



# Chapter 2

# Greedy Algorithms

Algorithm Theory  
WS 2015/16

Fabian Kuhn

# Greedy Algorithms

- No clear definition, but essentially:

**In each step make the choice that looks best at the moment!**

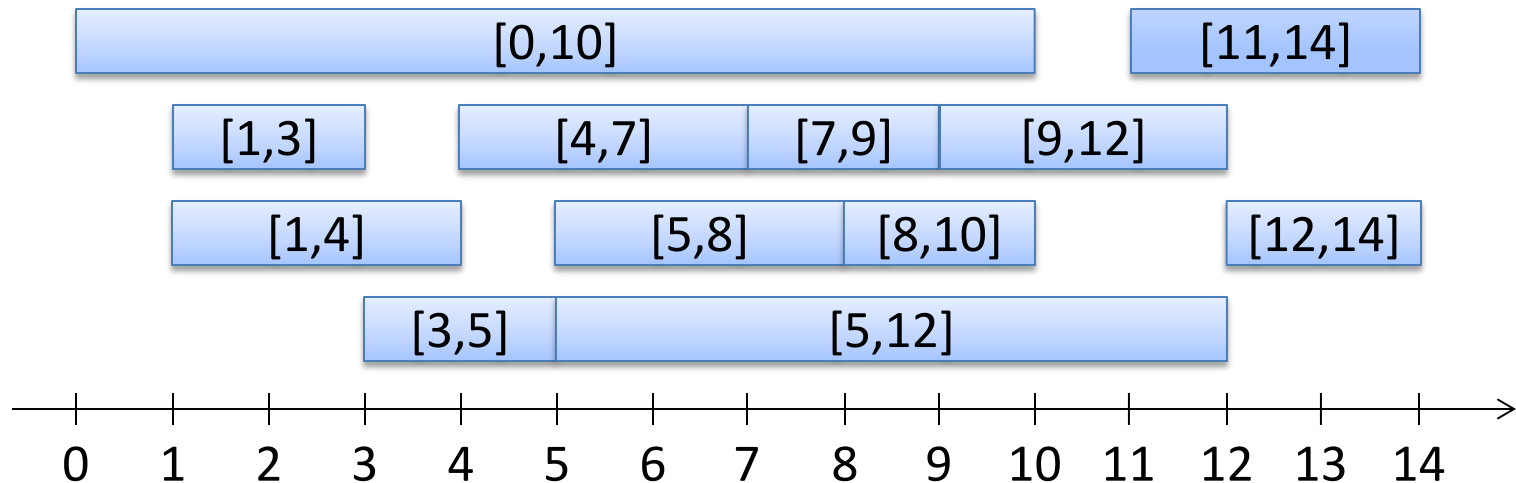
- Depending on problem, greedy algorithms can give
  - Optimal solutions
  - Close to optimal solutions
  - No (reasonable) solutions at all
- If it works, very interesting approach!
  - And we might even learn something about the structure of the problem

**Goal:** Improve understanding where it works (mostly by examples)

# Interval Scheduling

- **Given:** Set of **intervals**, e.g.

$[0,10], [1,3], [1,4], [3,5], [4,7], [5,8], [5,12], [7,9], [9,12], [8,10], [11,14], [12,14]$

