# 活动安排问题

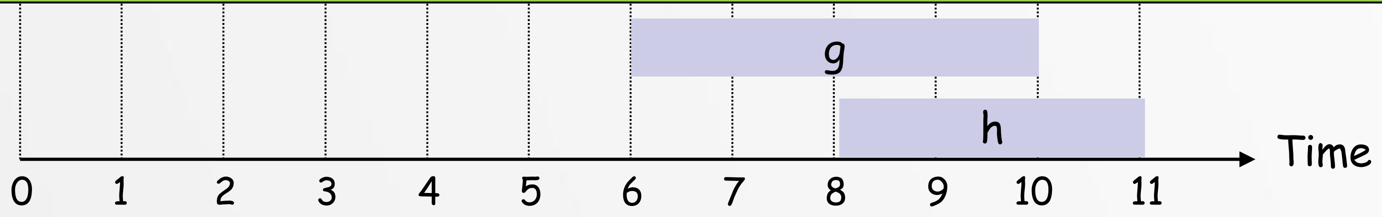
# 活动相容性

设有n个活动的集合E={1,2,…,n}，其中每个活动都要求使用同一资源，如演讲会场等，而在同一时间内只有一个活动能使用这一资源。每个活动i都有一个要求使用该资源的起始时间si和一个结束时间fi,且si <fi 。如果选择了活动i，则它在半开时间区间[si, fi)内占用资源。若区间[si, fi)与区间[sj, fj)不相交，则称活动i与活动j是相容的。也就是说，当si≥fj或sj≥fi时，活动i与活动j相容。

# 问题描述

- **活动安排问题.**
  - **有n个活动**
  - **活动 j 在 $s_j$,时刻开始， $f_j$时刻结束.**

**活动安排问题就是要在所给的活动集合中选出最大的相容活动子集合，是可以用贪心算法有效求解的很好例子。该问题要求高效地安排一系列争用某一公共资源的活动。贪心算法提供了一个简单、漂亮的方法使得尽可能多的活动能兼容地使用公共资源。**



**在下面所给出的解活动安排问题的贪心算法greedySelector :**

```
public static int greedySelector(int [] s, int [] f, boolean a[])
  {
     int n=s.length-1;
     a[1]=true;
     int j=1;
     int count=1;
     for (int i=2;i<=n;i++) {
       if (s[i]>=f[j]) {
         a[i]=true;
         j=i;
        count++;
        }
        else a[i]=false;
       }
     return count;
  }
```

**各活动的起始时间和结束时间存储于数组s和f中且按结束时间的非减序排列**
**$f_1<=f_2<=f_3..<=f_n$**

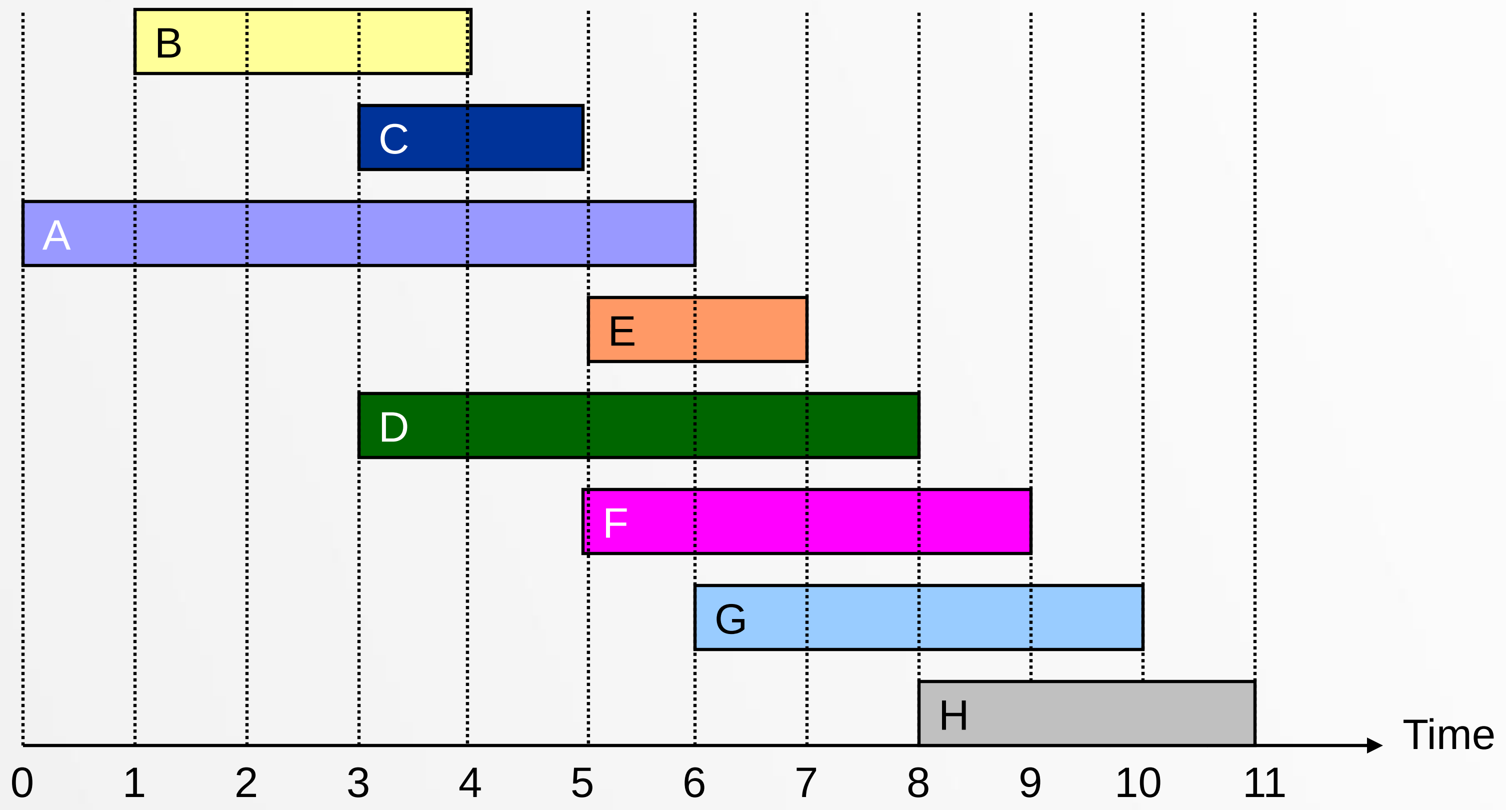
**用集合a来存储所选择的活动，活动i在集合中当且仅当a[i]=true,j记录最近一次加入a的活动**

