1. 根据List-Scheduling算法求解如下负载平衡问题。

List-Scheduling $(m, n, t_1, \ldots, t_n)$

For $i = 1$ to m

$L[i] = 0$.

$S[i] \leftarrow \emptyset$.

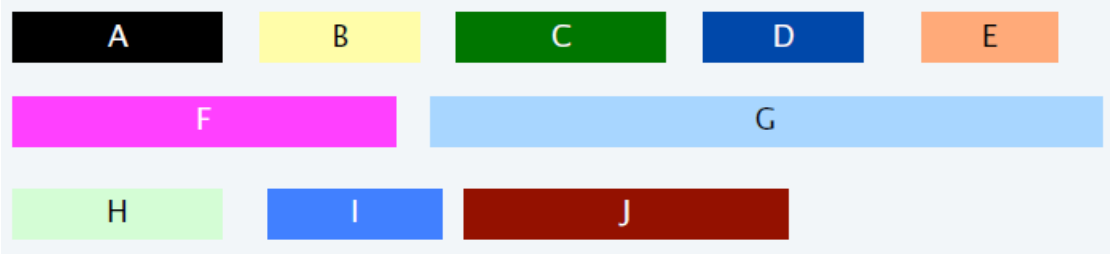
For $j = 1$ to n

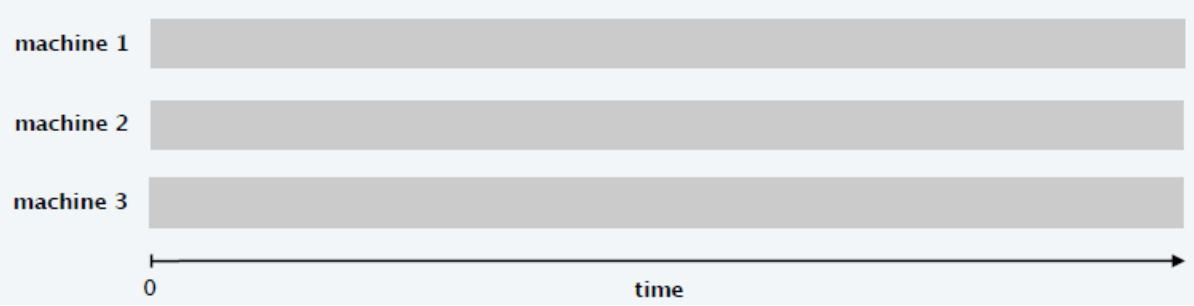
$i \leftarrow argmin_k L[k]$.

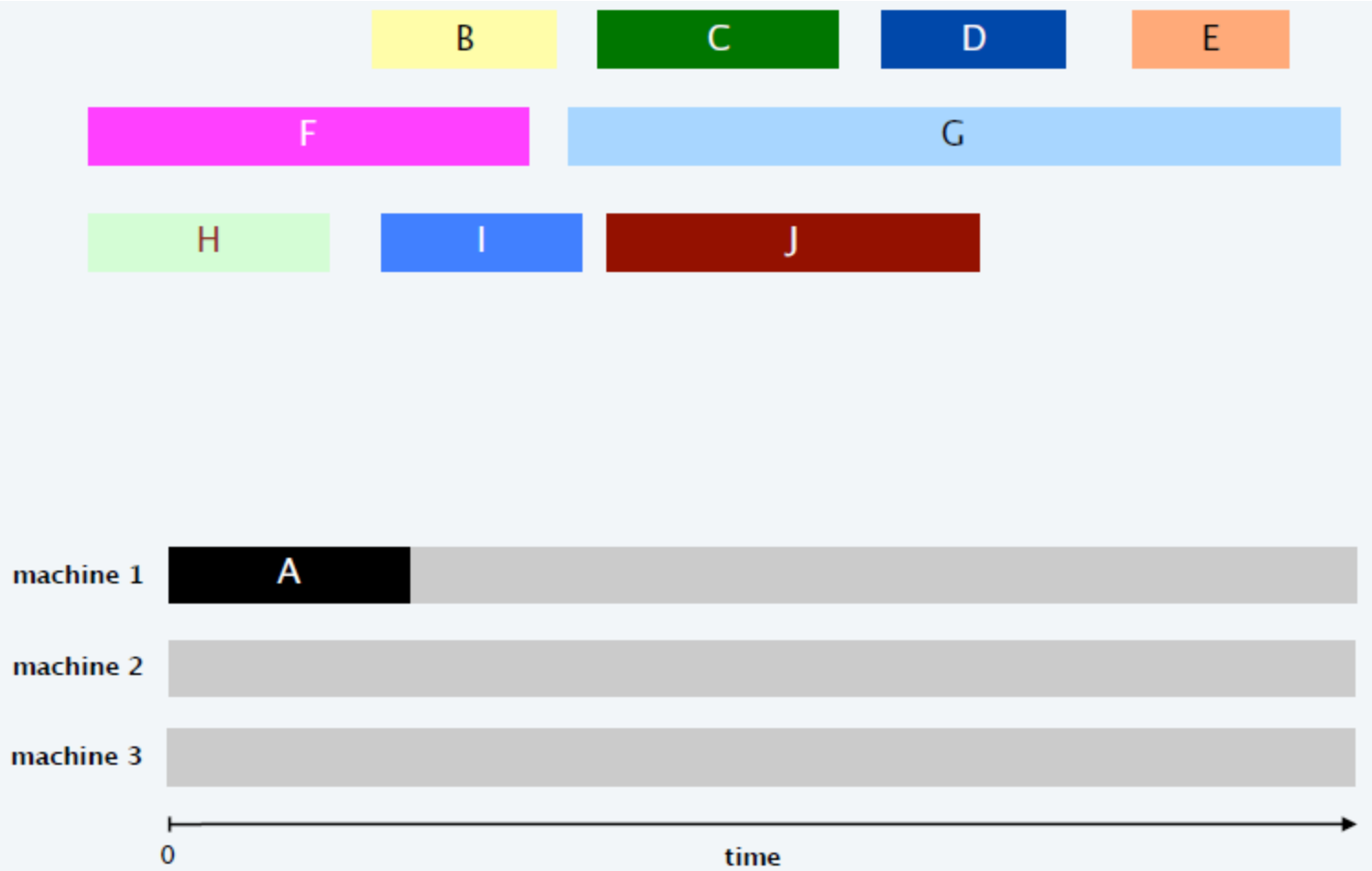
$S[i] \leftarrow S[i] \cup \{j\}$.