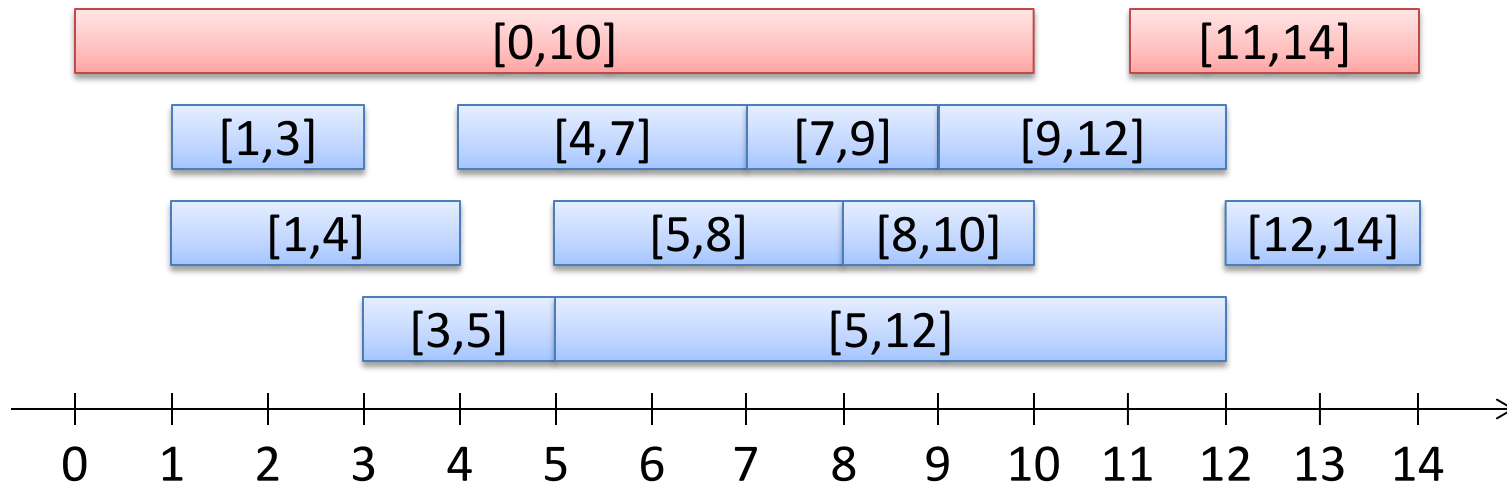


- **Goal:** Select largest possible non-overlapping set of intervals
  - Overlap at boundary ok, i.e.,  $[4,7]$  and  $[7,9]$  are non-overlapping
- **Example:** Intervals are room requests; satisfy as many as possible

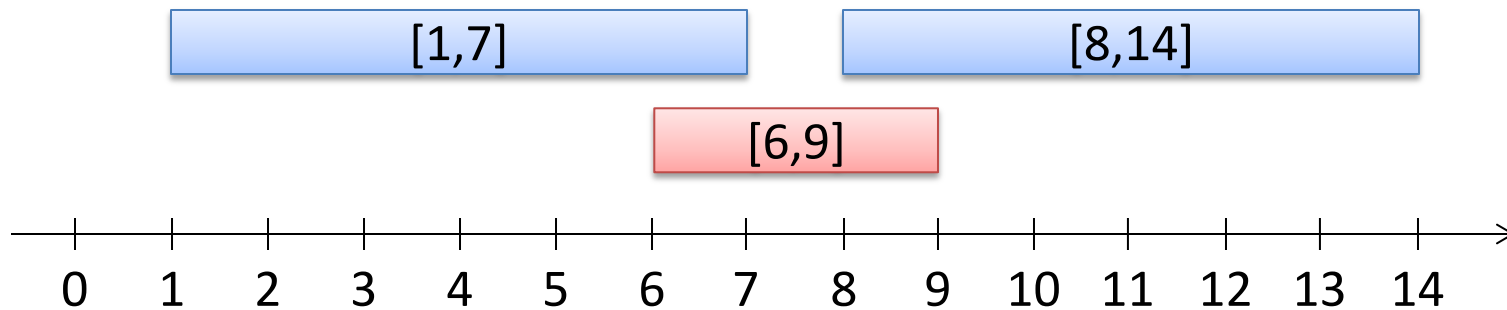
# Greedy Algorithms

- Several possibilities...

