Algorithms: Greedy Method

Activity Selection Problem

Course Code: CSE-2217
Data Structure and Algorithms II

Lec Saifur Rahman
Email: saifur@cse.uiu.ac.bd
Mobile: +8801303529289

Greedy Algorithms: Principles

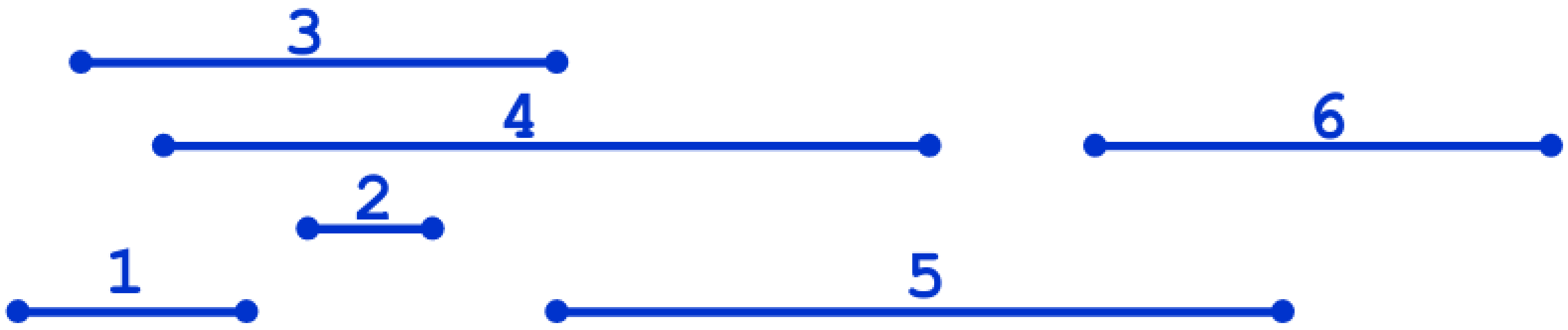
- A greedy algorithm always makes the choice that looks best at the moment.
- A greedy algorithm works in phases.
  At each phase:
  - You take the best you can get right now, without regard for future consequences.
  - Your hope that by choosing a local optimum at each step, you will end up at a global optimum.
  - For some problems, it works

The Activity Selection Problem

- Input: A set of activities $S = \{a_1, \dots, a_n\}$
  - Each activity a_i has a start time S_i and a finish time f_i, where $0 \leq S_i < f_i < \infty$
  - If selected, activity a_i takes place during the half-open time interval $[S_i, f_i)$
- Two activities are compatible if and only if their intervals do not overlap
- Output: a maximum-size subset of mutually compatible activities.

The Activity Selection Problem

- Formally:
  - Given a set S of n activities
    $s_i =$ start time of activity i
    $f_i =$ finish time of activity I
  - Find max-size subset A of compatible activities

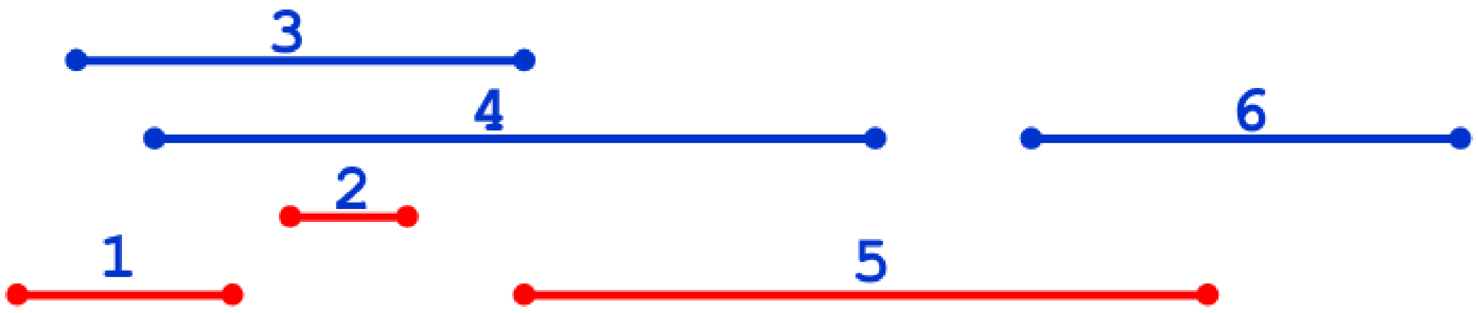


The Activity Selection Problem

- Formally:
  - Given a set S of n activities

    $s_i =$ start time of activity i

    $f_i =$ finish time of activity i
  - Find max-size subset A of compatible activities



The Activity Selection Problem

- Here are a set of start and finish times

i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14

- What is the maximum number of activities that can be completed?

The Activity Selection Problem

- Here are a set of start and finish times

i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14

- What is the maximum number of activities that can be completed?
  - $\{a_3, a_9, a_{11}\}$ can be completed

The Activity Selection Problem

- Here are a set of start and finish times

i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14

- What is the maximum number of activities that can be completed?
  - $\{a_3, a_9, a_{11}\}$ can be completed
  - But so can $\{a_1, a_4, a_8, , a_{11}\}$ which is a larger set

The Activity Selection Problem

- Here are a set of start and finish times

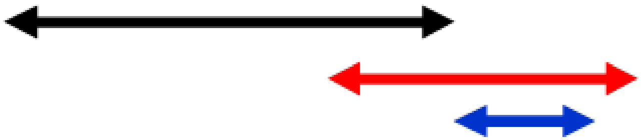
i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14

