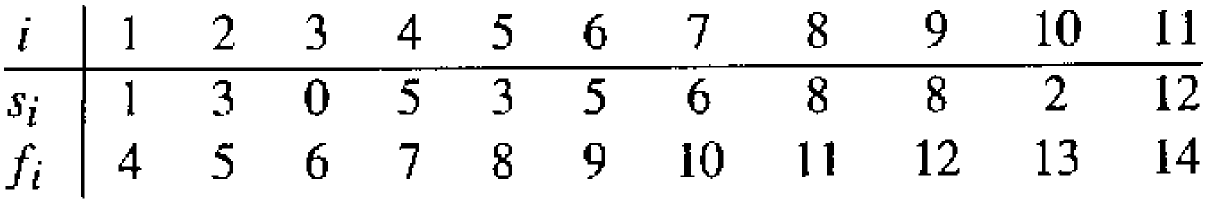
An Activity Selection Problem (Conference Scheduling Problem)

- **Input: A set of activities S = {$a_1, \ldots, a_n$}**
- Each activity has start time and a finish time
  - $a_i = (s_i, f_i)$
- Two activities are compatible if and only if their interval does not overlap
- **Output: a maximum-size subset of mutually 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

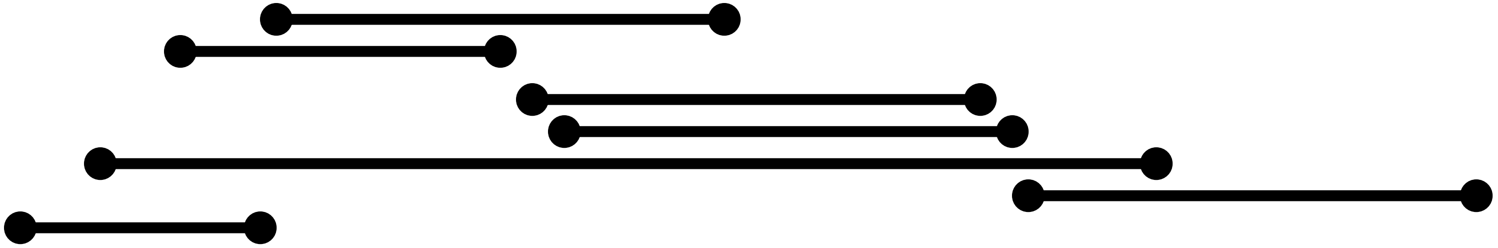
- What is the maximum number of activities that can be completed?
  - $\{a_3, a_9, a_{11}\}$ can be completed
  - But so can $\{a_1, a_4, a_8, a_{11}\}$ which is a larger set
  - But it is not unique, consider $\{a_2, a_4, a_9, a_{11}\}$

The Activity Selection Problem

Input: list of time-intervals L

Output: a non-overlapping subset S of the intervals

Objective: maximize |S|



3,7

2,4

5,8

6,9

1,11

10,12

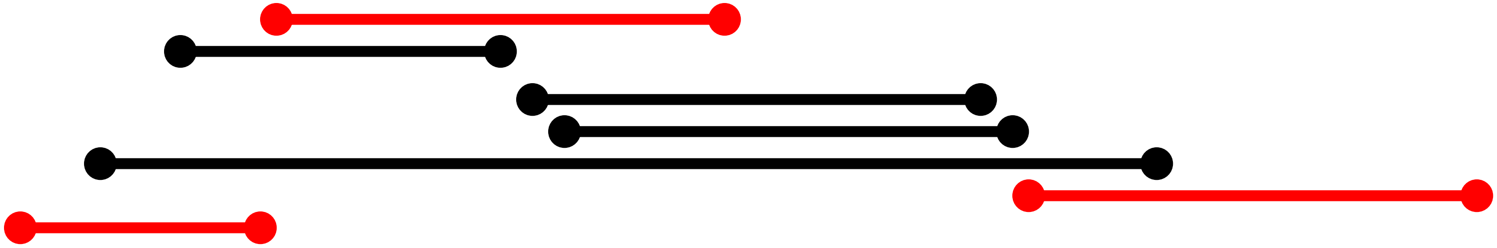
0,3

The Activity Selection Problem

Input: list of time-intervals L

Output: a non-overlapping subset S of the intervals

Objective: maximize |S|



Answer = 3

3,7
2,4
5,8
6,9
1,11
10,12
0,3

The Activity Selection Problem

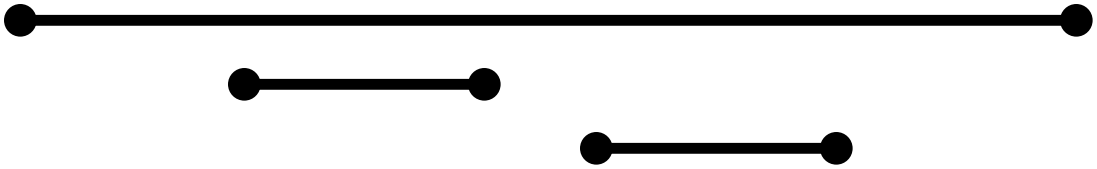
Algorithm 1:

1. sort the activities by the starting time
2. pick the first activity a
3. remove all activities conflicting with a
4. repeat

The Activity Selection Problem

Algorithm 1:

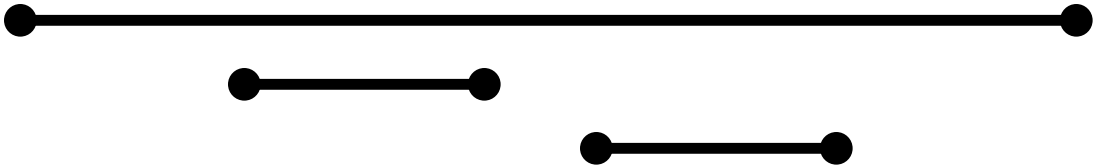
1. sort the activities by the starting time
2. pick the first activity ***"a"***
3. remove all activities conflicting with ***"a"***
4. repeat



The Activity Selection Problem

Algorithm 1:

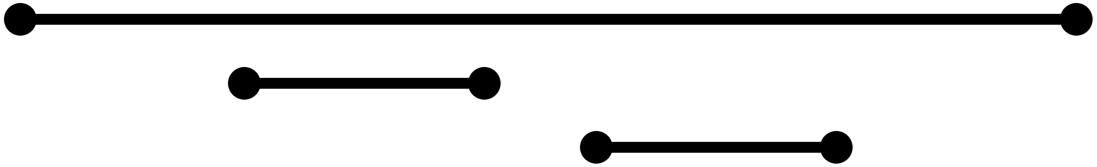
1. ~~sort the activities by the starting time~~
2. ~~pick the first activity ***“a”***~~
3. ~~remove all activities conflicting with ***“a”***~~
4. ~~repeat~~



The Activity Selection Problem

Algorithm 2:

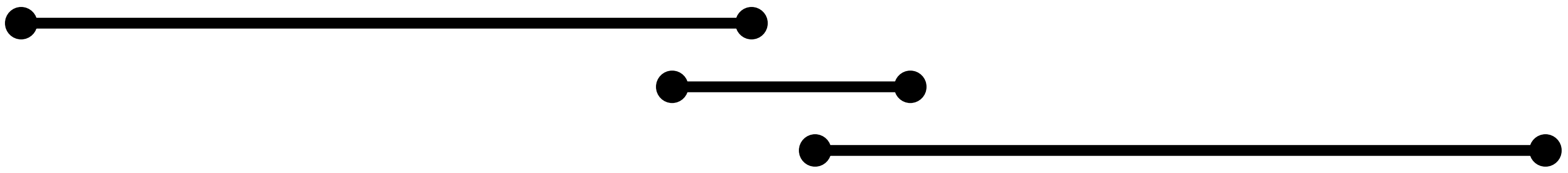
1. sort the activities by length
2. pick the shortest activity ***"a"***
3. remove all activities conflicting with ***"a"***
4. repeat



The Activity Selection Problem

Algorithm 2:

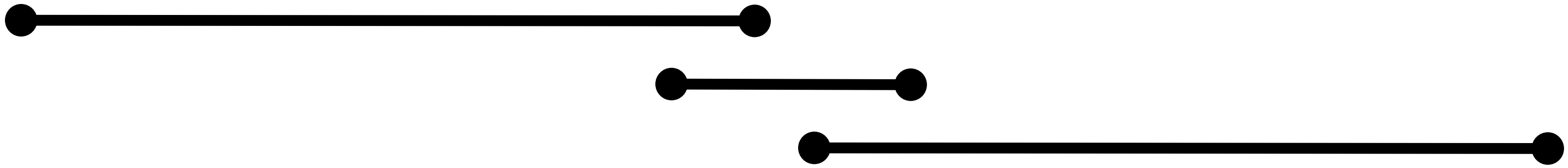
1. sort the activities by length
2. pick the shortest activity ***"a"***
3. remove all activities conflicting with ***"a"***
4. repeat



The Activity Selection Problem

Algorithm 2:

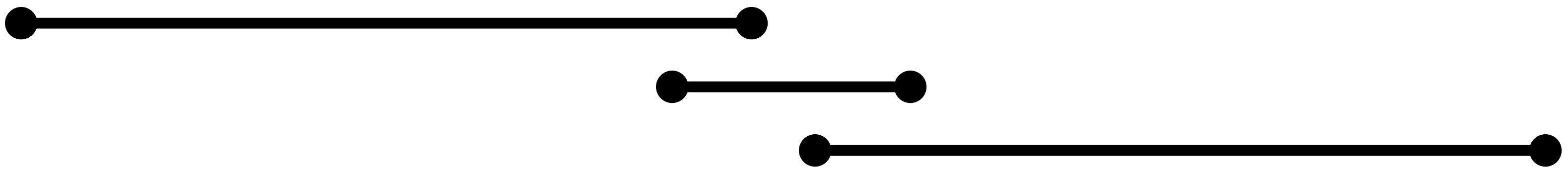
1. ~~sort the activities by length~~
2. ~~pick the shortest activity ***“a”***~~
3. ~~remove all activities conflicting with ***“a”***~~
4. ~~repeat~~



The Activity Selection Problem

Algorithm 3:

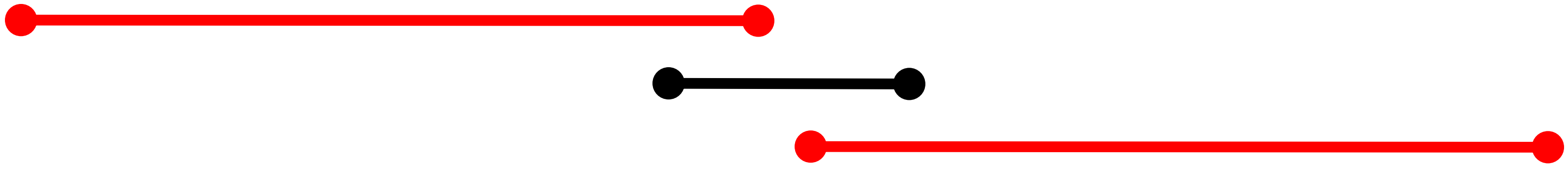
1. **sort** the activities by **ending time**
2. **pick** the activity which **ends first**
3. remove all activities conflicting with a
4. repeat