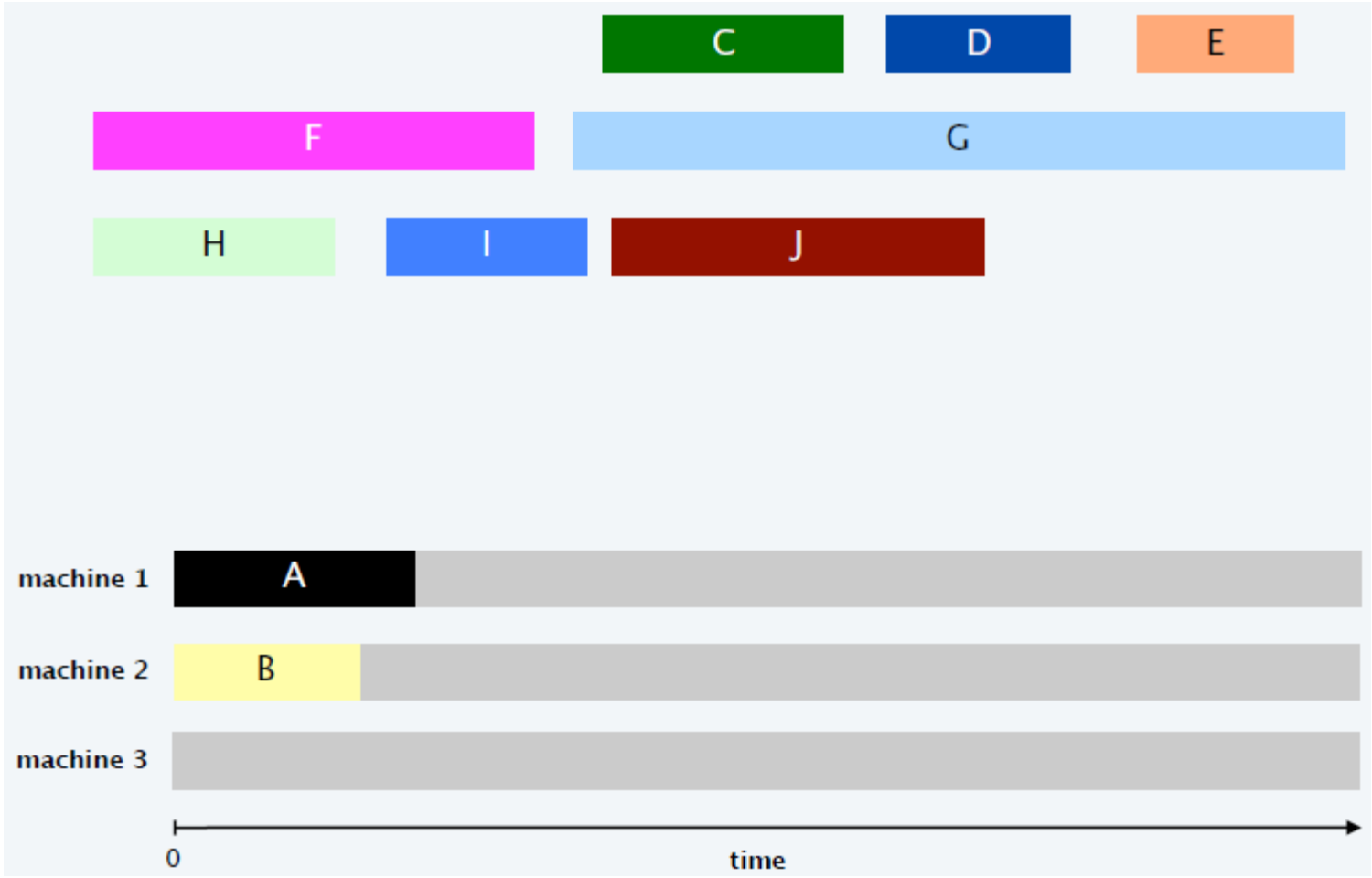
$L[i] \leftarrow L[i] + t_j$.

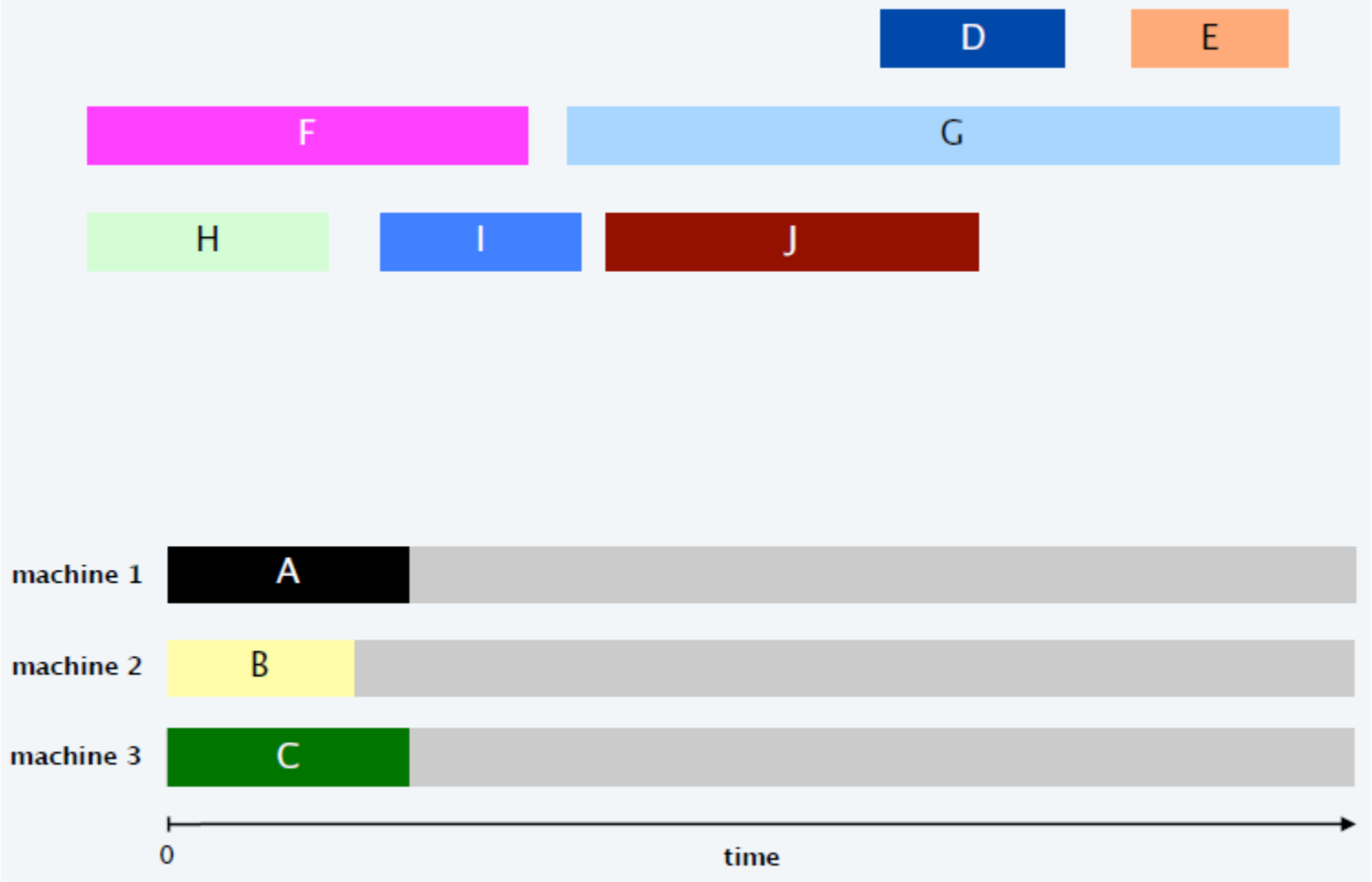
Return $S[1], S[2], \ldots, S[m]$.