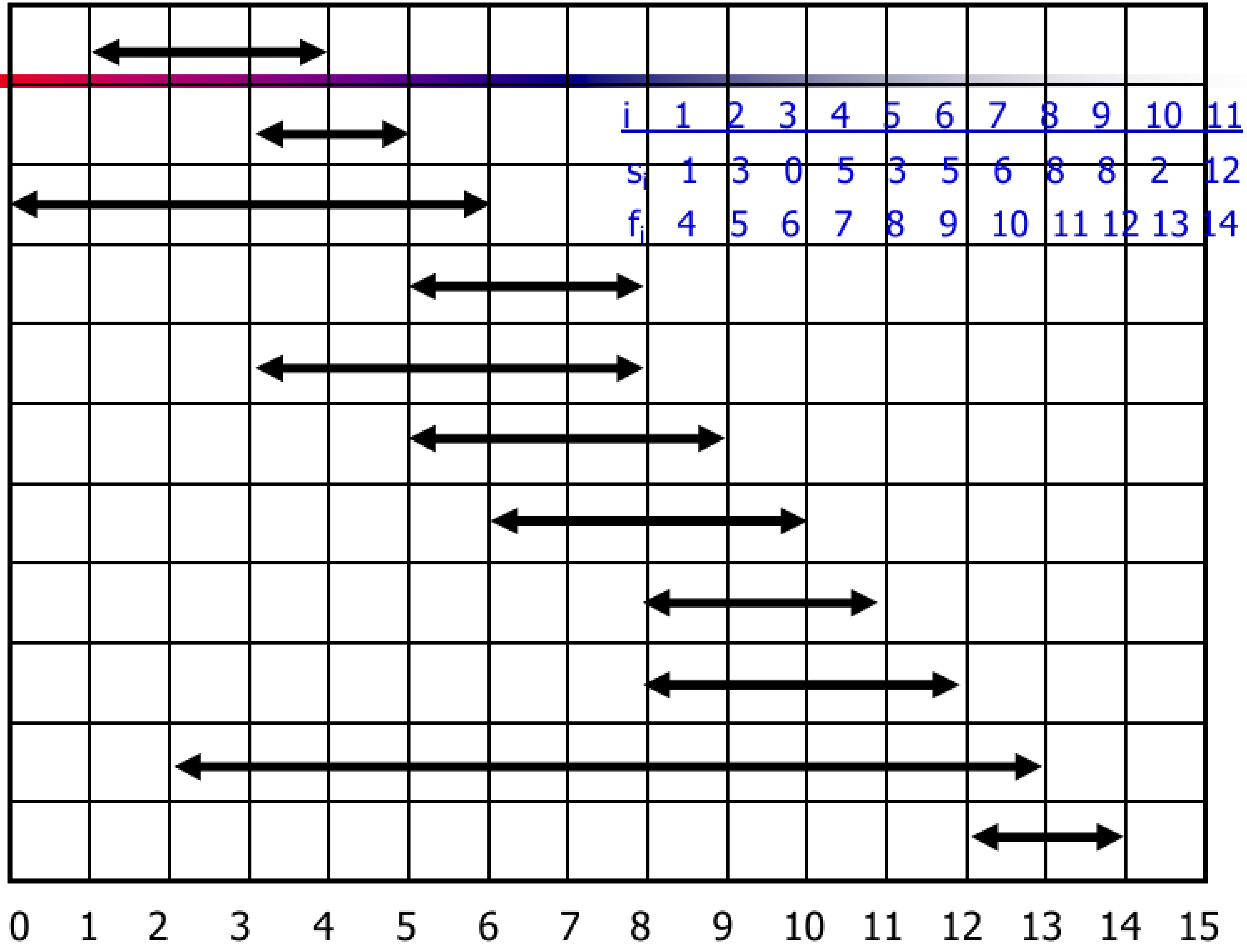
- What is the maximum number of activities that can be completed?
  - $\{a_3, a_9, a_{11}\}$ can be completed
  - But so can $\{a_1, a_4, a_8, a_{11}\}$ which is a larger set
  - But it is not unique, consider $\{a_2, a_4, a_9, a_{11}\}$

Interval Representation

- Here are a set of start and finish times

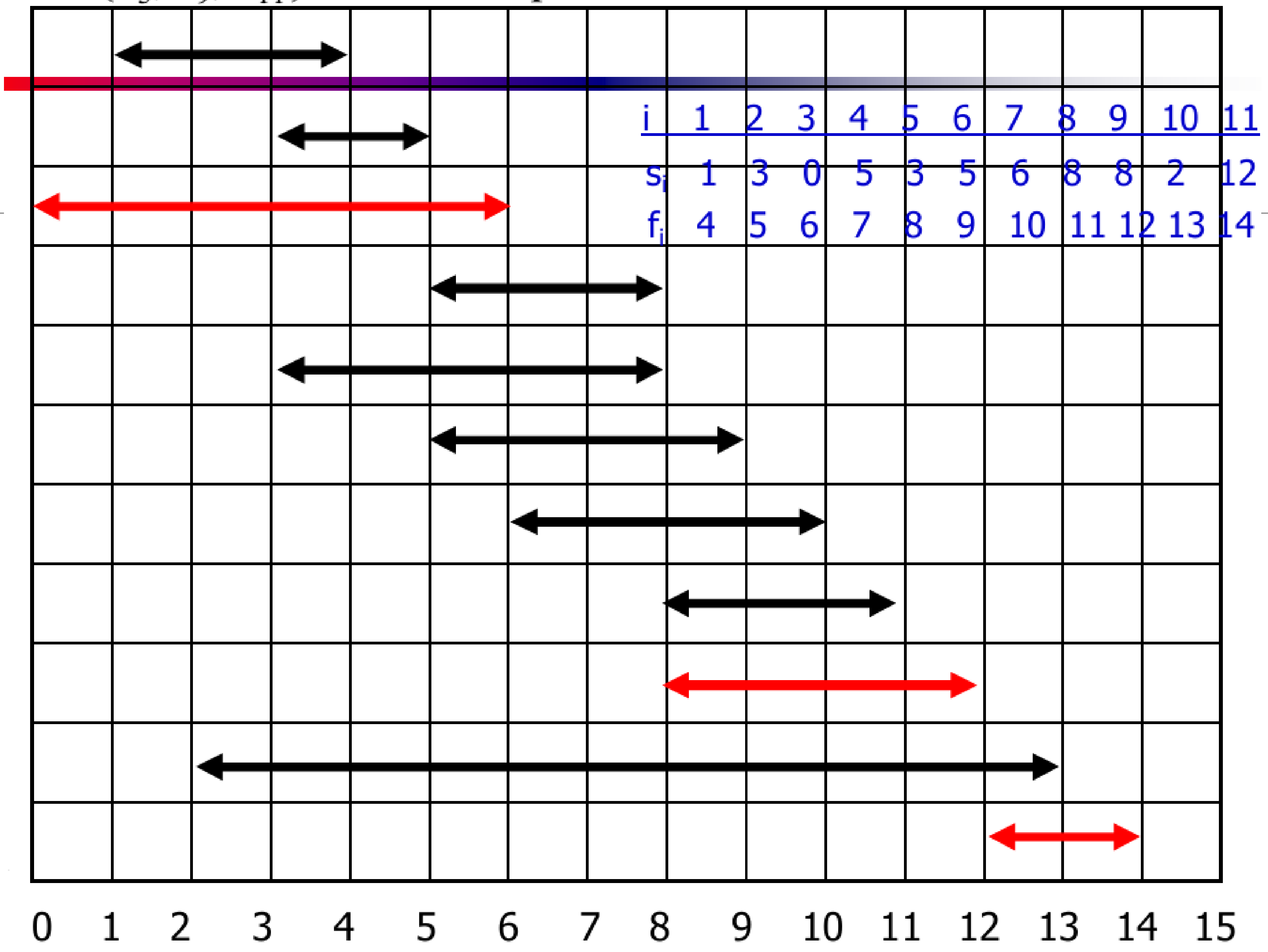
i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14