The Activity Selection Problem

Algorithm 3:

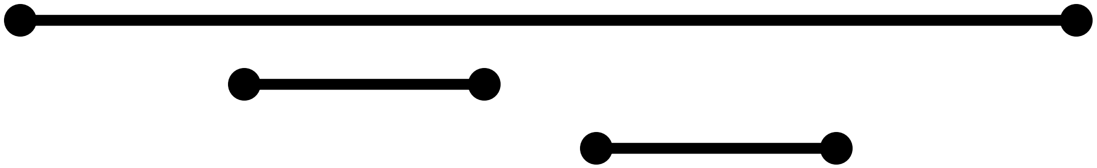
1. sort the activities by ending time
2. pick the activity which ends first
3. remove all activities conflicting with a
4. repeat



The Activity Selection Problem

Algorithm 3:

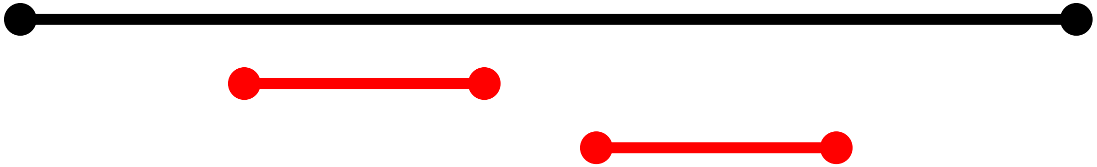
1. sort the activities by ending time
2. pick the activity which ends first
3. remove all activities conflicting with a
4. repeat



The Activity Selection Problem

Algorithm 3:

1. sort the activities by ending time
2. pick the activity which ends first
3. remove all activities conflicting with a
4. repeat



The Activity Selection Problem

Algorithm 3:

1. sort the activities by ending time
2. pick the activity a which ends first
3. remove all activities conflicting with a
4. repeat

Theorem:

Algorithm 3 gives an optimal solution to the activity selection problem.

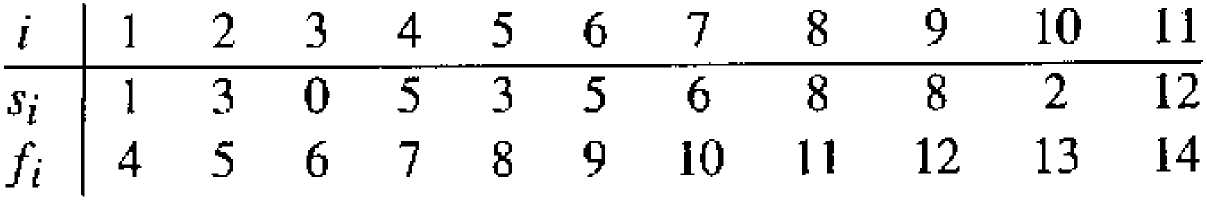
Activity Selection Algorithm

Idea: At each step, select the activity with the smallest finish time that is compatible with the activities already chosen.

```
Greedy-Activity-Selector(s, f)
    n <– length[s]
    A <– {1}                     {Automatically select first activity}
    j <– 1                       {Last activity selected so far}
    for i <– 2 to n do
        if si >= fj then
            A <– A U {i}         {Add activity i to the set}
            j <– i               {record last activity added}
    return A
```

The Activity Selection Problem

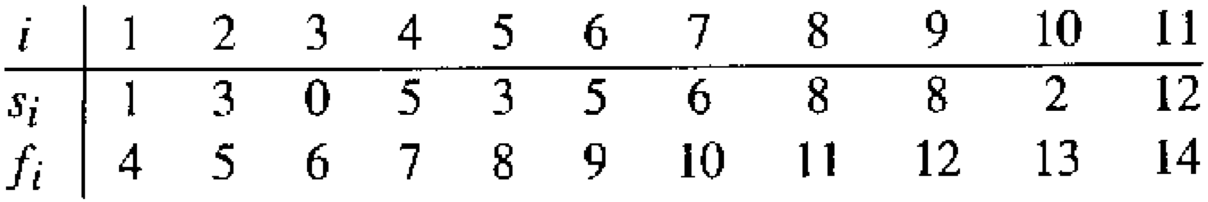
- Here are a set of start and finish times



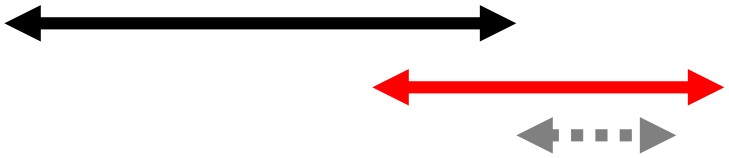
i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14