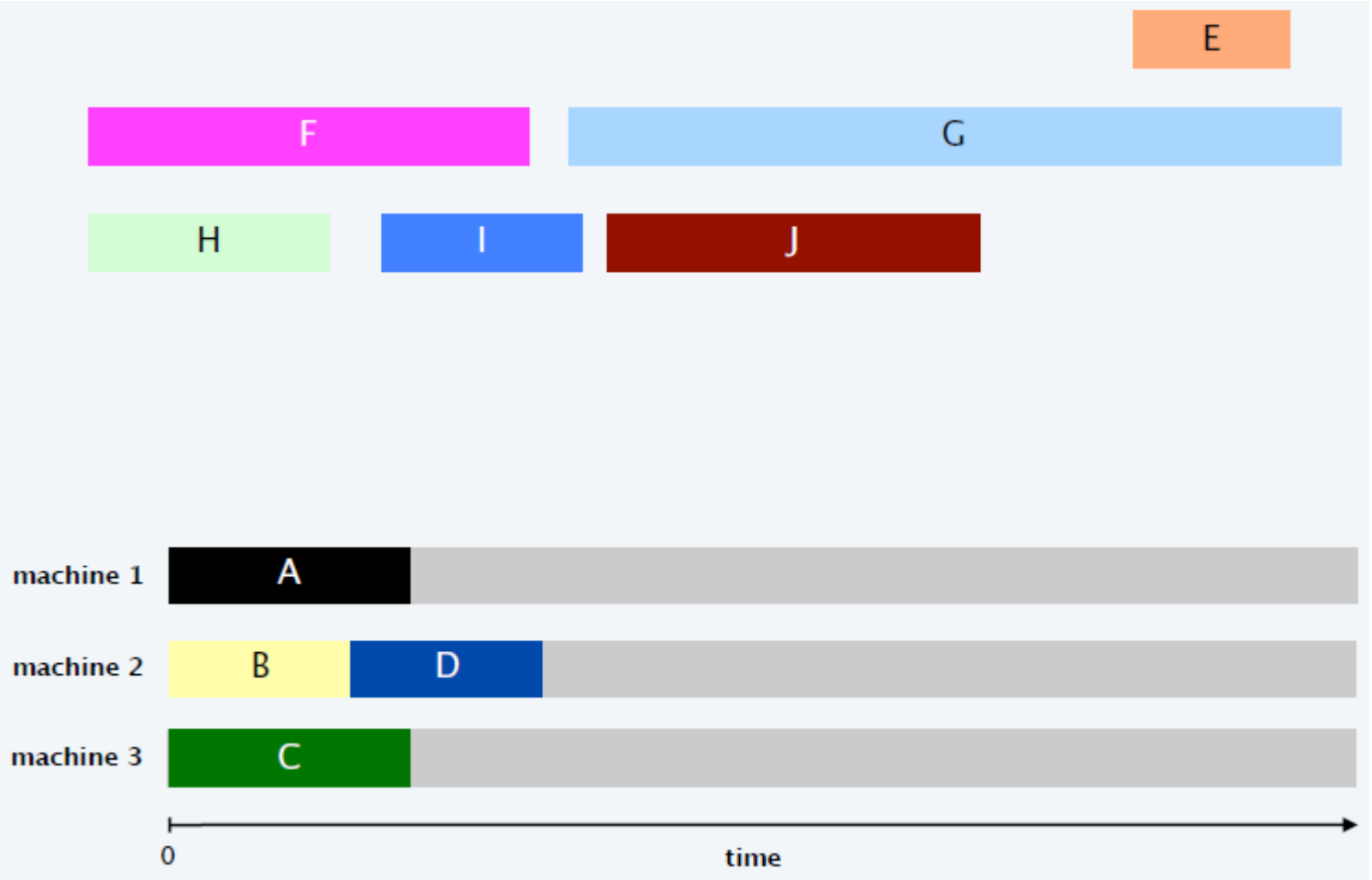


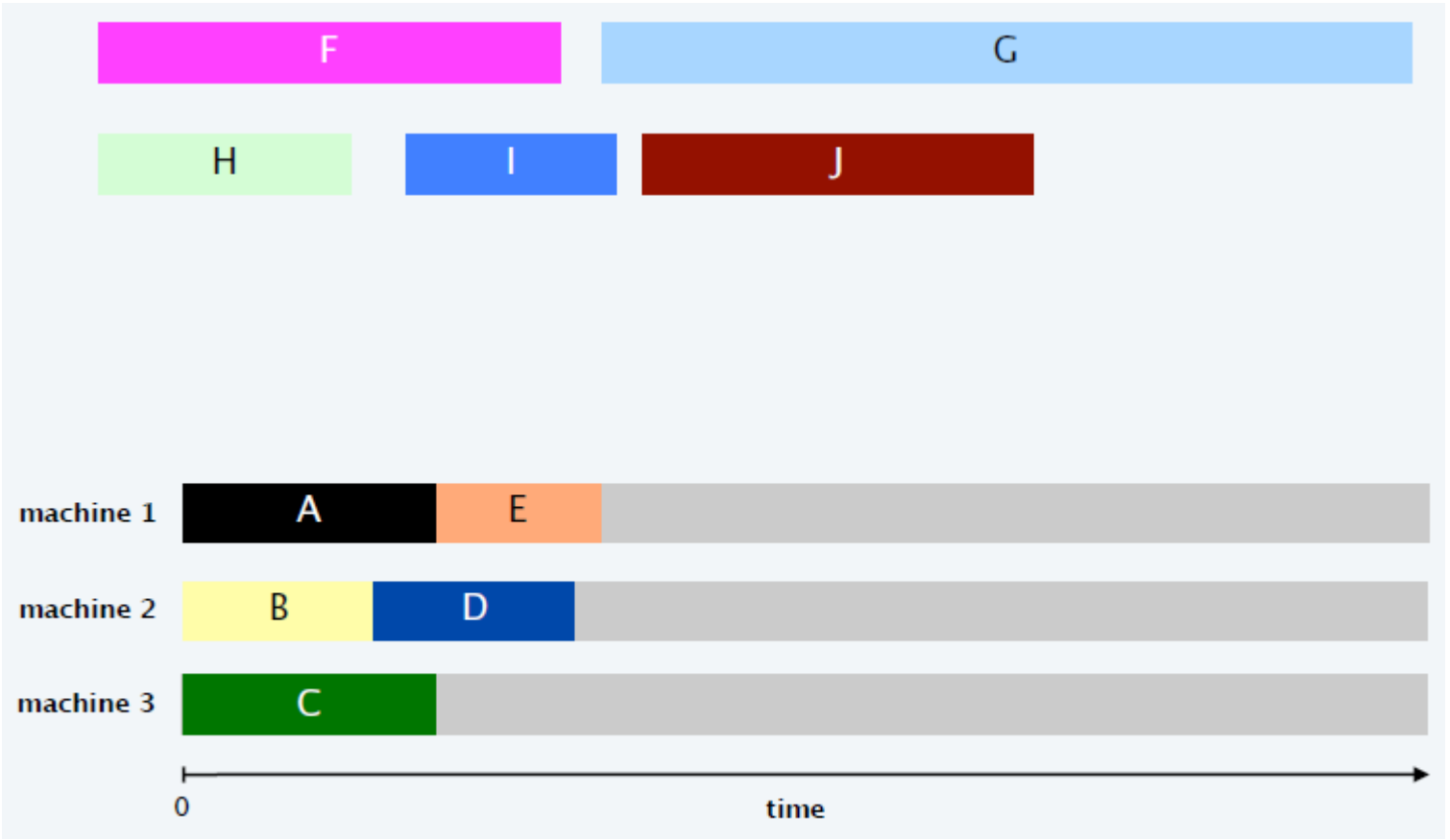


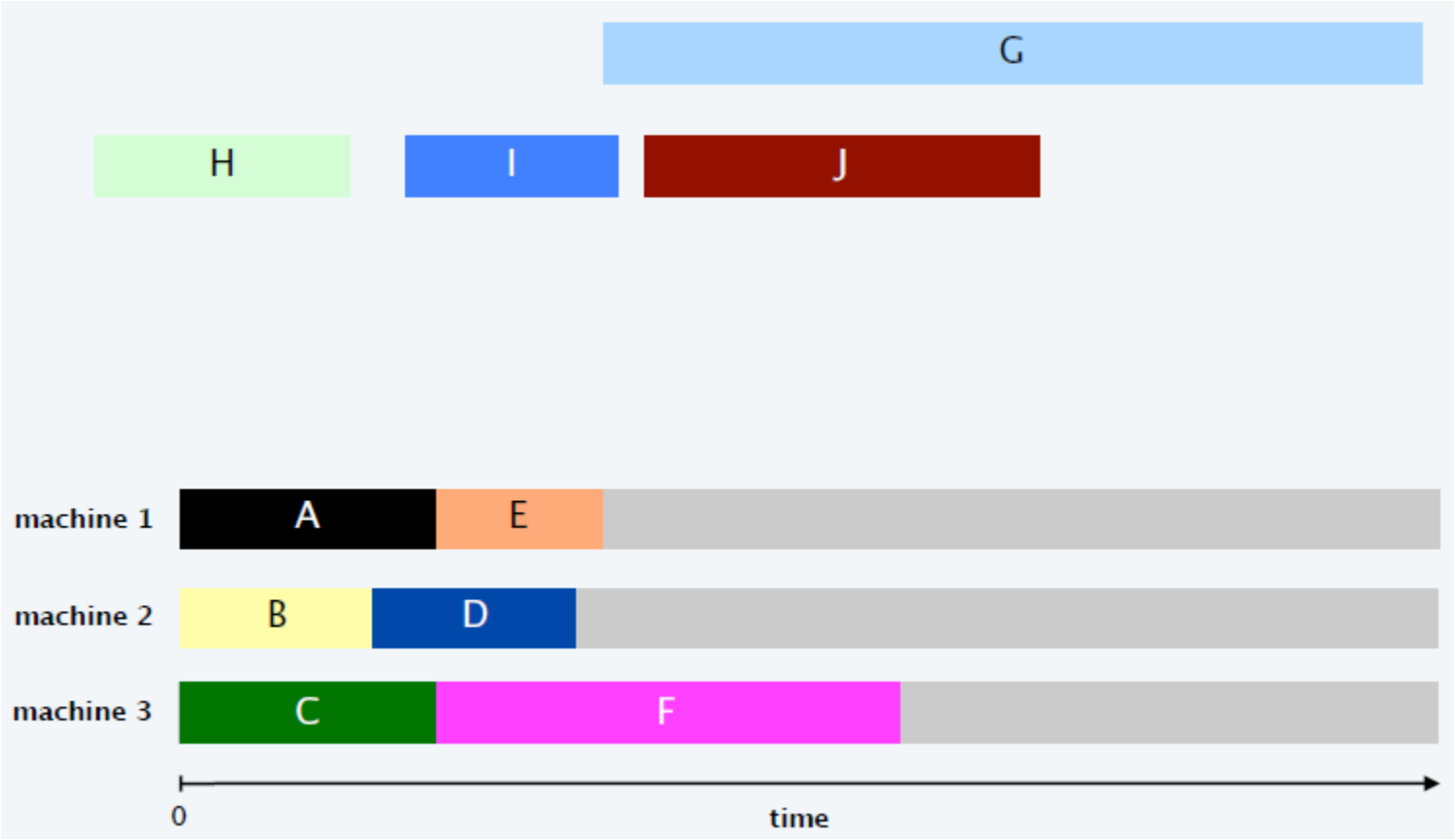

B
C
D
E
F
G
H
I
J
machine 1
A
machine 2
machine 3
0
time

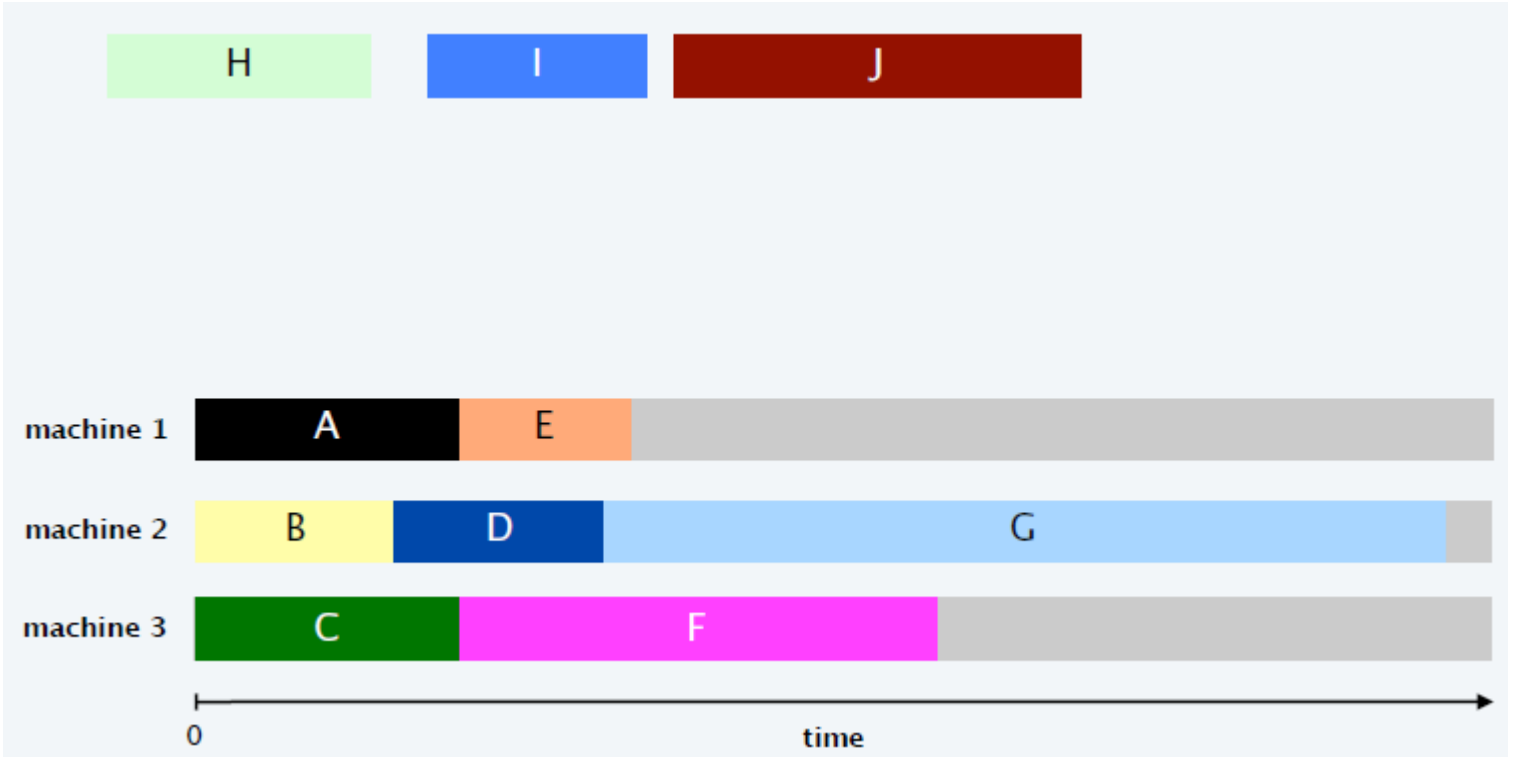