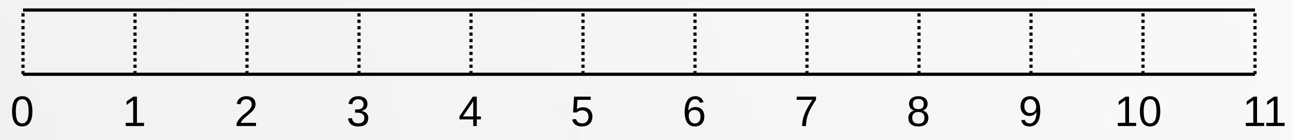
由于输入的活动以其完成时间的**非减序**排列，所以算法**greedySelector**每次总是选择**具有最早完成时间**的相容活动加入集合A中。直观上，按这种方法选择相容活动为未安排活动留下尽可能多的时间。也就是说，该算法的贪心选择的意义是**使剩余的可安排时间段极大化**，以便安排尽可能多的相容活动。

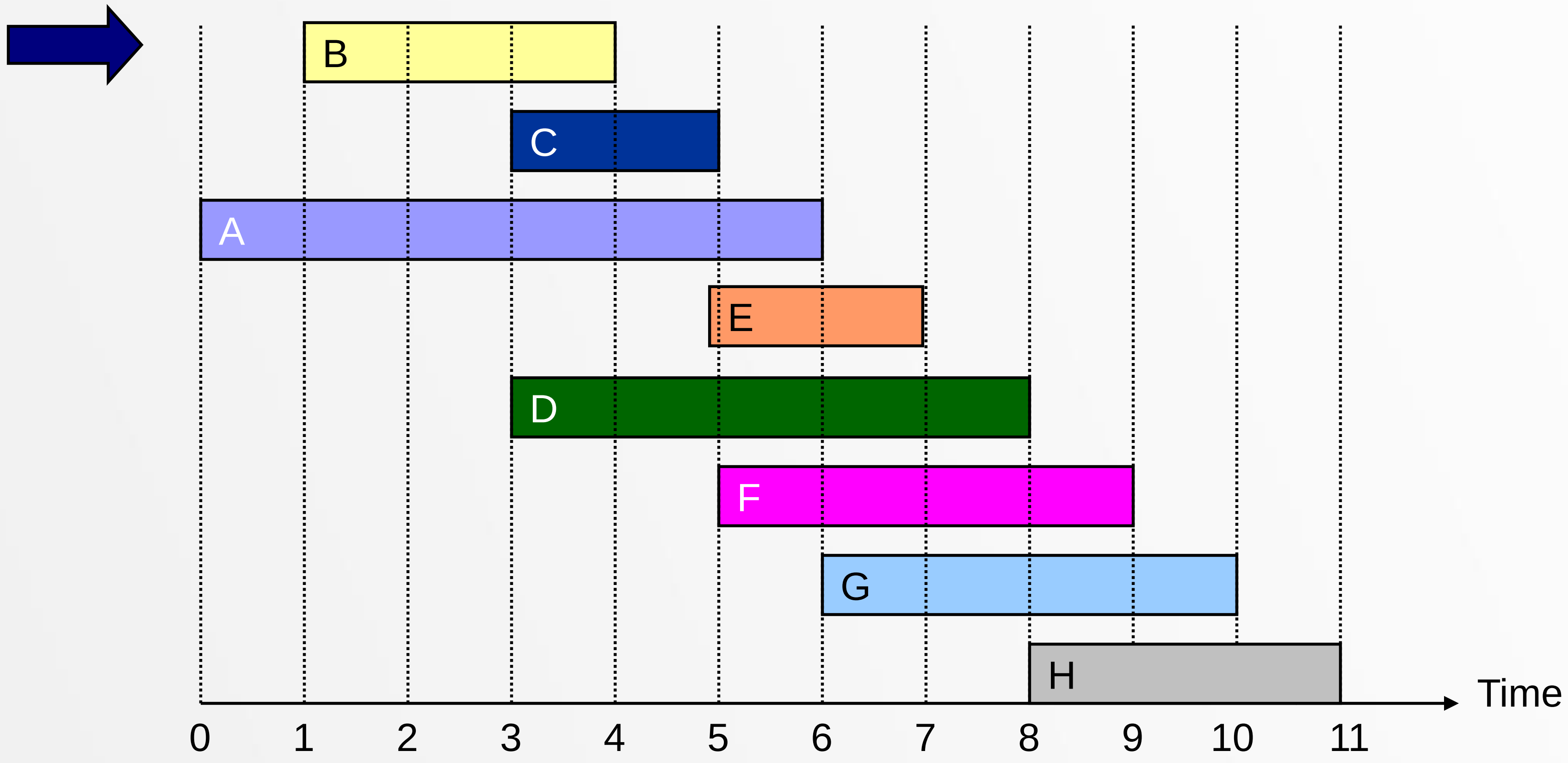
算法**greedySelector**的效率极高。当输入的活动已按结束时间的非减序排列，算法只需**O(n)**的时间安排n个活动，使最多的活动能相容地使用公共资源。如果所给出的活动未按非减序排列，可以用**O(nlogn)**的时间重排。