**Choose first available interval:**

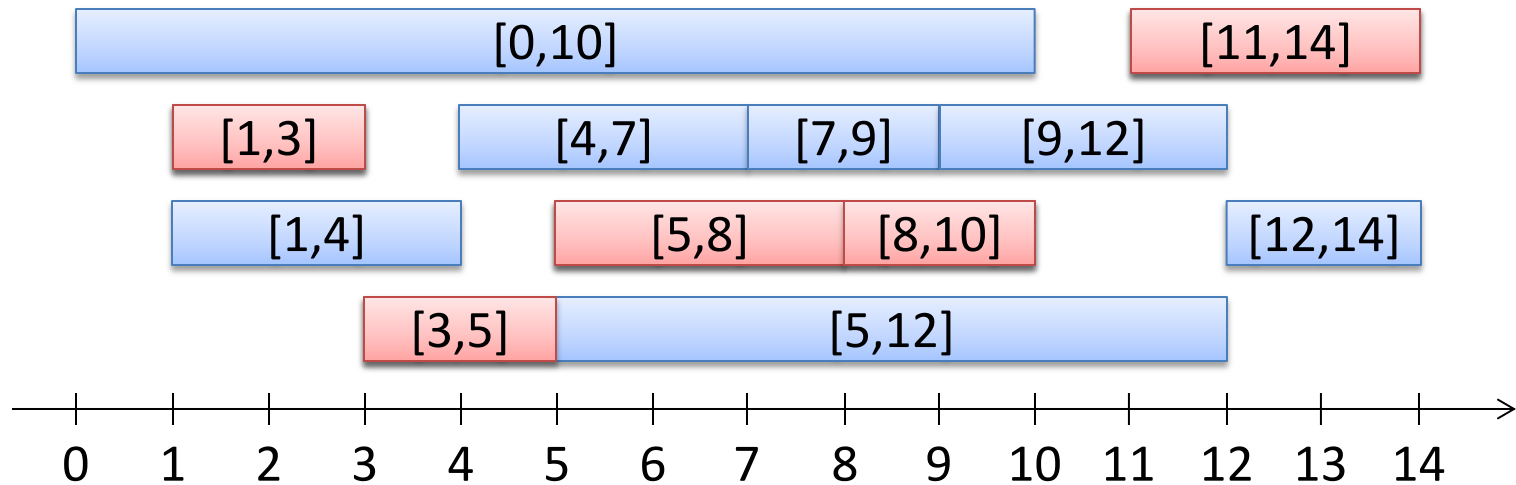


**Choose shortest available interval:**



# Greedy Algorithms

Choose available request with earliest finishing time:



$R :=$  set of all requests;  $S :=$  empty set;