C
D
E
F
G
H
I
J
machine 1
A
machine 2
B
machine 3
0
time

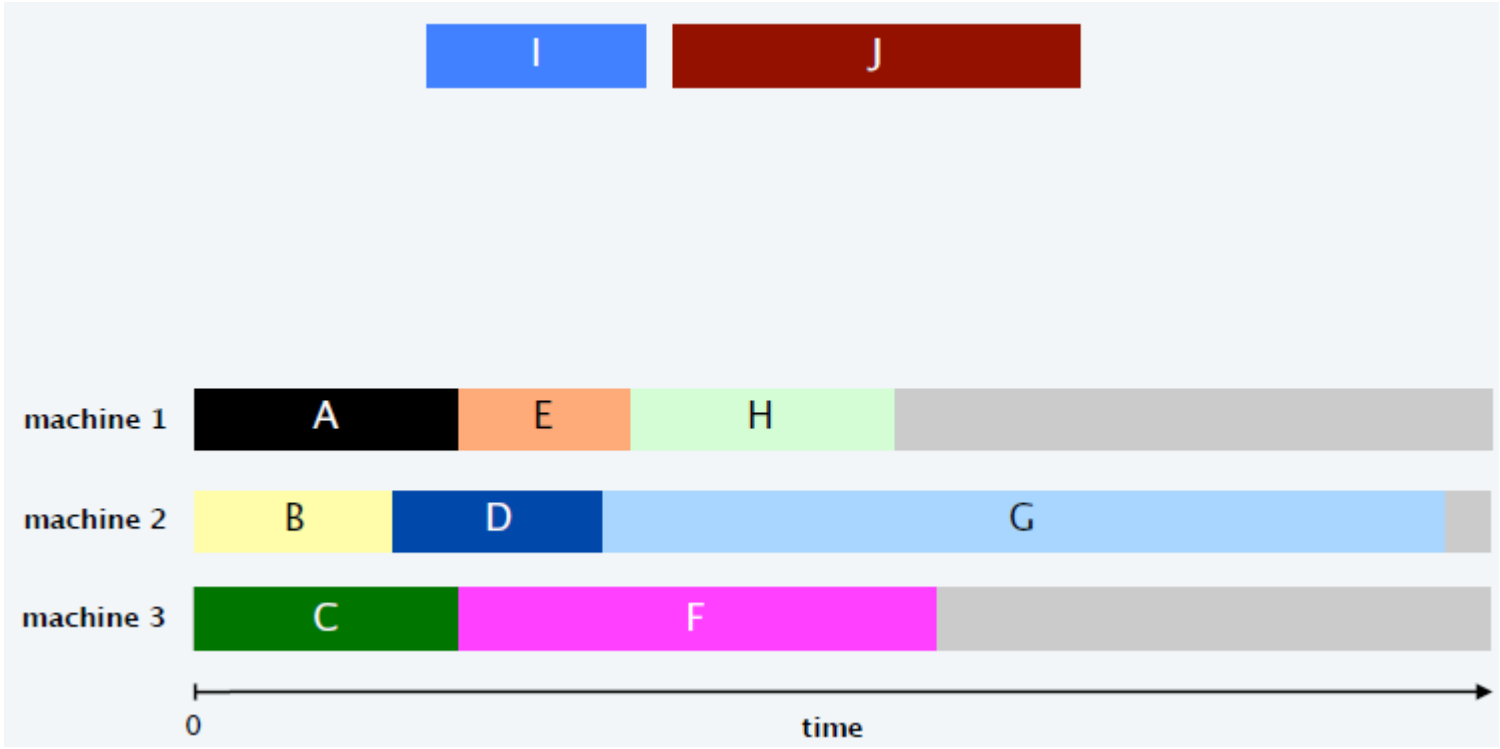

D
E
F
G
H
I
J
machine 1
A
machine 2
B
machine 3
C
0
time

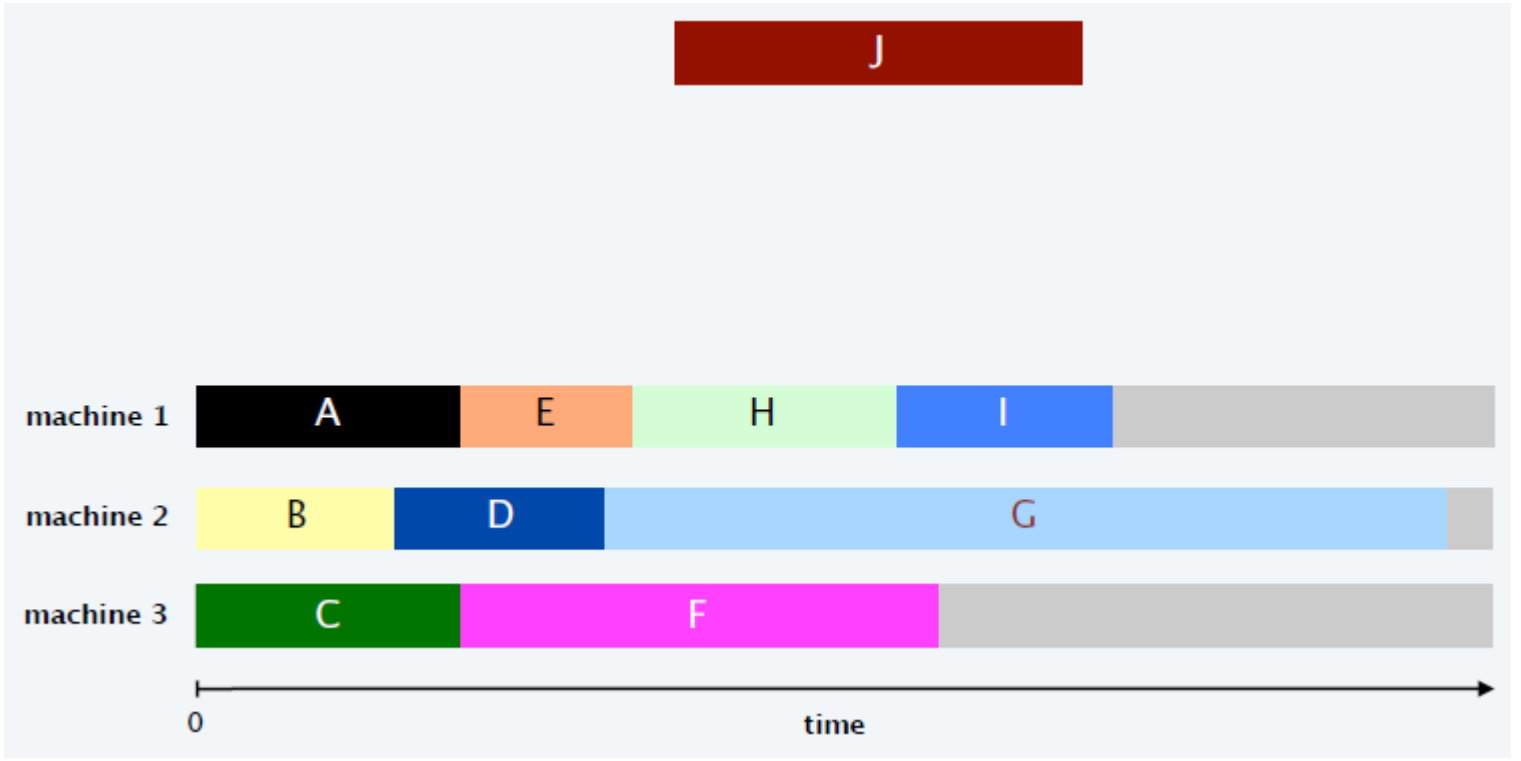

E
F
G
H
I
J