i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14

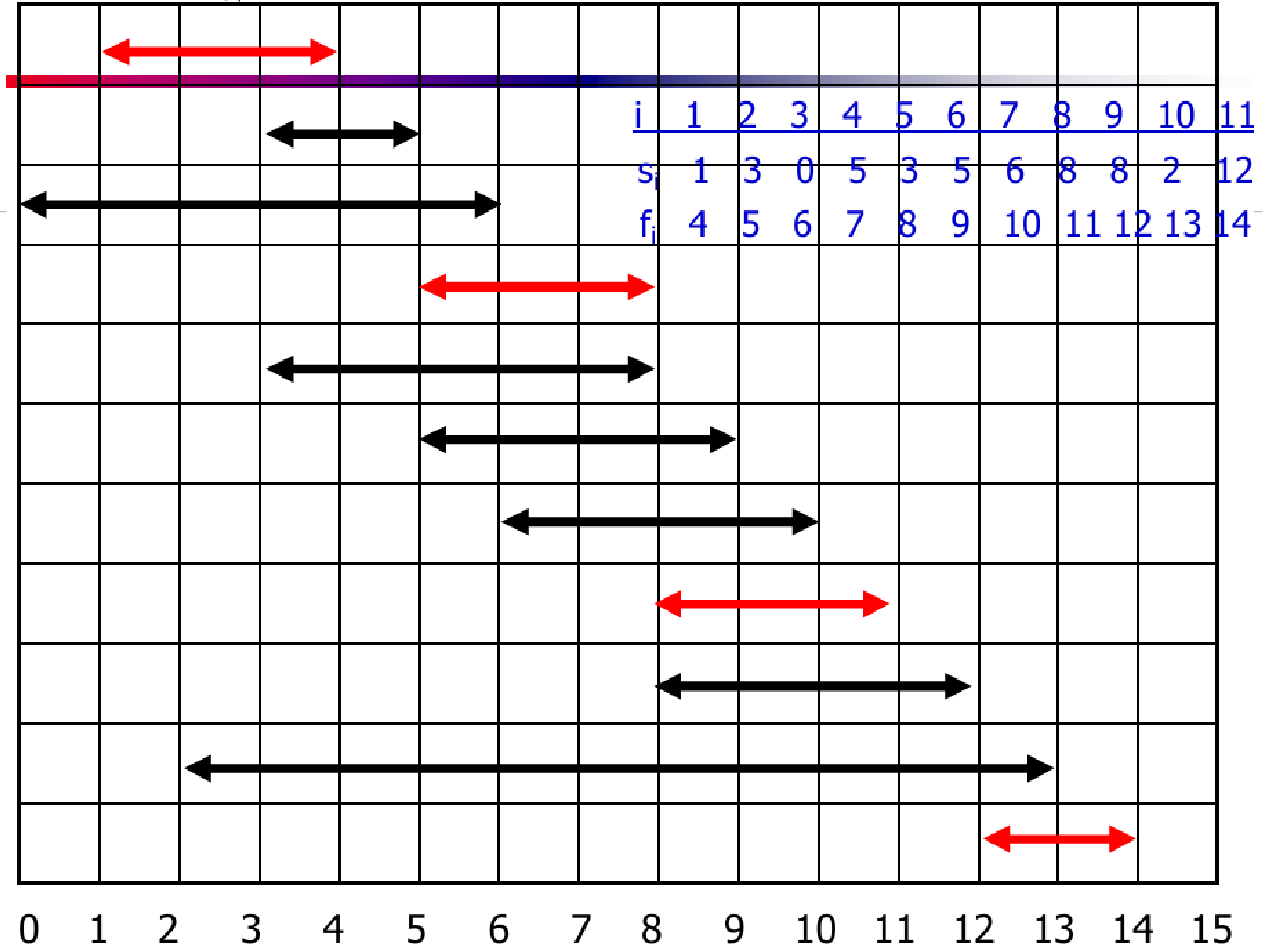
$\{a_3, a_9, a_{11}\}$ can be completed



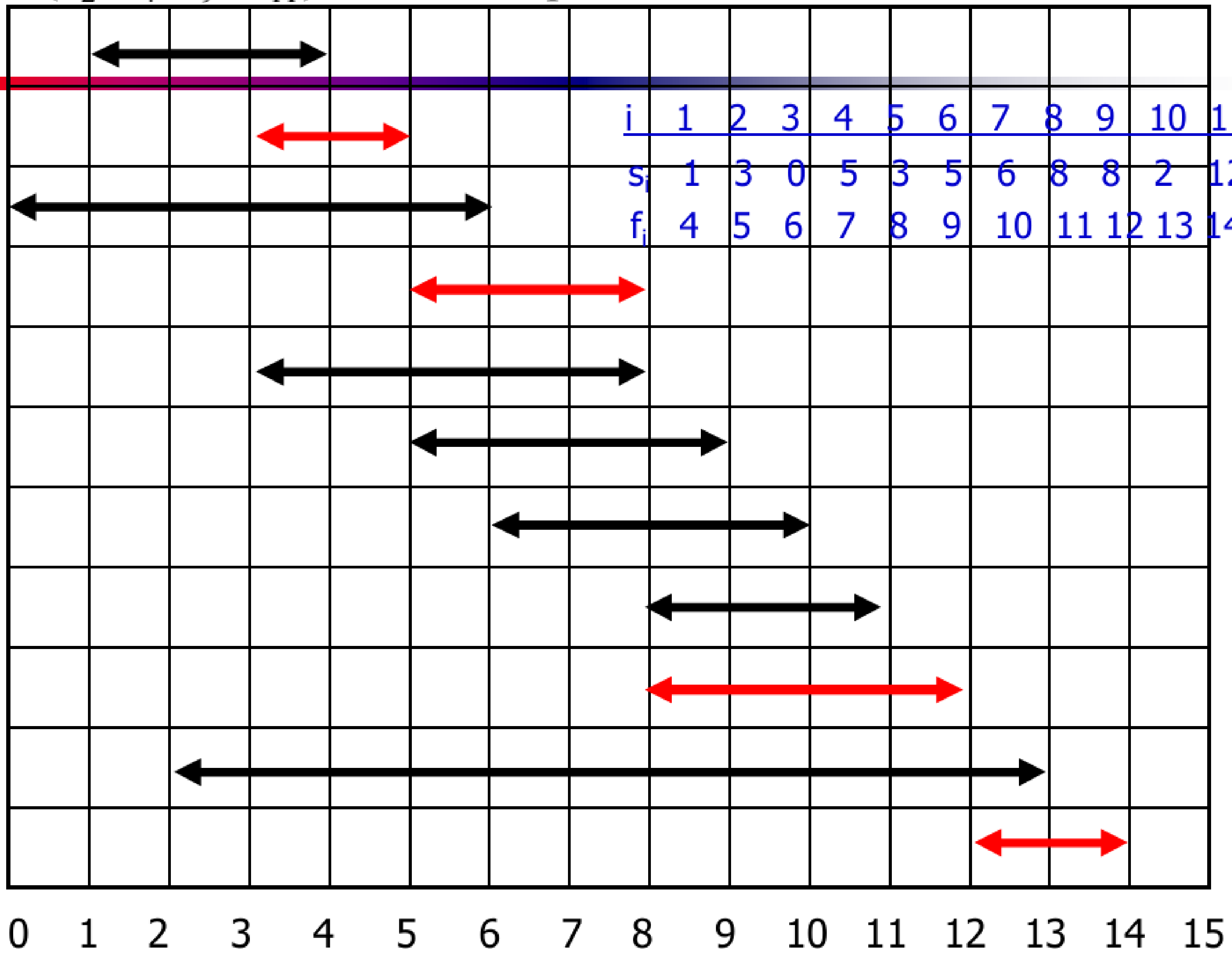
i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14

$\{a_1, a_4, a_8, a_{11}\}$ can be completed

i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14



$\{a_2, a_4, a_9, a_{11}\}$ can be completed