**while**  $R$  is not empty **do**

    choose  $r \in R$  with smallest finishing time

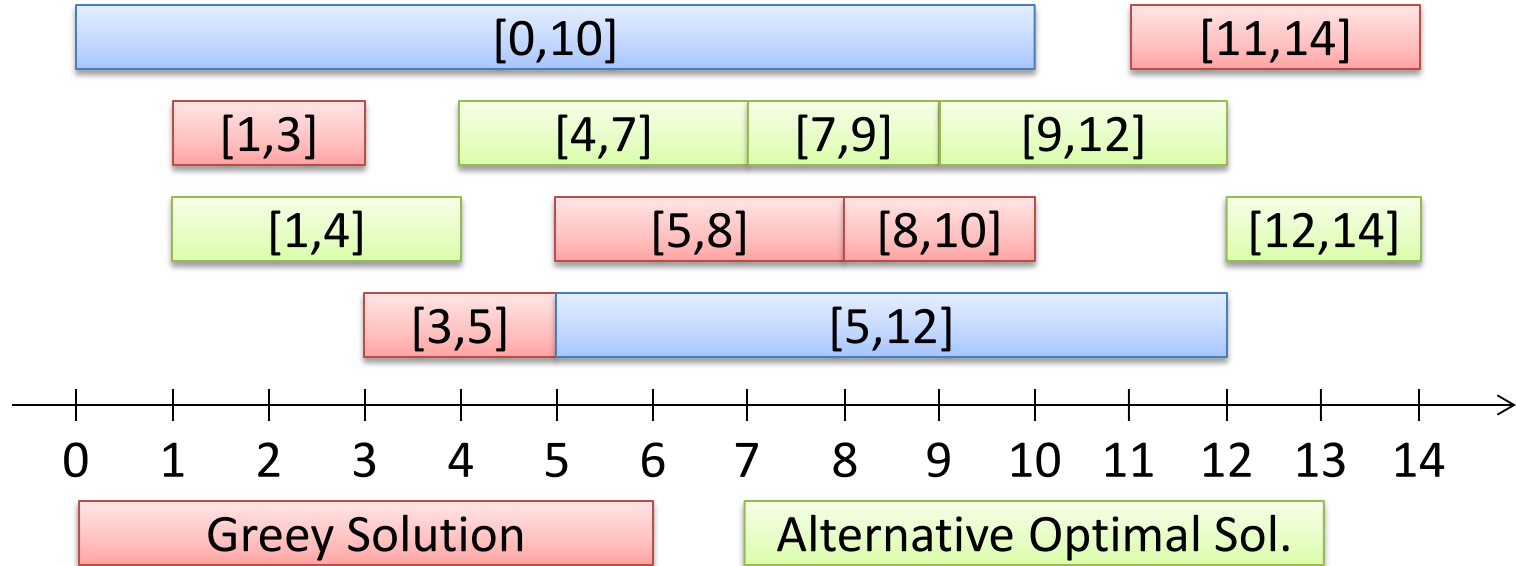
    add  $r$  to  $S$

    delete all requests from  $R$  that are not compatible with  $r$

**end**                   //  $S$  is the solution

# Earliest Finishing Time is Optimal

- Let  $O$  be the set of intervals of an optimal solution
- Can we show that  $S = O$ ?
  - No...

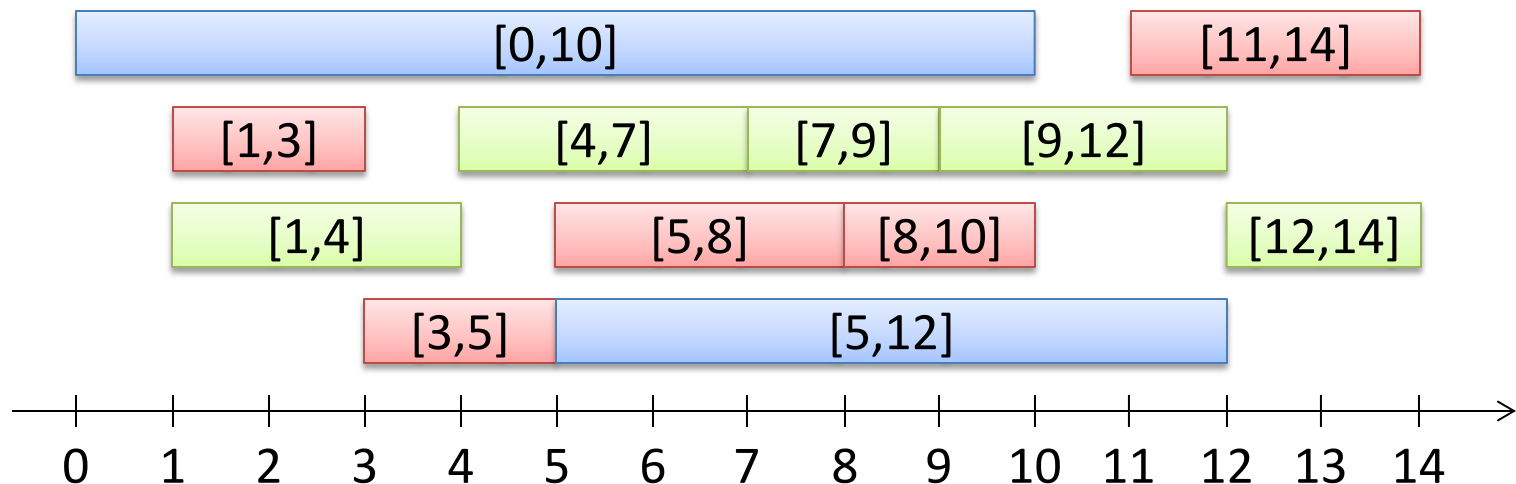


- Show that  $|S| = |O|$ .

# Greedy Stays Ahead

- Greedy Solution:  
 $[a_1, b_1], [a_2, b_2], \dots, [a_{|S|}, b_{|S|}]$ , where  $b_i \leq a_{i+1}$
- Optimal Solution:  
 $[a_1^*, b_1^*], [a_2^*, b_2^*], \dots, [a_{|O|}^*, b_{|O|}^*]$ , where  $b_i^* \leq a_{i+1}^*$
- Assume that  $b_i = \infty$  for  $i > |S|$  and  $b_i^* = \infty$  for  $i > |O|$

**Claim:** For all  $i \geq 1$ ,  $b_i \leq b_i^*$



# Greedy Stays Ahead

**Claim:** For all  $i \geq 1$ ,  $b_i \leq b_i^*$

Proof (by induction on  $i$ ):