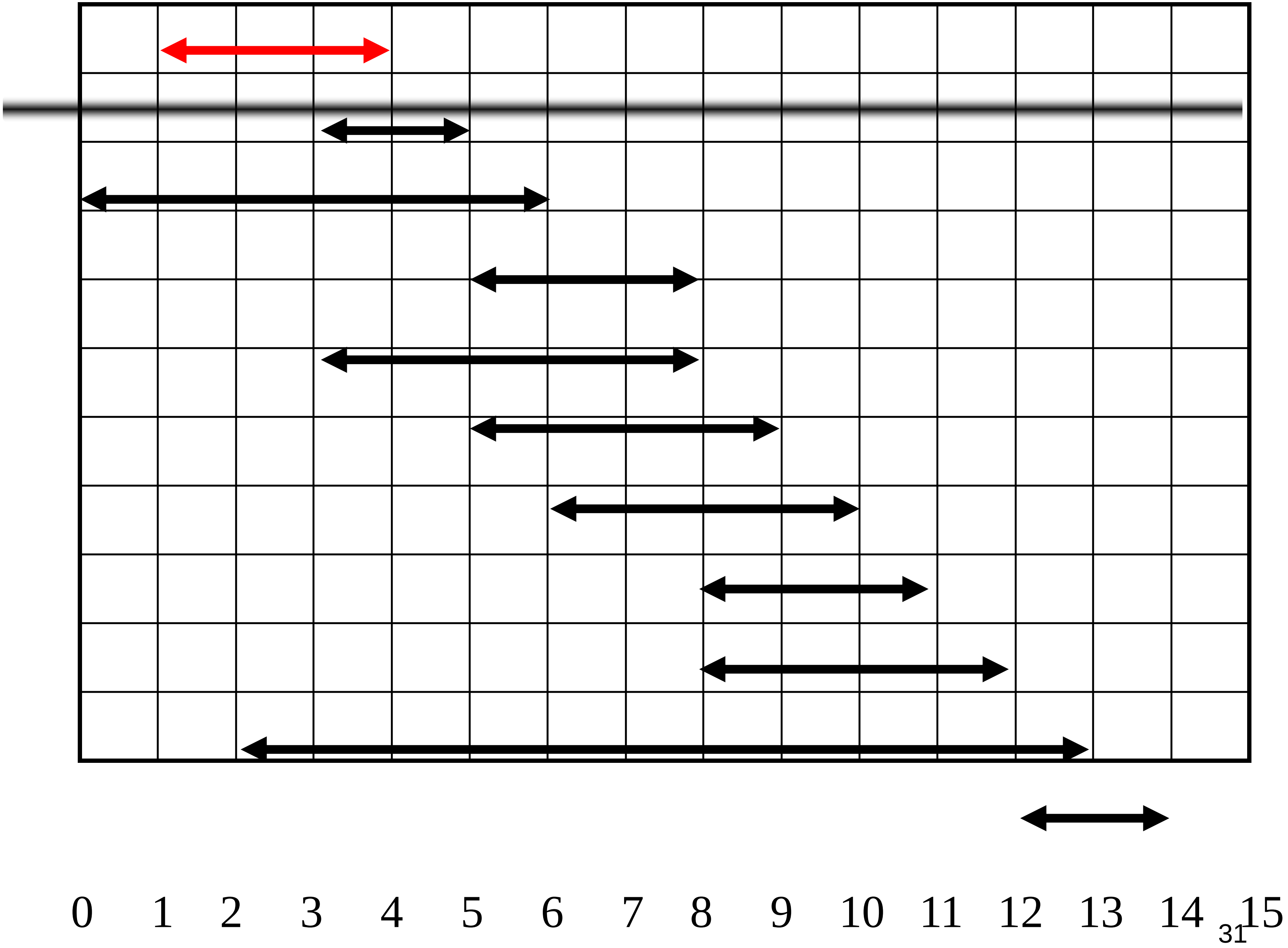
- What is the maximum number of activities that can be completed?
  - $\{a_3, a_9, a_{11}\}$ can be completed
  - But so can $\{a_1, a_4, a_8, a_{11}\}$ which is a larger set
  - But it is not unique, consider $\{a_2, a_4, a_9, a_{11}\}$