i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14

Solving the Activity Selection Problem

- The greedy choice can be applied to find solutions (the maximum number of activities that can be performed) by using
  - Earliest finish time

    Always gives optimal solution

  - Earliest start time

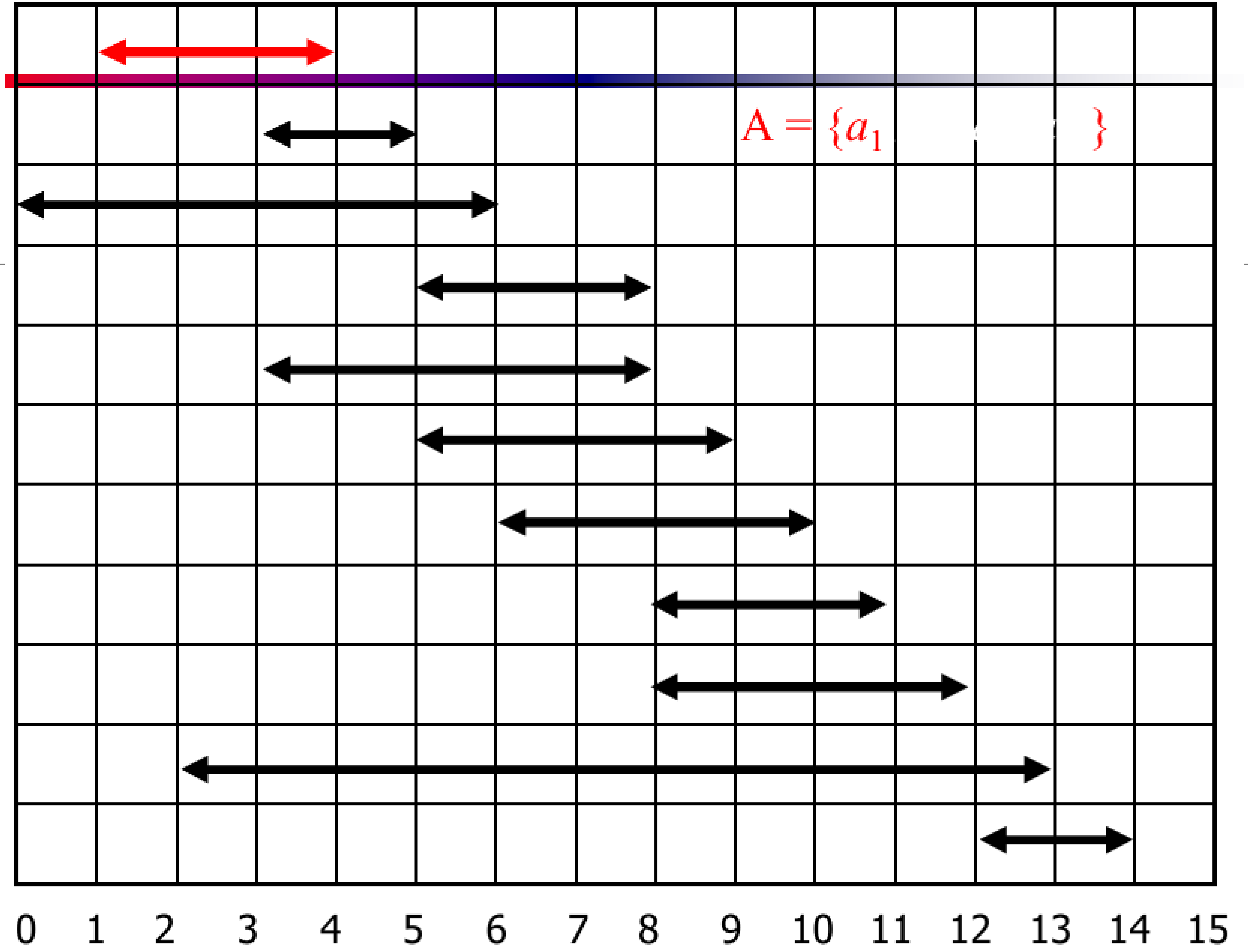
    Does not always give optimal solution