**Corollary:** Earliest finishing time algorithm is optimal.



# Weighted Interval Scheduling

Weighted version of the problem:

- Each interval has a weight
- Goal: Non-overlapping set with maximum total weight

Earliest finishing time greedy algorithm fails:

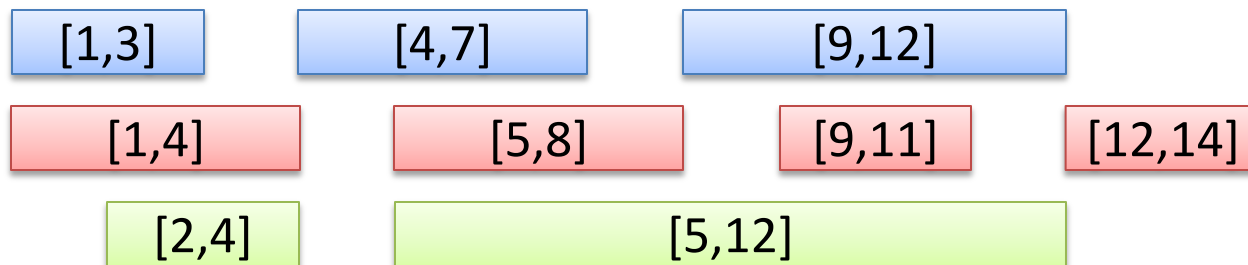
- Algorithm needs to look at weights
- Else, the selected sets could be the ones with smallest weight...

No simple greedy algorithm:

- We will see an algorithm using another design technique later.

# Interval Partitioning

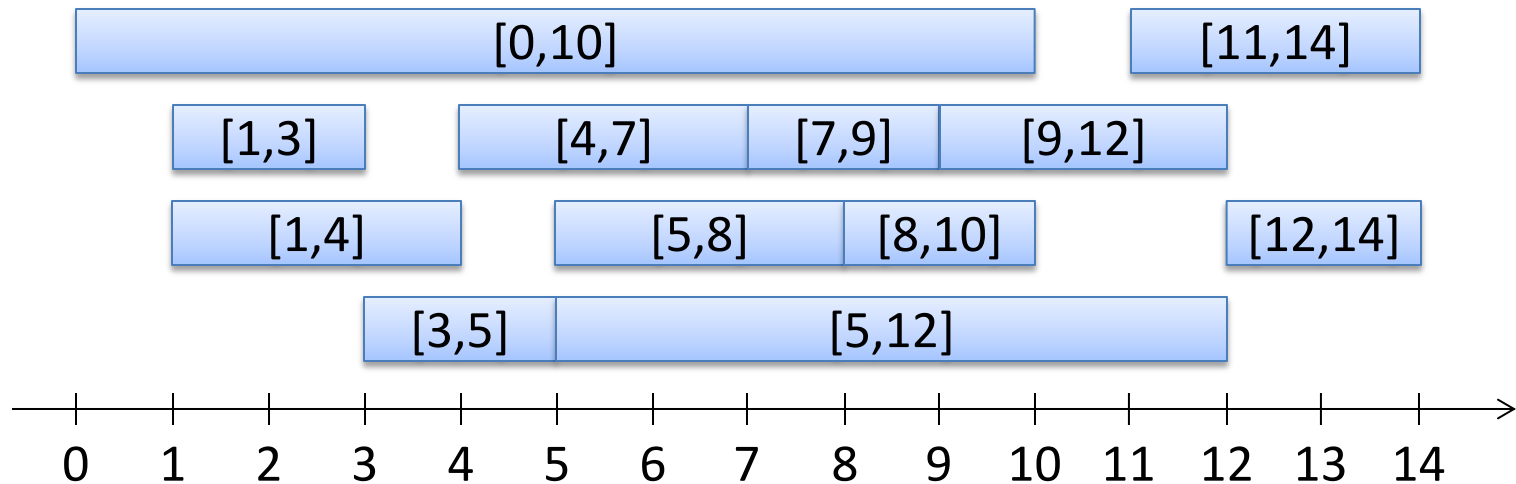
- **Schedule all intervals:** Partition intervals into **as few as possible non-overlapping sets of intervals**
  - Assign intervals to different resources, where each resource needs to get a non-overlapping set
- **Example:**
  - Intervals are requests to use some room during this time
  - Assign all requests to some room such that there are no conflicts
  - Use as few rooms as possible
- **Assignment to 3 resources:**