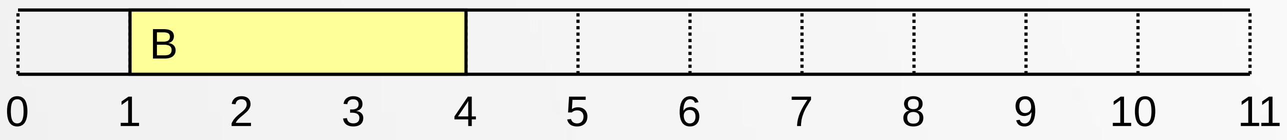
**例：** 设待安排的11个活动的开始时间和结束时间按结束时间的非减序排列如下：

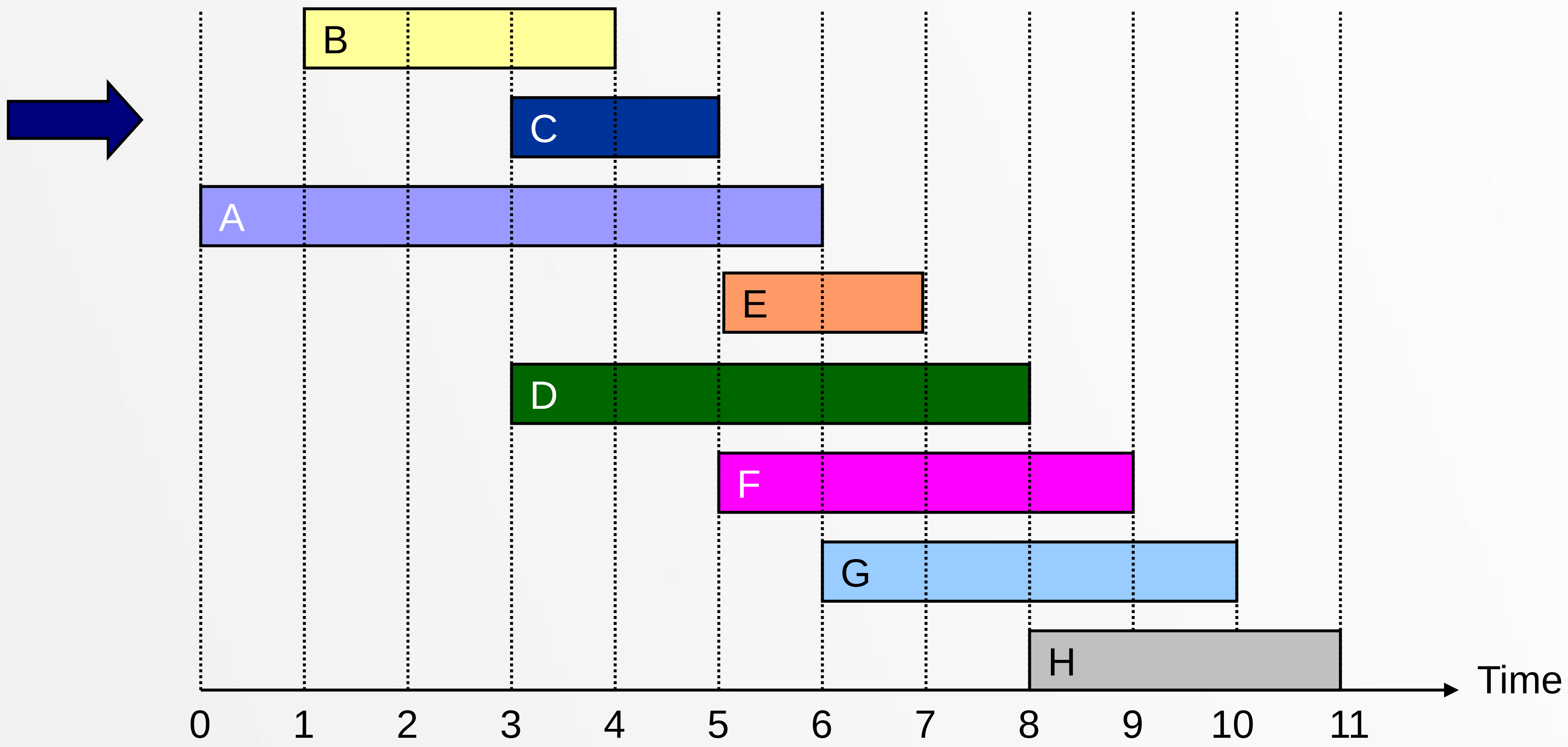
| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **S[i]** | 1 | 3 | 0 | 5 | 3 | 5 | 6 | 8 | 8 | 2 | 12 |
| **f[i]** | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |

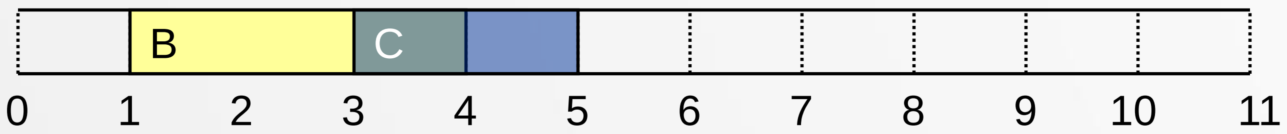

B
C
A
E
D
F
G
H
Time
0 1 2 3 4 5 6 7 8 9 10 11

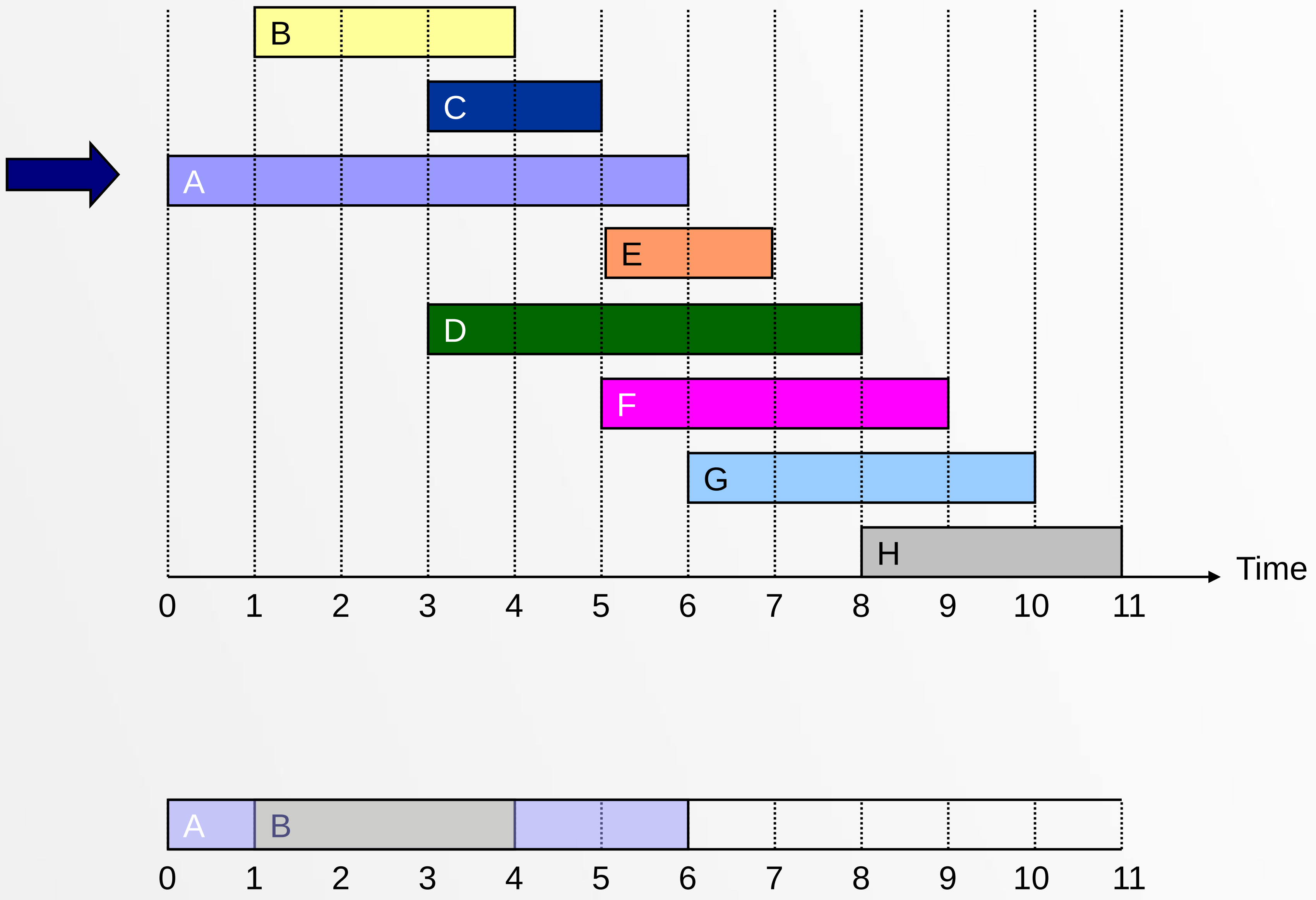

0 1 2 3 4 5 6 7 8 9 10 11

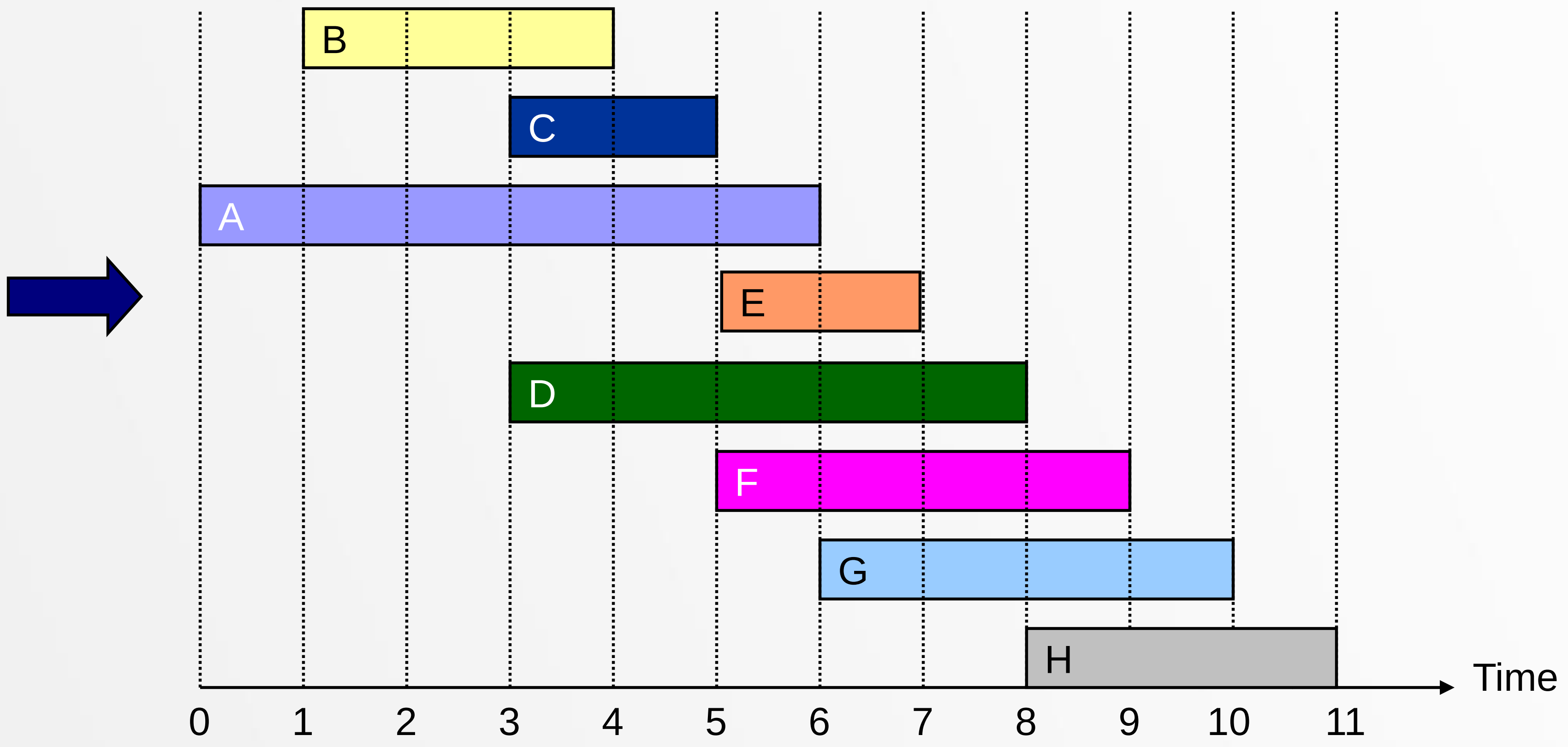

B
C
A
E
D
F
G
H
Time
0 1 2 3 4 5 6 7 8 9 10 11

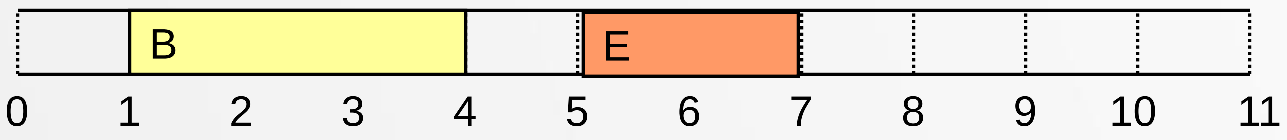

B
0 1 2 3 4 5 6 7 8 9 10 11

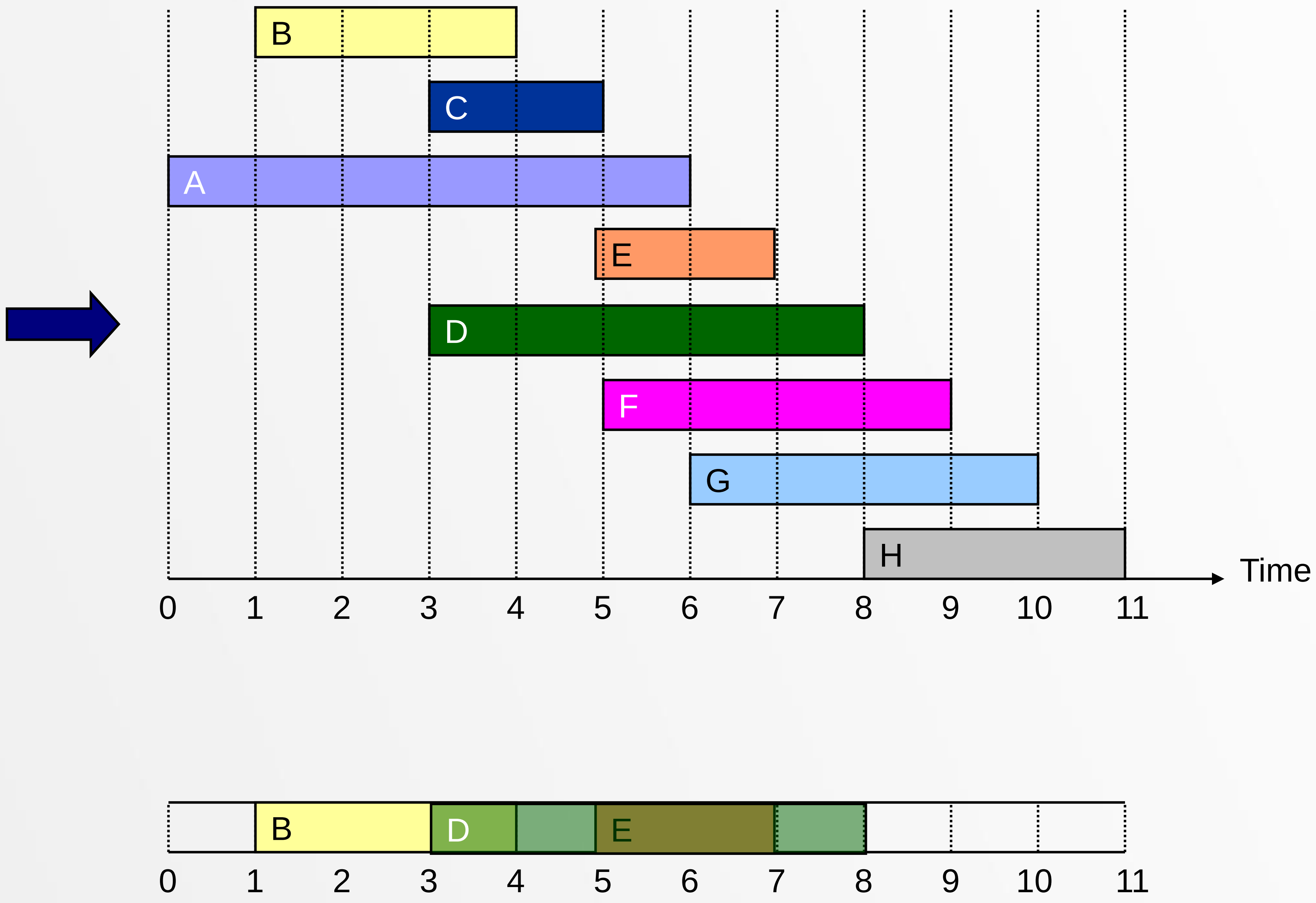

B
C
A
E
D
F
G
H