machine 1
A
machine 2
B
D
machine 3
C
0
time

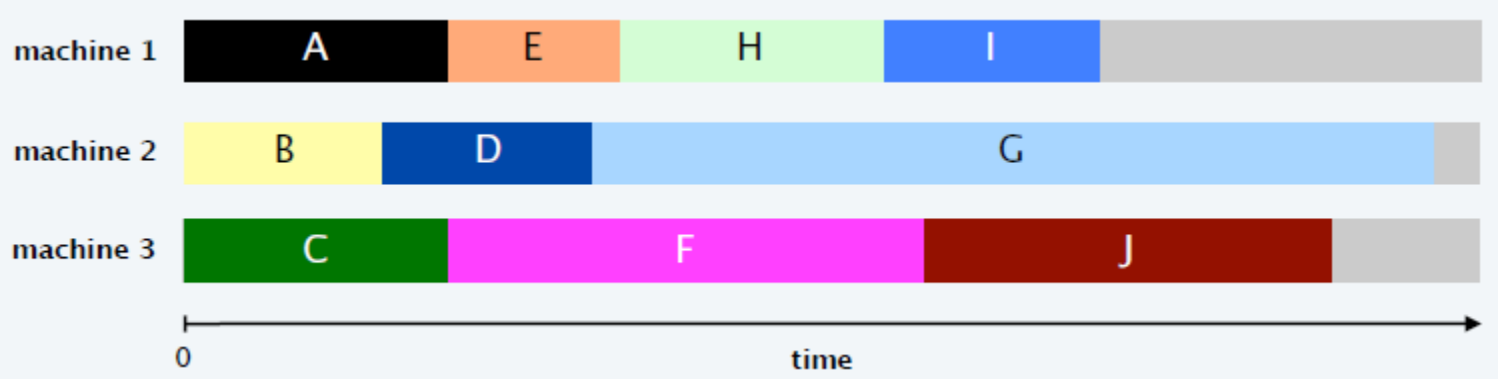

F
G
H
I
J
machine 1
A
E
machine 2
B
D
machine 3
C
0
time

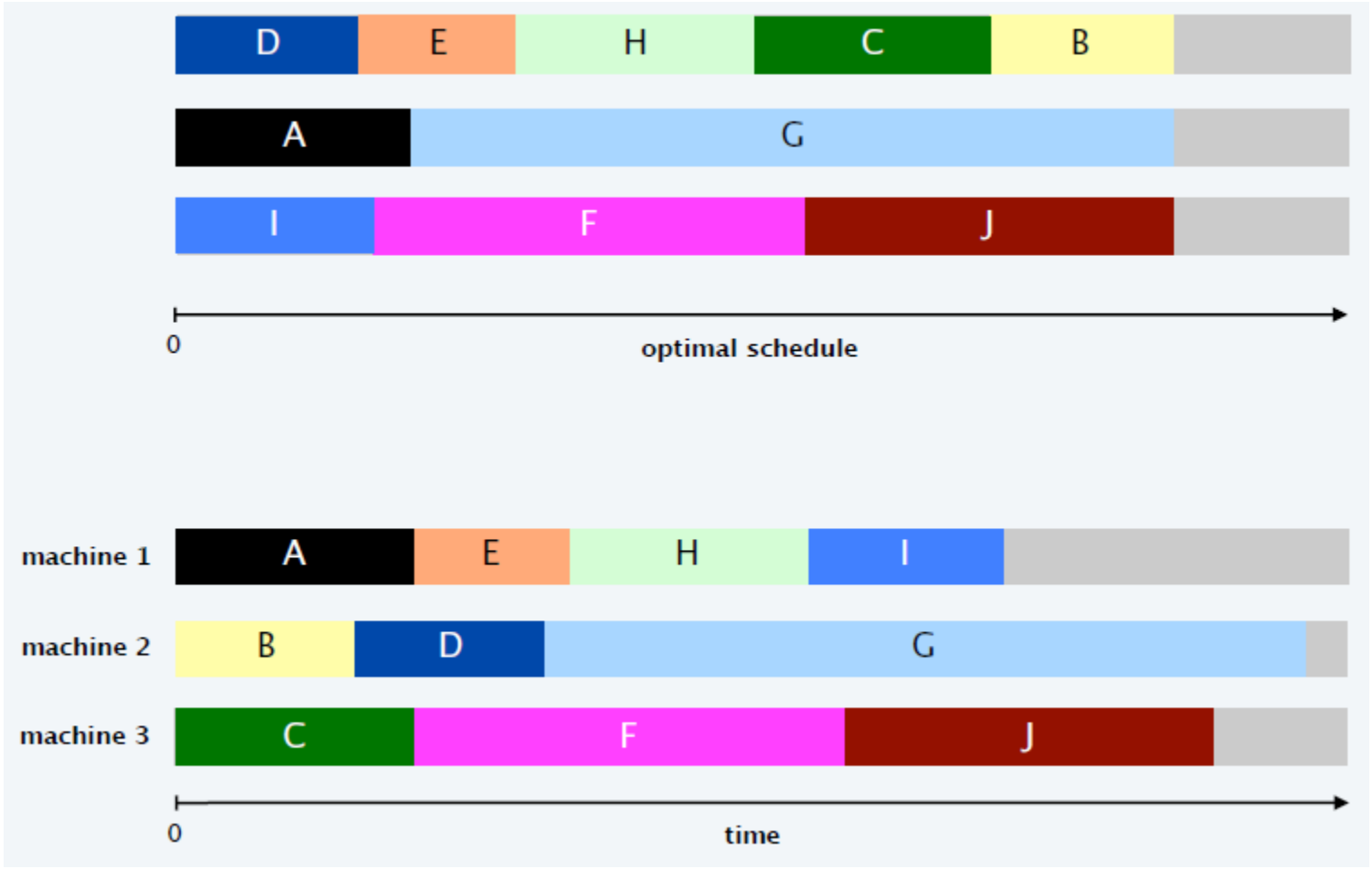

G
H
I
J
machine 1
A
E
machine 2
B