# Depth

## Depth of a set of intervals:

- Maximum number passing over a single point in time
- Depth of initial example is 4 (e.g.,  $[0,10],[4,7],[5,8],[5,12]$ ):



**Lemma:** Number of resources needed  $\geq$  depth

# Greedy Algorithm

Can we achieve a partition into “depth” non-overlapping sets?

- Would mean that the only obstacles to partitioning are local...

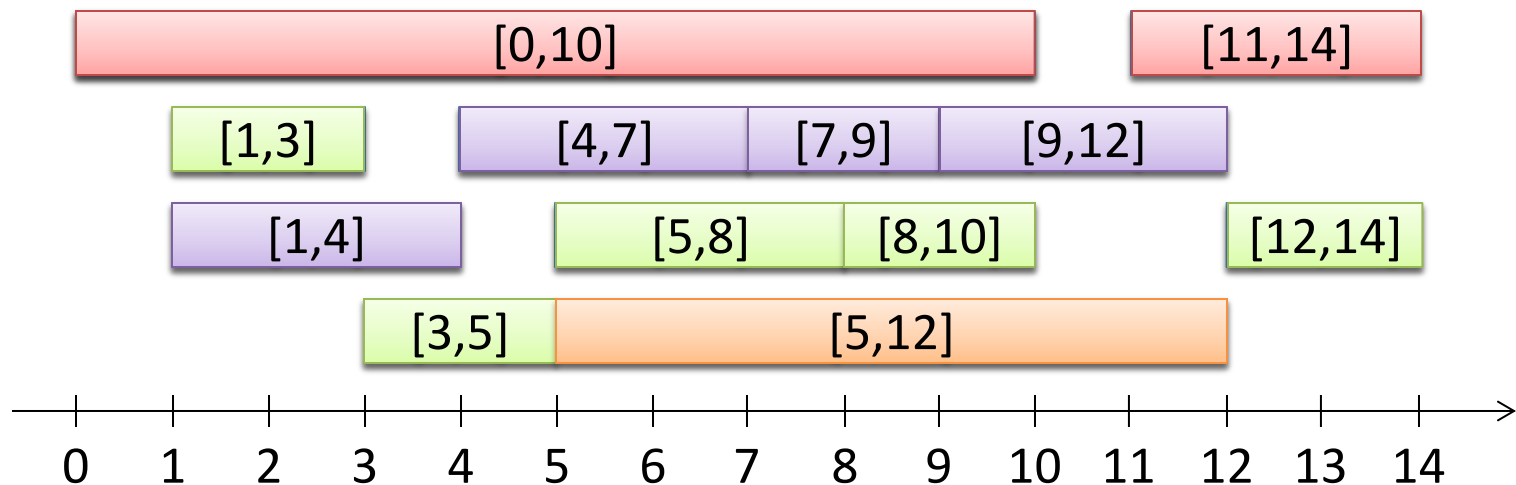
## Algorithm:

- Assigns labels 1, ... to the sets; same label  $\rightarrow$  non-overlapping
1. sort intervals by starting time:  $I_1, I_2, \dots, I_n$
  2. **for**  $i = 1$  **to**  $n$  **do**
  3.     assign smallest possible label to  $I_i$   
    (possible label: different from conflicting intervals  $I_j, j < i$ )
  4. **end**

# Interval Partitioning Algorithm

## Example:

- Labels:    



- Number of labels = depth = 4

# Interval Partitioning: Analysis

## Theorem:

- a) Let  $d$  be the depth of the given set of intervals. The algorithm assigns a label from  $1, \dots, d$  to each interval.
- b) Sets with the same label are non-overlapping

## Proof:

- b) holds by construction
- For a):
  - All intervals  $I_j, j < i$  overlapping with  $I_i$ , overlap at the beginning of  $I_i$
  
  - At most  $d - 1$  such intervals  $\rightarrow$  some label in  $\{1, \dots, d\}$  is available.