  - Shortest interval

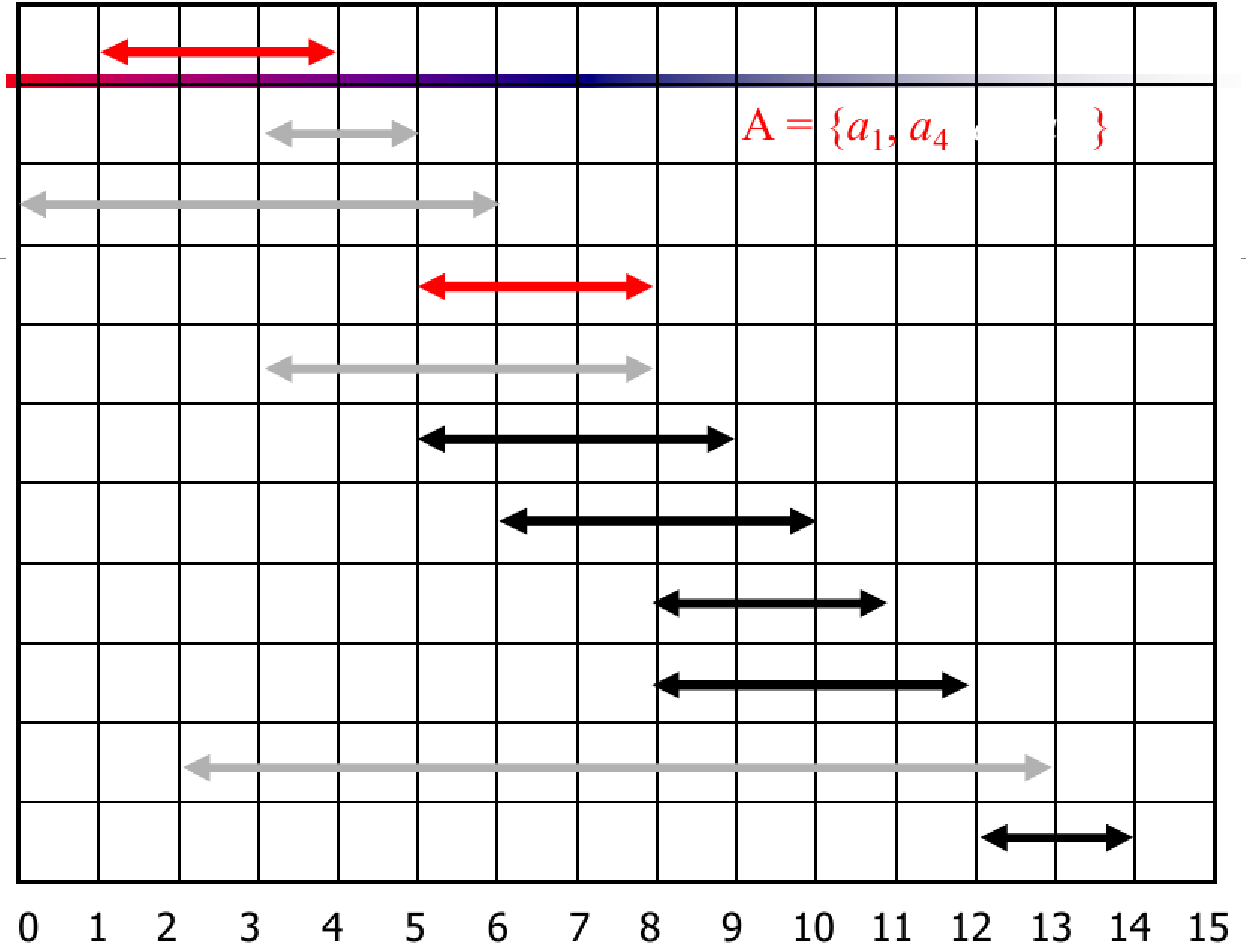
    Does not always give optimal solution