D
machine 3
C
F
0
time


H
I
J
machine 1
A
E
machine 2
B
D
G
machine 3
C
F
0
time

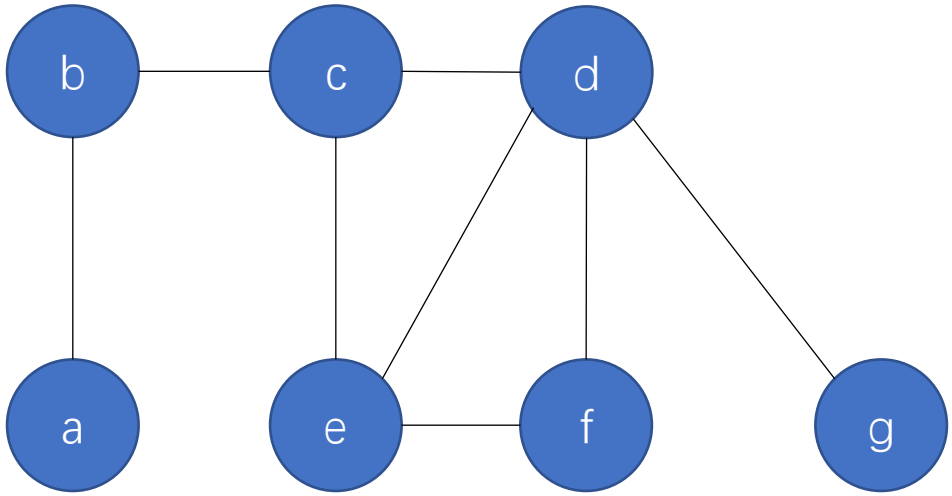

I
J
machine 1
A
E
H
machine 2
B
D
G
machine 3
C
F
0
time


J
machine 1
A
E
H
I
machine 2
B
D
G