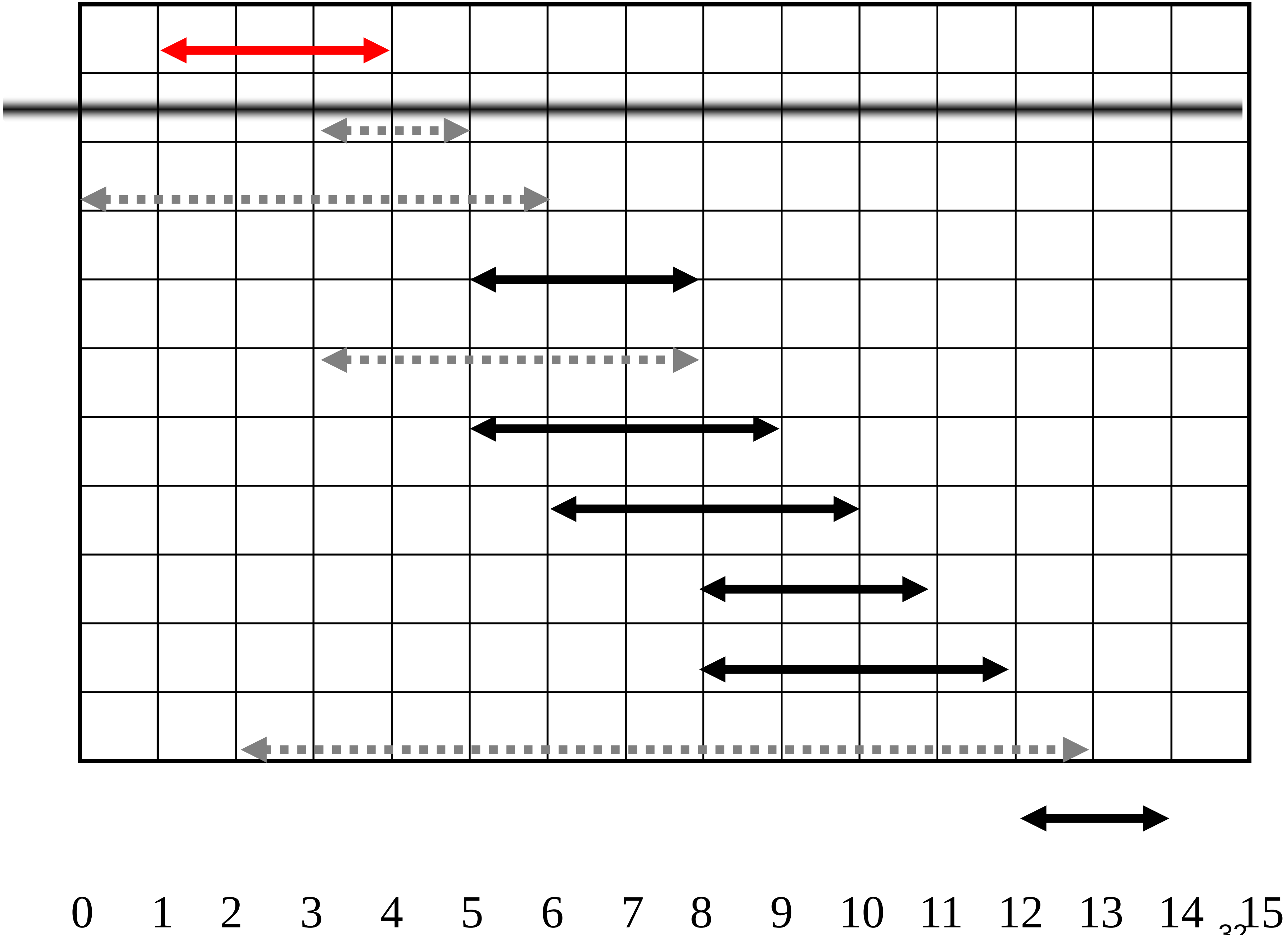
Interval Representation

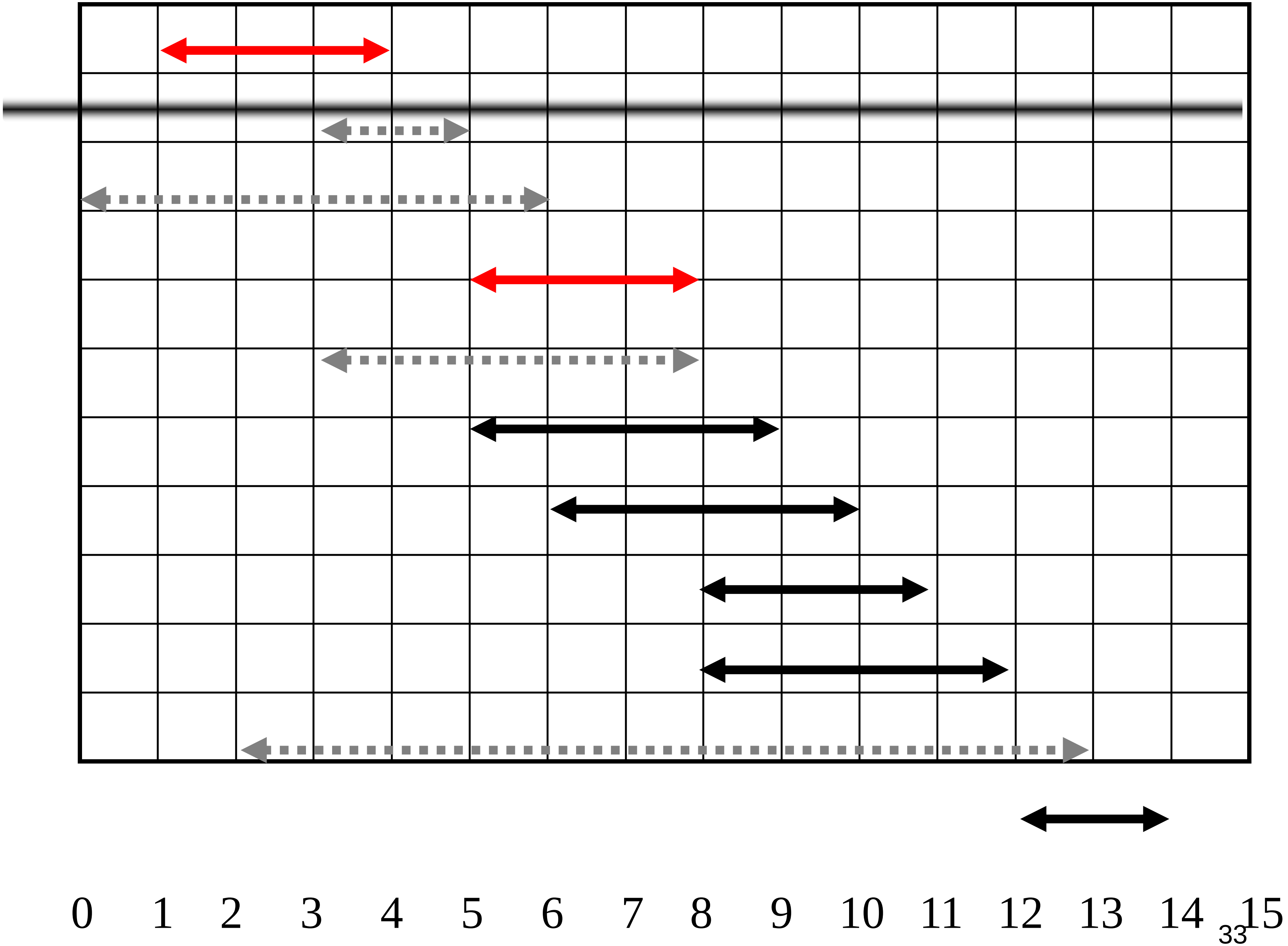