Time
0
1
2
3
4
5
6
7
8
9
10
11

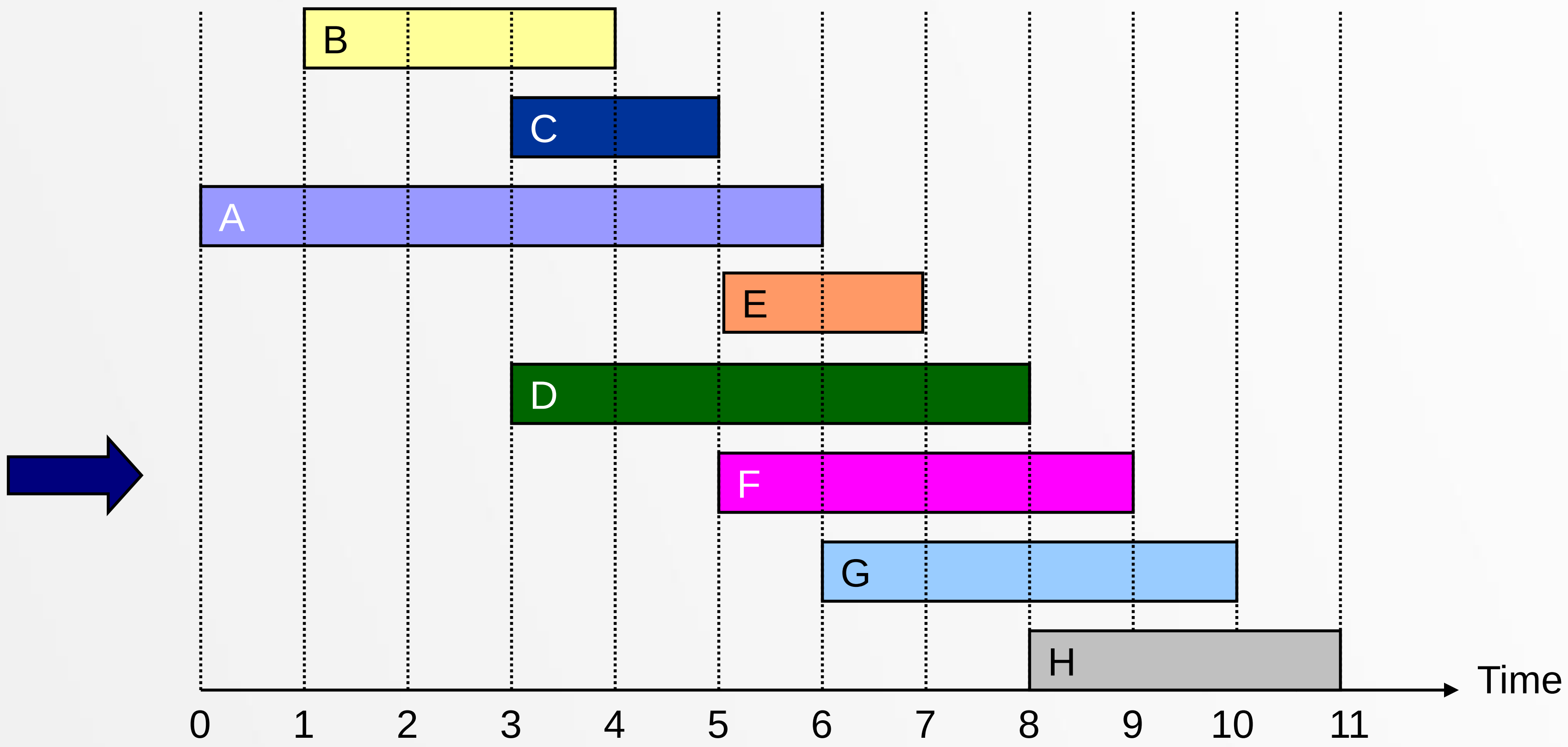

B
C
0
1
2
3
4
5
6
7
8
9
10
11

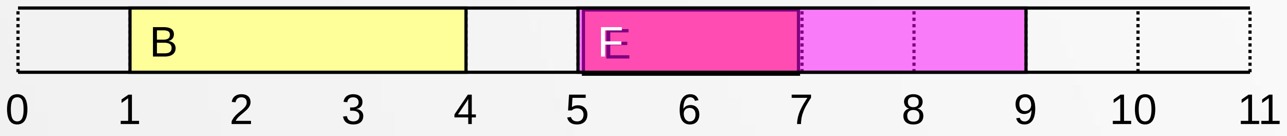

B
C
A
E
D
F
G
H
Time
0 1 2 3 4 5 6 7 8 9 10 11
A
B
0 1 2 3 4 5 6 7 8 9 10 11

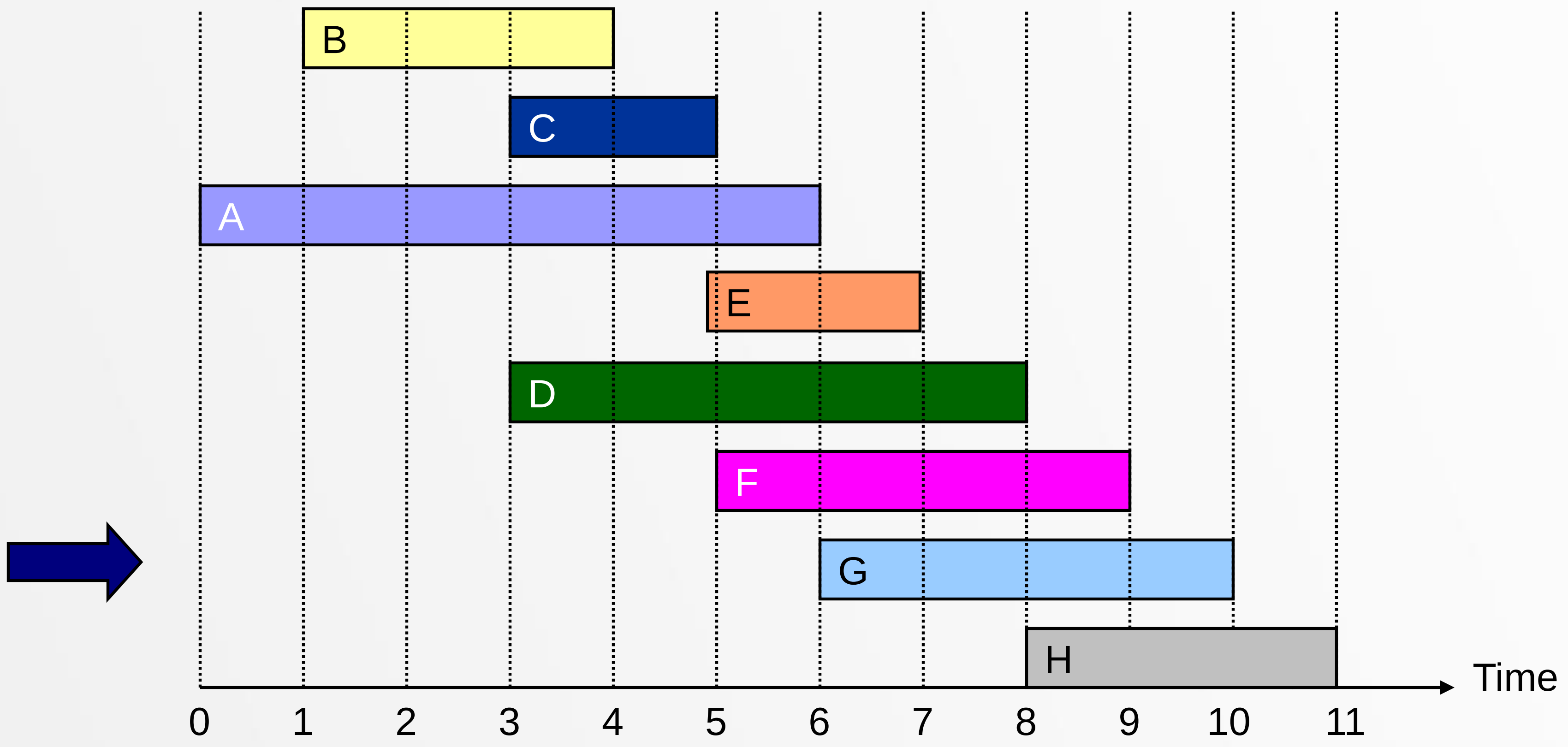

B
C
A
E
D
F
G
H
Time
0
1
2
3
4
5
6
7
8
9
10
11

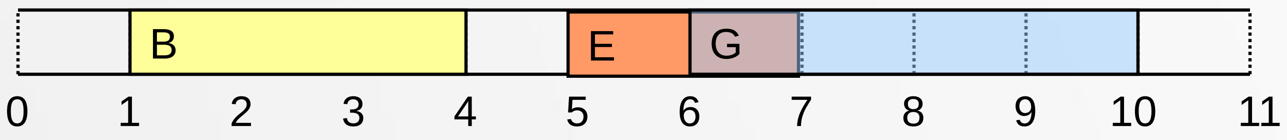

B
E
0
1
2
3
4
5
6
7
8