machine 3
C
F
0
time

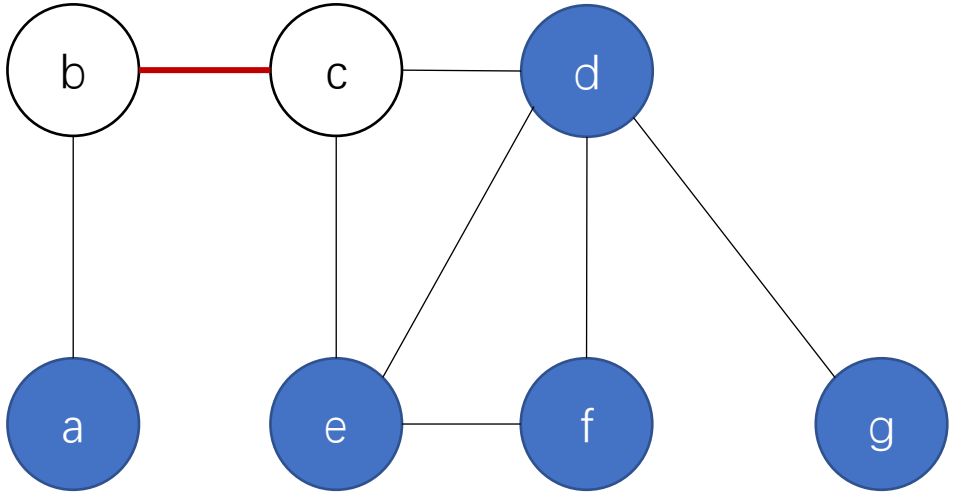

machine 1
A
E
H
I
machine 2
B
D
G
machine 3
C
F
J
0
time


D
E
H
C
B
A
G
I
F
J
0
optimal schedule
machine 1
A
E
H
I
machine 2
B
D
G
machine 3
C
F
J
0
time

