Interval Scheduling

- Job $j$ starts at $s_j$ and finishes at $f_j$.
- Two jobs compatible if they don’t overlap.
- Goal: find maximum subset of mutually compatible jobs.

Interval Scheduling: Greedy Solution

- **Greedy template.** Consider jobs in some order. Take each job provided it’s compatible with the ones already taken.
- **Idea 1: Earliest start time.** Consider jobs in ascending order of start time $s_j$. 
Interval Scheduling: Greedy Solution

- **Greedy template.** Consider jobs in some order. Take each job provided it's compatible with the ones already taken.
- **Idea 1:** Earliest start time. Consider jobs in ascending order of start time $s_j$.
- **Idea 2:** Shortest interval. Consider jobs in ascending order of interval length $f_j - s_j$.

**Time**

0  1  2  3  4  5  6  7  8  9  10  11

**Jobs**

a, g
b, c, h
f, g, h, e, a, b, c, d

3-2

4-1

4-2
Interval Scheduling: Greedy Solution

- **Greedy template.** Consider jobs in some order. Take each job provided it's compatible with the ones already taken.
- **Idea 3:** Fewest conflicts. For each job, count the number of conflicting jobs \( c_j \). Schedule in ascending order of conflicts \( c_j \).
- **Idea 4:** Earliest finish time. Consider jobs in ascending order of finish time \( f_j \).
Hallmark of a Greedy Algorithm

- Sort data according to some criteria
- Consider each piece of data in sorted order and make a local decision
- Result is globally optimal (if this problem is amenable to a greedy solution!) 
- Complexity is generally no better than $O(n \log n)$ due to the sort
- Important to prove that the solution is optimal