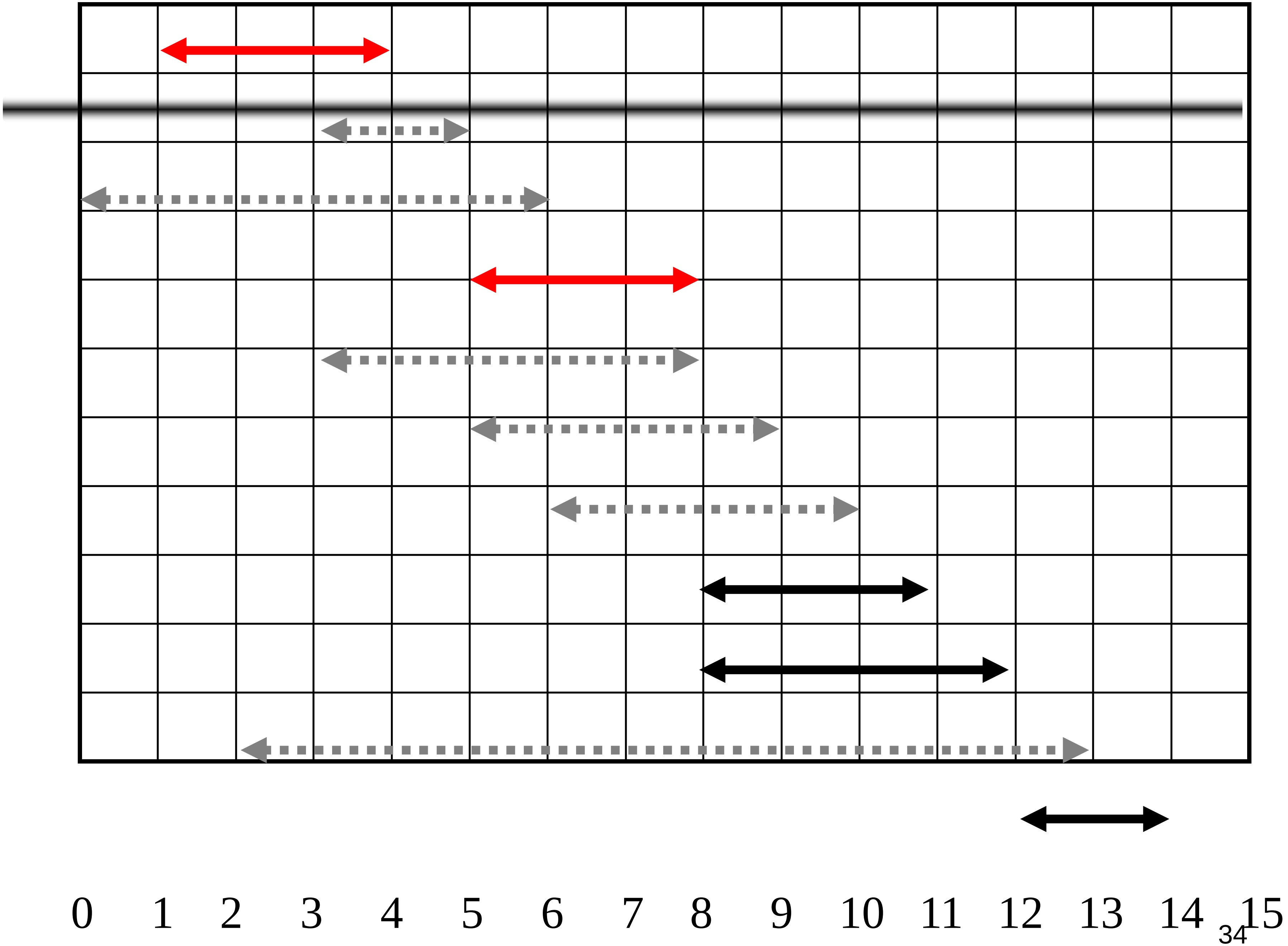
i	1	2	3	4	5	6	7	8	9	10	11
s_i	1	3	0	5	3	5	6	8	8	2	12
f_i	4	5	6	7	8	9	10	11	12	13	14