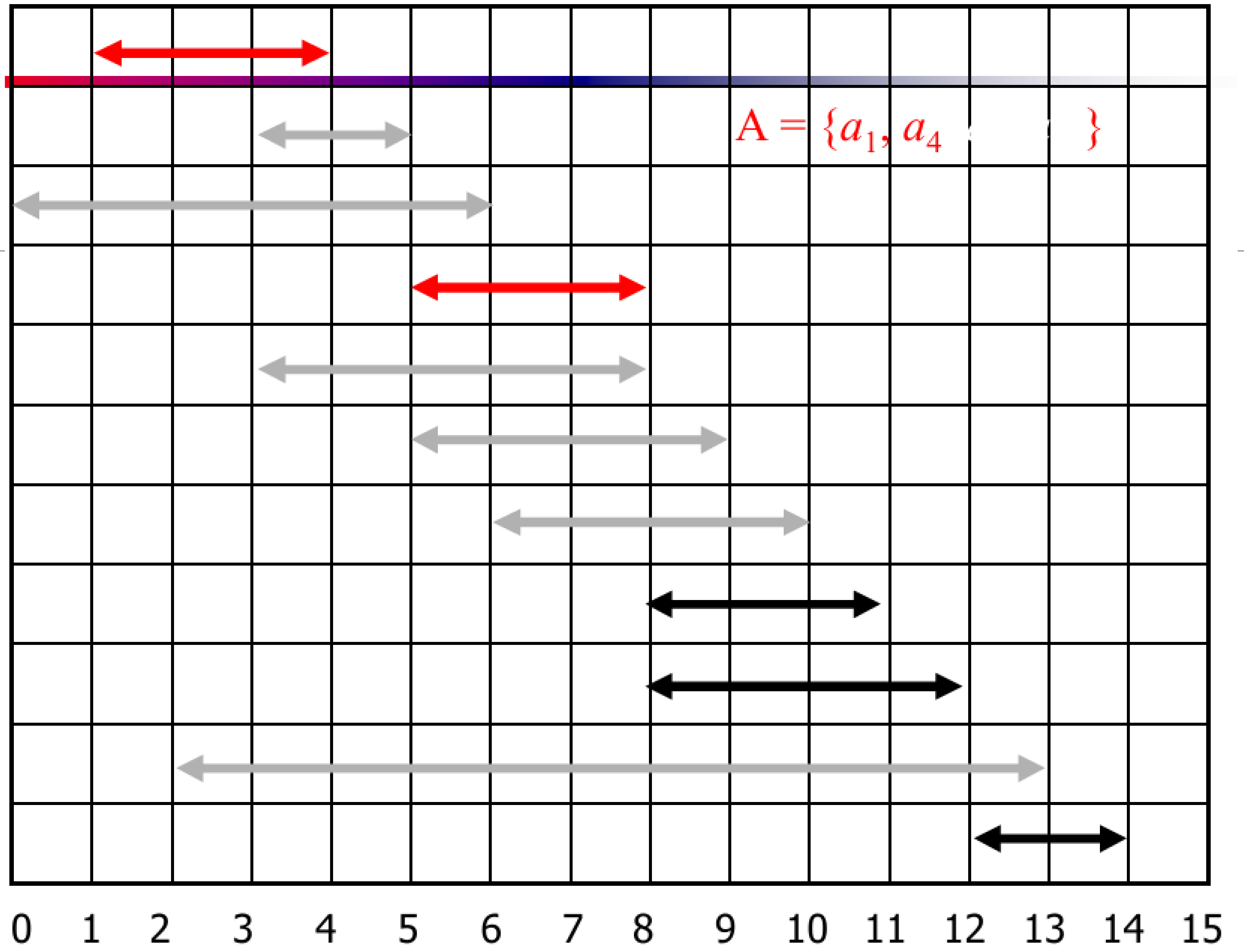
Early Finish Greedy

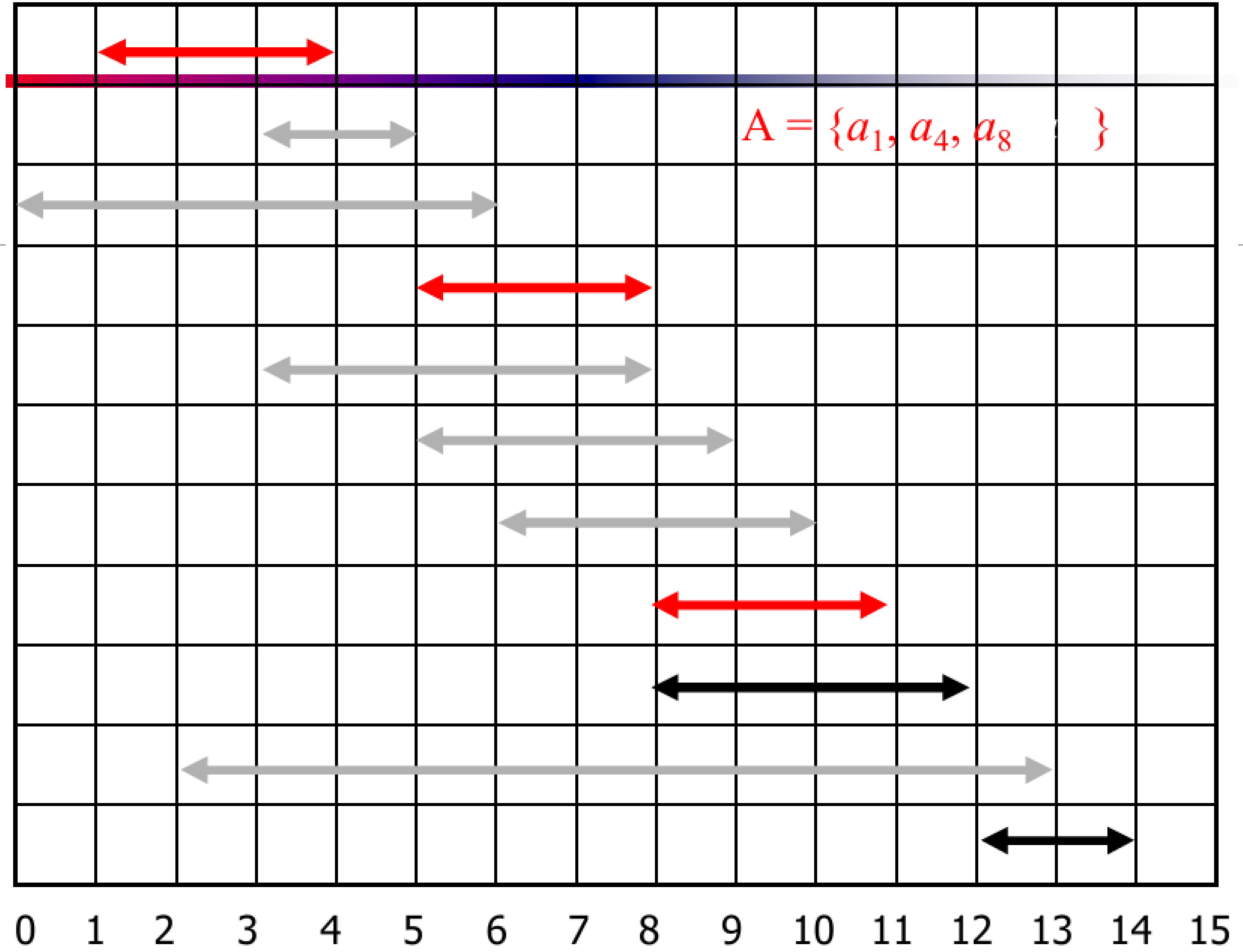
- Select the activity with the earliest finish
- Eliminate the activities that could not be scheduled
- Repeat!