9
10
11

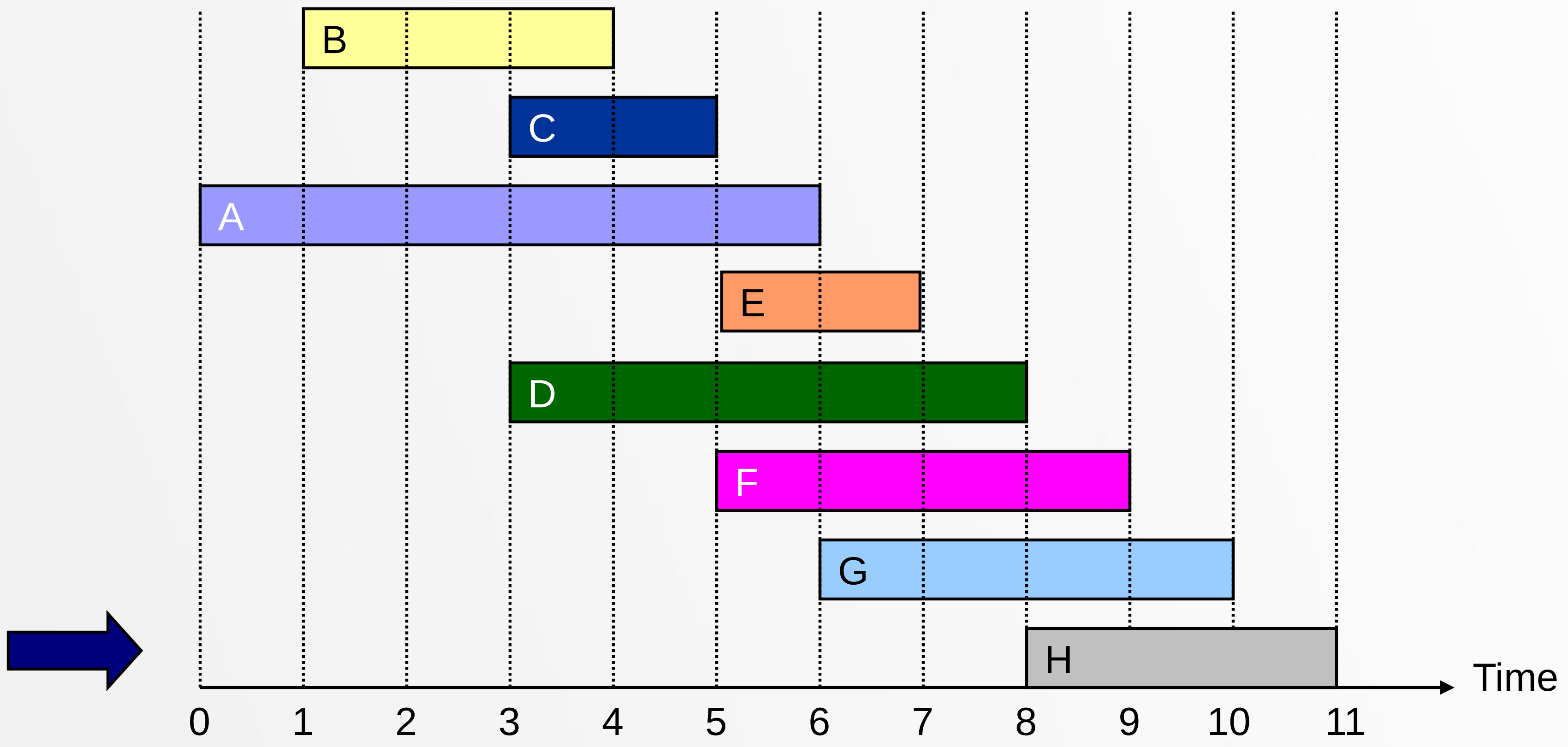

B
C
A
E
D
F
G
H
Time
0
1
2
3
4
5
6
7
8
9
10
11
B
D
E
0
1
2
3
4
5
6
7
8
9
10
11

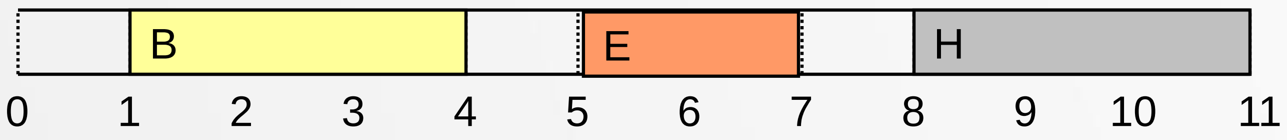

B
C
A
E
D
F
G
H
Time
0
1
2
3
4
5
6
7
8
9
10
11


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


B
C
A
E
D
F
G
H
Time
0 1 2 3 4 5 6 7 8 9 10 11


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


