some level of being convincing) that *EFT* is indeed optimal.

- In addition, proofs give us insight into the limitations of an algorithm and also what is and is not necessary to establish the desired properties. For example, the proof does not rely on the exact manner in which we break "ties" (between intervals with the same finishing time). Hence while an algorithm needs exact specification, any tie breaking rule will work!

- Proofs also can yield additional facts as we will see in the case of interval colouring and MST problems

Charging arguments

- A common method for proving optimality and approximation results for a optimization algorithm *ALG* is by a ***charging argument***. For a <u>profit maximization</u> problem we want to charge the profit of an arbitrary solution (and hence that of an optimal solution *OPT*) to the profit of your *ALG.* The goal is to argue that not too much profit from *OPT* gets charged to *ALG.*

- For a <u>cost minimization</u> problem, we want to charge the cost of *ALG* to an *OPT* solution and argue that not too much cost from *ALG is charged to OPT.*

Charging argument for EFT (as discussed in the tutorial sections)

- For the ISP problem, the profit of an algorithm is simply the number of intervals selected. We will write $|ALG(\mathcal{I})|$ (resp. $|OPT(\mathcal{I}))$ for the profit of algorithm *ALG* (resp. an optimum solution) on input set $\mathcal{I}$. Then to show optimality of EFT for ISP, it suffices to show that there is a 1-1 function *h* mapping $OPT(\mathcal{I})$ into $EFT(\mathcal{I})$. (Since OPT denotes an optimum solution the mapping must be onto.)

Charging argument to obtain approximation bound

- As stated in the first class, I like to integrate some results about approximation algorithms as we proceed rather than treat approximation algorithms as a separate topic.
- We can easily adapt the EFT algorithm so as to apply to the JISP problem. In the JISP problem we extend the meaning of two intervals being compatible if they do not intersect and if they do not belong to the same job class.
- Claim: For the JISP problem we can show that the same *h is a 2-1* function mapping *OPT*($\mathcal{I}$) into *EFT*($\mathcal{I}$).

The *m*-ISP problem

- The *m* “machine” interval scheduling problem schedules a set of intervals on *m* machines so that intervals assigned to the same machine do not intersect.
- Consider the following two extensions of the one machine EFT algorithm:

First fit vs Best fit EFT

```
1. First fit EFT
   Sort intervals so that $f_1 \leq f_2 \ldots \leq f_n$
   For $i : 1..n$
       Let $k = min_\ell : J(i)$ does not intersect intervals on machine $\ell; 0$ if no such $\ell$
       $\sigma(i) := k$ % $\sigma(i)$ specifies if and on which machine interval $J(i)$ is scheduled
   End For

2. Best fit EFT
   Sort intervals so that $f_1 \leq f_2 \ldots \leq f_n$
   For $k : 1..m$
       $e_k := -0$ % $e_k$ specifies the latest completion for intervals on machine $k$
   End For
   For $i : 1..n$
       Let $k = argmin_\ell : s_i - e_\ell > 0$ or $k = 0$ if no such $\ell$
       $\sigma(i) := k$ % $\sigma(i)$ specifies if and on which machine interval $J(i)$ is scheduled
       $e_k := f_i$
   End For
```

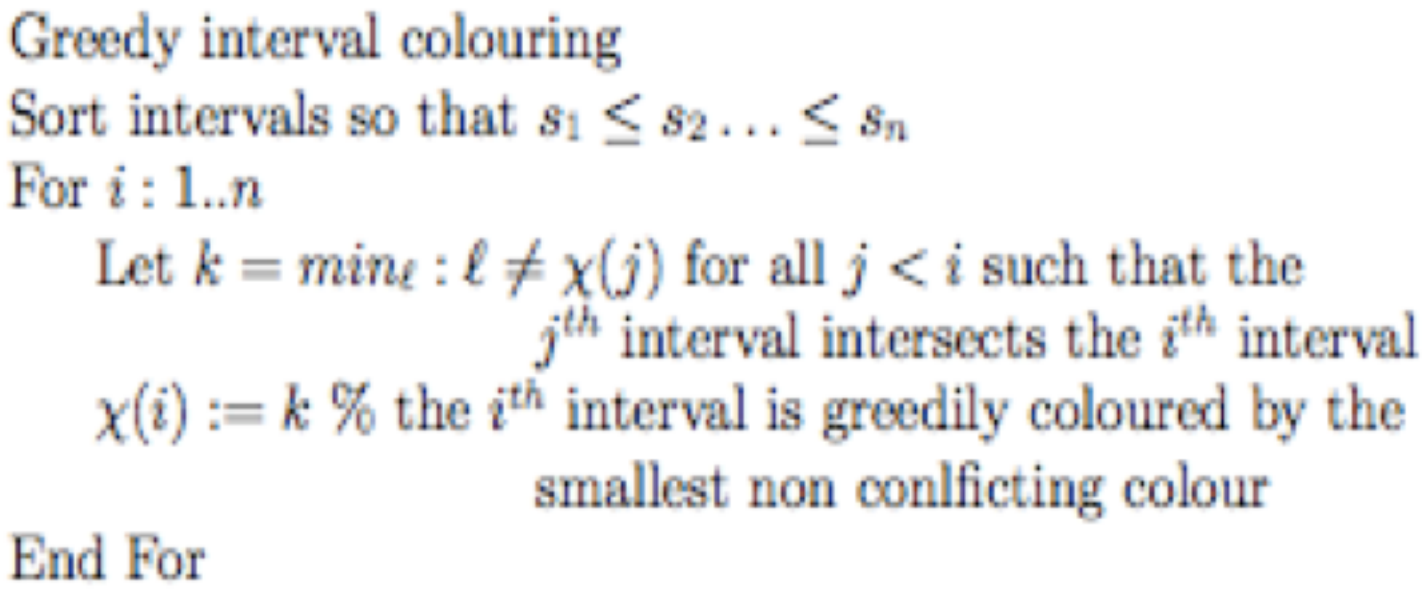
Interval colouring

- We will now consider a minimization problem; namely given a set of intervals, we want to colour all intervals so that intervals given the same colour do not intersect and the goal is to try to minimize the number of colours used.
- We could simply apply the *m*-machine ISP for increasing *m* until we found the smallest *m* that is sufficient. (Note: This is a simple example of a ***polynomial time reduction*** which is an essential concept when we study NP-completeness.)

Greedy interval colouring

- Consider the *EST* (earliest starting time) for interval colouring. Namely, having sorted the intervals by non decreasing starting times, we assign each interval the smallest numbered colour that is feasible given the intervals already coloured. (Recall that *EST* is a terrible algorithm for ISP.) Note that this algorithm is "equivalent" to *LFT* (latest finishing time first).
- Theorem: *EST* is optimal for interval colouring

Greedy interval colouring



```
Greedy interval colouring
Sort intervals so that $s_1 \leq s_2 \ldots \leq s_n$
For $i : 1..n$
    Let $k = min_\ell : \ell \neq \chi(j)$ for all $j < i$ such that the
                        $j^{th}$ interval intersects the $i^{th}$ interval
    $\chi(i) := k$ % the $i^{th}$ interval is greedily coloured by the
                        smallest non conlficting colour
End For
```

Proof of optimality (sketch)

- The proof technique we will use here is also one often used for proving approximations.
- The idea is to find some bound (or bounds) that *any solution* must satisfy and then relate that to the algorithms solution.
- In this case, consider the maximum number of intervals in the input set that intersect at any given point. The number of colours must be at least this large.
- It remains to show that the greedy algorithm will never use more than this number of colours.