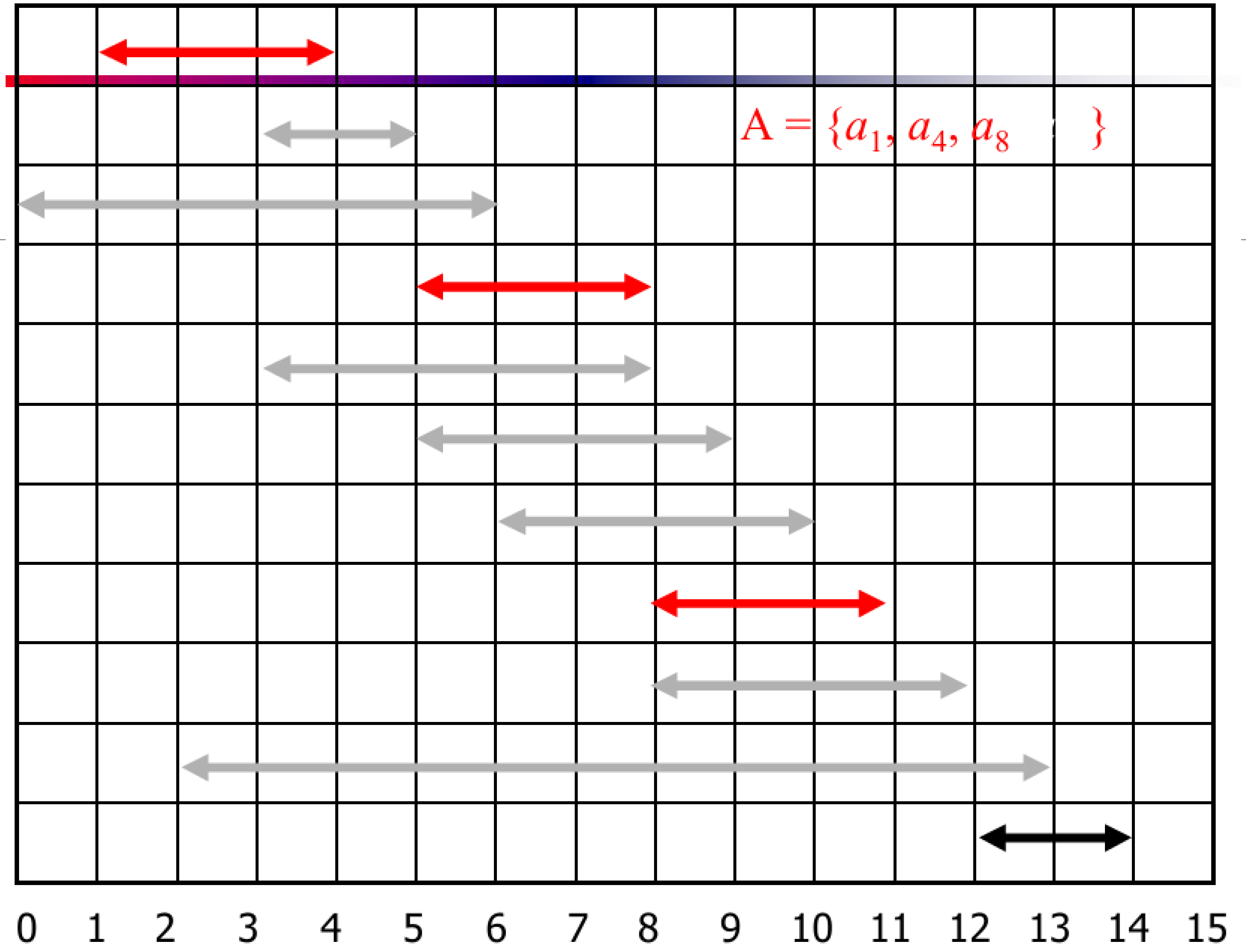


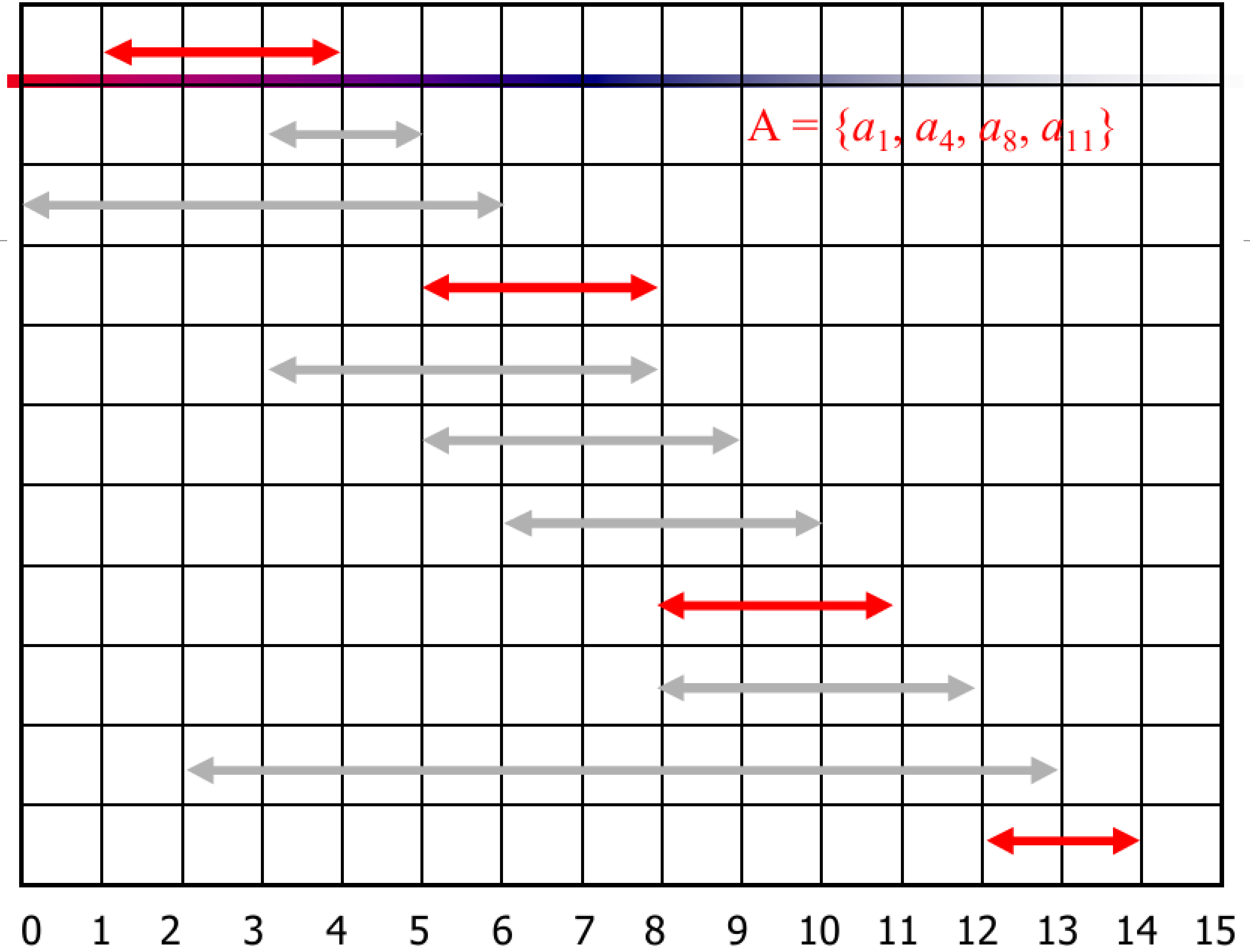
$A = \{a_1$ $\}$
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15


$A = \{a_1, a_4$ $\}$
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15


$A = \{a_1, a_4 \}$
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15


$A = \{a_1, a_4, a_8 \quad \}$
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15


$A = \{a_1, a_4, a_8 \}$
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15


$A = \{a_1, a_4, a_8, a_{11}\}$
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Assuming activities are sorted by finish time

```
GREEDY-ACTIVITY-SELECTOR(s, f)
n ← length[s]
A ← {a_1}
i ← 1
for m ← 2 to n
    do if s_m ≥ f_i
        then A ← A ∪ {a_m}
             i ← m
return A
```

- Running time: $\Theta(n)$ if activities are already sorted by finish time; else $\Theta(n\log n)$

Why is it Greedy?

- Greedy in the sense that it leaves as much opportunity as possible for the remaining activities to be scheduled.
- The greedy choice is the one that maximizes the amount of unscheduled time remaining.

- We will show that this algorithm uses the following properties
  - The algorithm satisfies the greedy-choice property.
  - The problem has the optimal substructure property.

Elements of Greedy Strategy