B
C
A
E
D
F
G
H
Time
0
1
2
3
4
5
6
7
8
9
10
11


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

贪心算法并不总能求得问题的**整体最优解**。但对于活动安排问题，贪心算法greedySelector却总能求得的整体最优解，即它最终所确定的相容活动集合A的规模最大。

**证明：** 设E={1,2,...n}活动集合（数学归纳推出矛盾）

由于E中活动安排结束时间的非减序排列，活动1具有最早的完成时间。

1）**证明活动安排问题的最优解以贪心选择开始，包含活动1**

设 是所给活动安排问题的一个最优解，A中活动按结束时间非减排序，A中第一个活动是活动k

a.若k=1则得证

b.若k>1，则设 $B = A - \{K\} \cup \{1\}$ 。由于f1≤fk，且A中活动是相容的，故B中活动也是相容的。由于A与B中活动数目相同，故A与B都是最优的活动安排。因此总存在以贪心选择开始的最优活动安排 。

1）**证明活动安排问题的最优解与贪心选择解一致**

原问题简化为对E中所有活动与活动1相容的活动进行活动安排的子问题。

若A是原问题的最优解，则A’=A-｛1｝是活动安排问题E’=｛i∈E，si≥f1}的最优解。若能找到E’的一个解B’，B’包含比A’更多的活动，则将活动1加入B’，将产生E的一个解B，它包含比A更多的活动，这与A是原问题的最优解矛盾，因此A’是活动安排问题E’=｛i∈E，si≥f1}的最优解。因此每一步所做出的贪心选择都将问题简化为一个更小的与原问题具有相同形式的子问题。以此对贪心算法选择次数用数学归纳法，则知贪心算法将产生原问题的最优解。