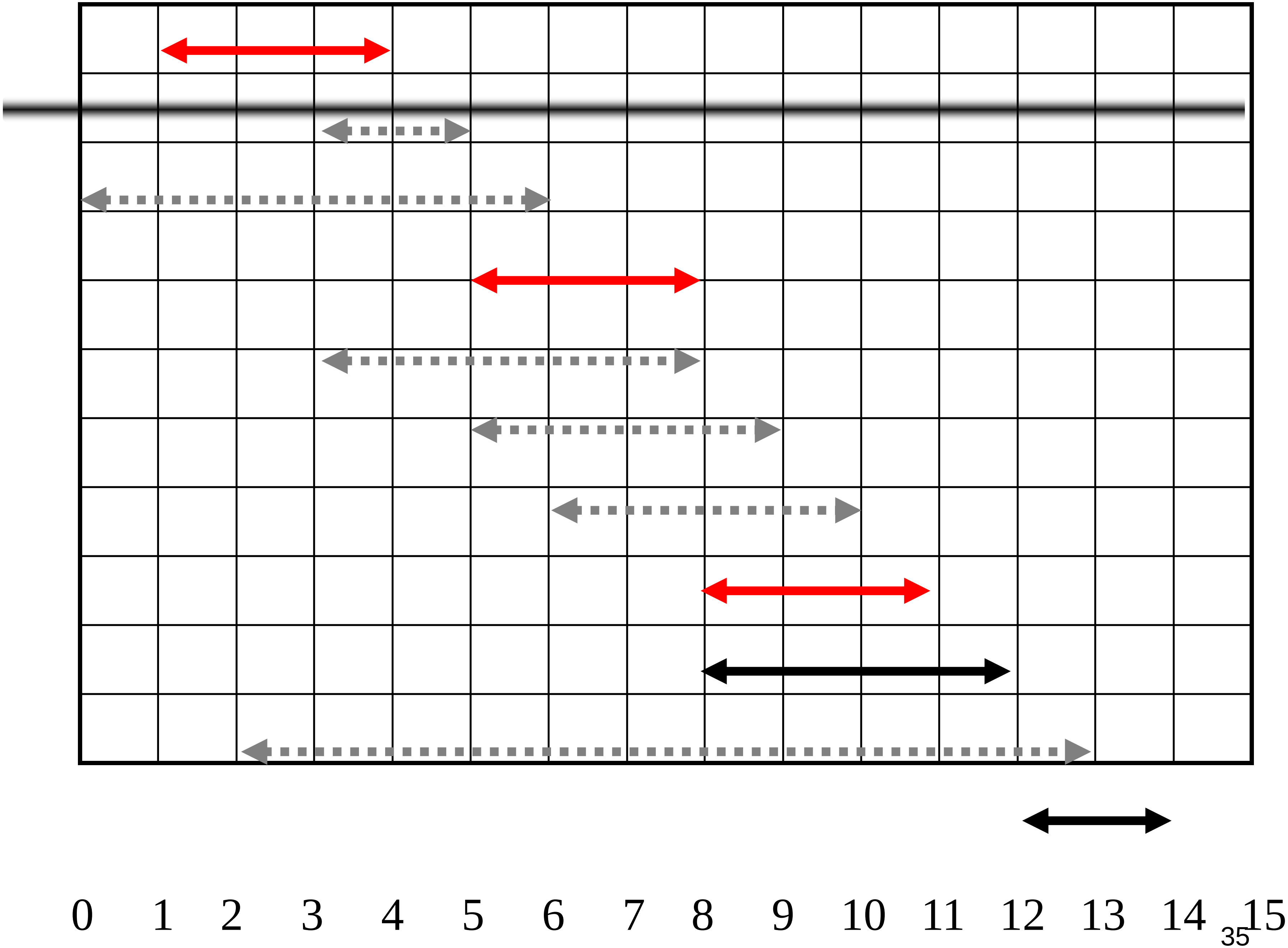


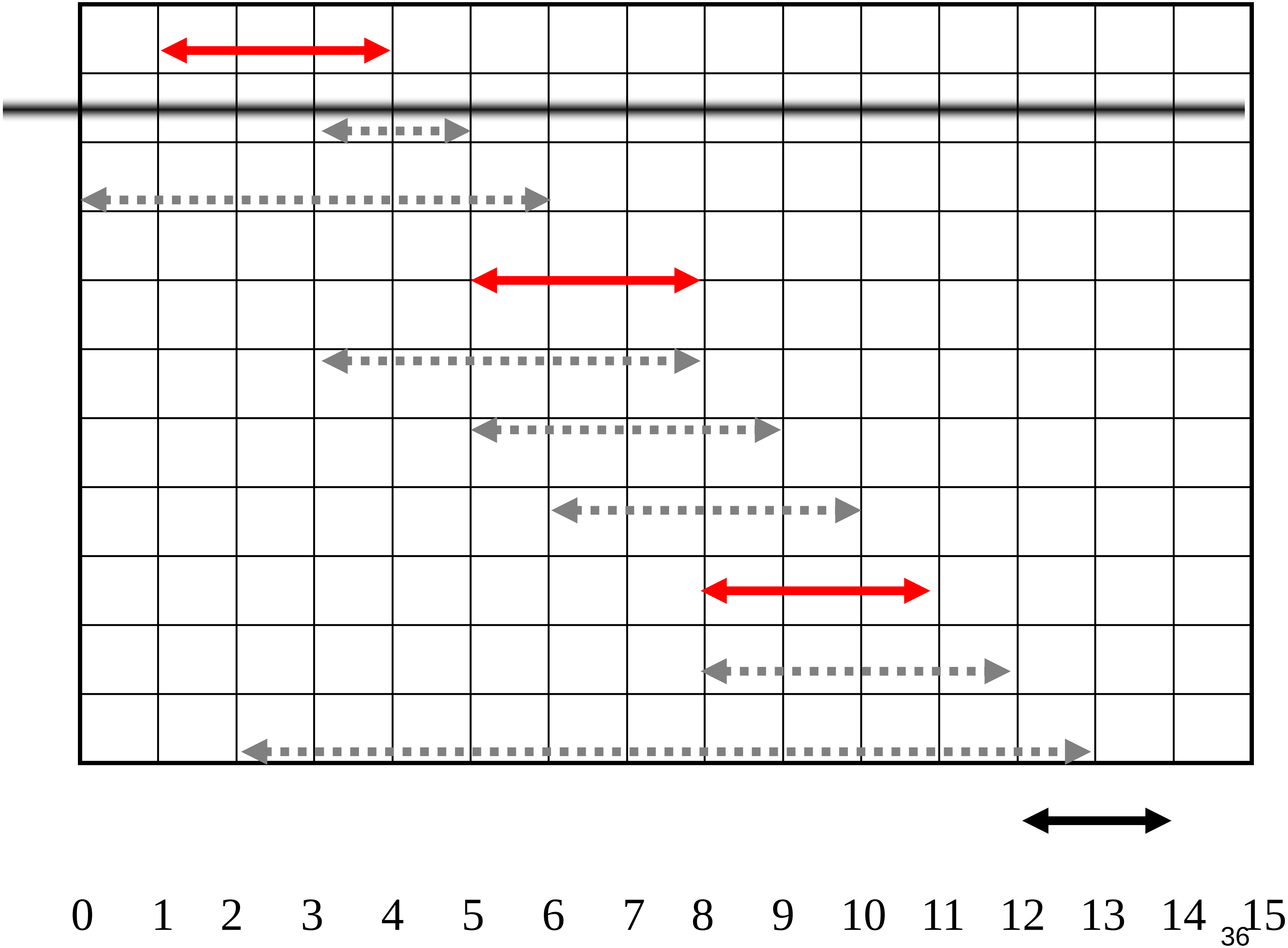

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

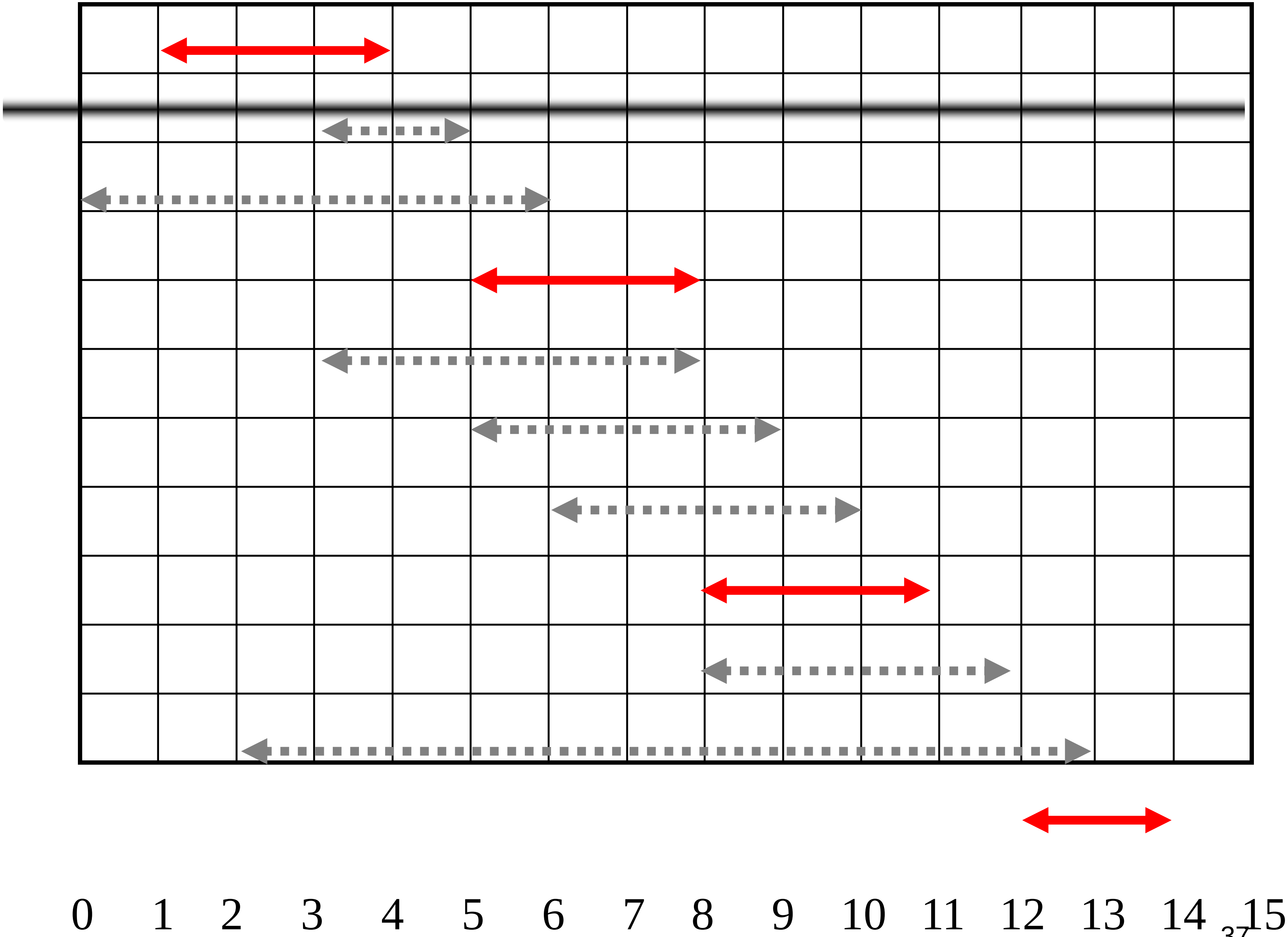

0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15


0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15


0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15


0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15


0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15


0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15