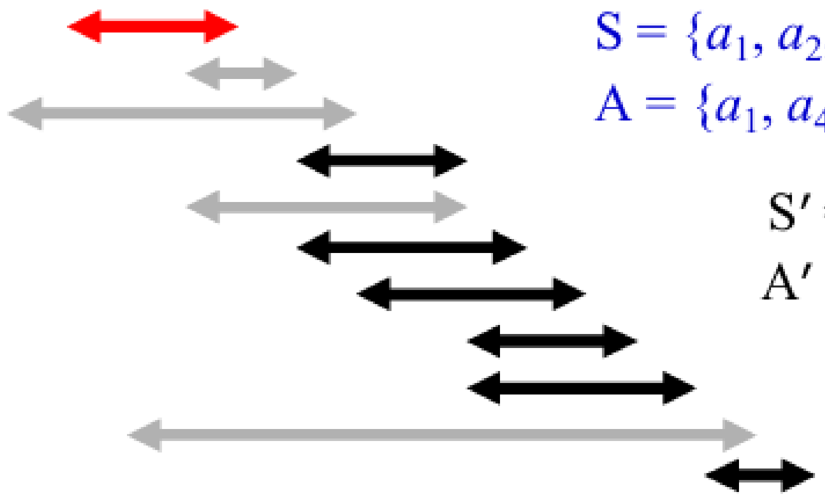
- A greedy algorithm makes a sequence of choices, each of the choices that seems best at the moment is chosen
  - NOT always produces an optimal solution
- Two ingredients that are exhibited by most problems that lend themselves to a greedy strategy
  - Greedy-choice property
  - Optimal substructure property

Greedy-Choice Property

- A globally optimal solution can be arrived at by making a locally optimal (greedy) choice
  - Make whatever choice seems best at the moment and then solve the sub-problem arising after the choice is made.
  - The choice made by a greedy algorithm may depend on choices so far, but cannot depend on any future choices or on the solutions to sub-problems
- Of course, we must prove that a greedy choice at each step yields a globally optimal solution.

Optimal Substructure Property

- A problem exhibits optimal substructure if an optimal solution to the problem contains within it optimal solutions to sub-problems.
  - If an optimal solution A to S begins with activity 1, then A’ = A – {1} is optimal to S’ = {i ∈ S: $s_i \geq f_1$}



$S = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{10}, a_{11}\}$

$A = \{a_1, a_4, a_8, a_{11}\}$

$S' = \{a_4, a_6, a_7, a_8, a_9, a_{11}\}$

$A' = \{a_4, a_8, a_{11}\}$