

The Greedy Scheduling Algorithm

Input: Tasks $\{t_1, t_2, \dots, t_n\}$, each with starting time s_i and finish time f_i

Solution: Valid scheduling on m machines. Precisely, a function $g : \{1, 2, \dots, n\} \rightarrow \{1, 2, \dots, m\}$ such that, for any $1 \leq i < j \leq n$, if $g(i) = g(j)$, then tasks t_i and t_j do not *overlap* (that is, either $f_i < s_j$ or $f_j < s_i$).

Measure: Minimize m

```
1 Sort the task according to starting time. At this moment, if  $i < j$ , then  $s_i \leq s_j$ .
2  $m \leftarrow 0$ 
3 for  $i \leftarrow 1$  to  $n$ 
4   if (there is a machine  $j$  where task  $t_i$  fits)
5      $g(i) \leftarrow j$  // put  $t_i$  on machine  $j$ 
6   else
7      $m \leftarrow m + 1$ 
8      $g(i) \leftarrow m$  // put  $t_i$  on machine  $m$ 
9   endif
10 endfor
```