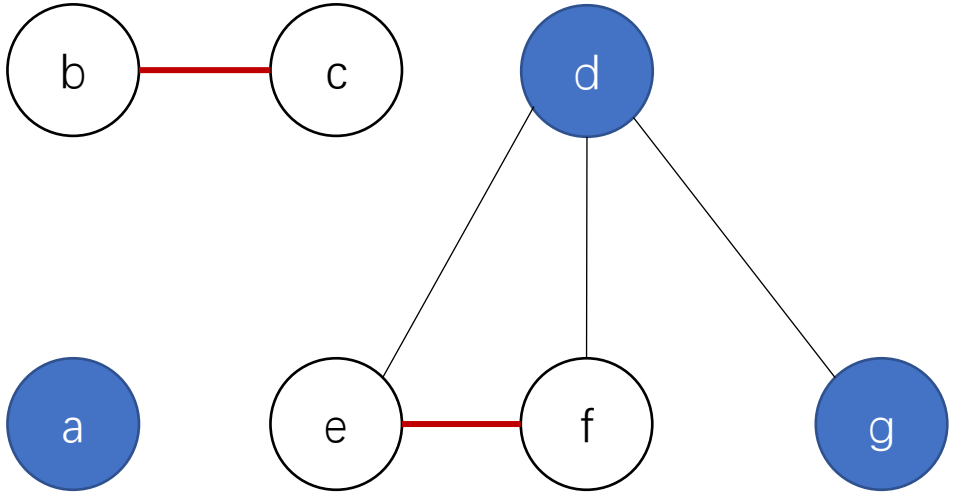
2. 根据VCOVER-APPROX算法找出下图的一个顶点覆盖。



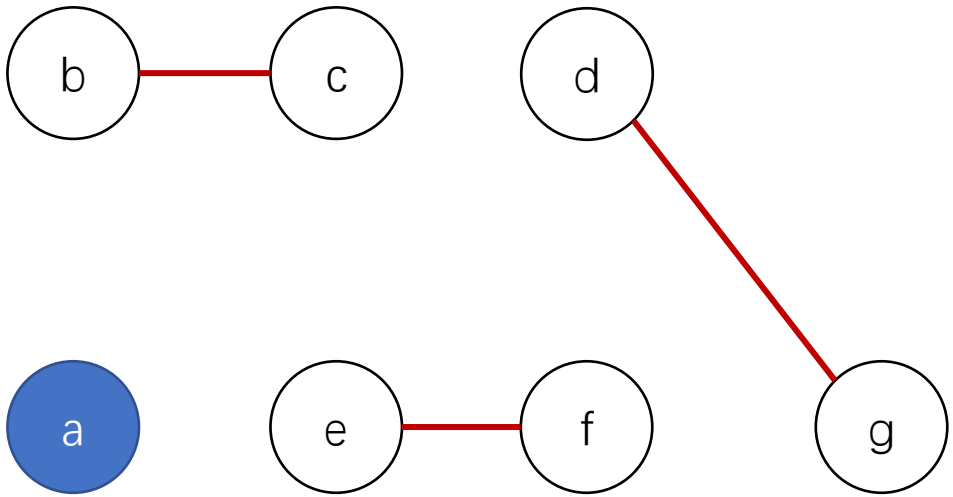
VCOVER-APPROX算法
输入： 无向图 G=(V,E)
输出：G的一个顶点覆盖C
1. C ← { }
2. while E ≠ { }
3. 设 e=(u,v)为E中的任意边
4. C ← C ∪ {u,v}
5. 删除 e 和 E中所有与u和v相关联的边
6. end while



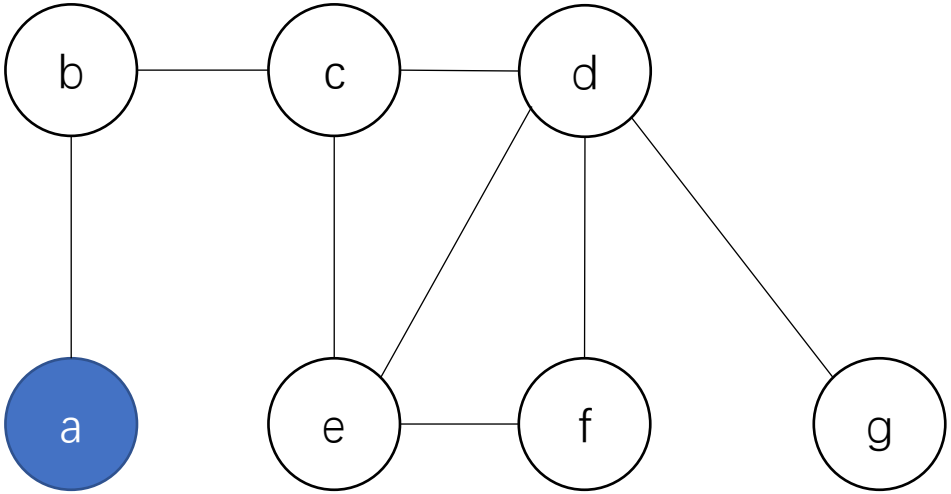
选择(b, c)边，并删去b、c所连接的边



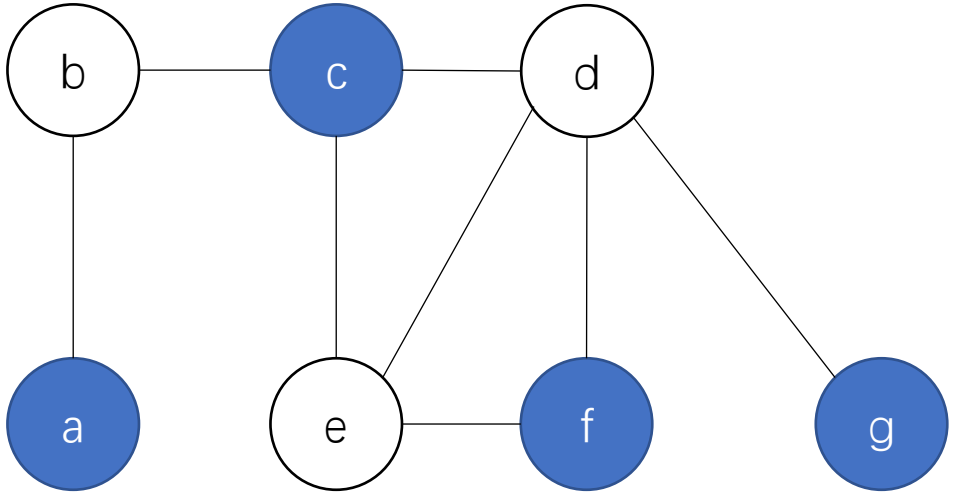
选择(e,f)边，并删去e、f所连接的边



选择(d, g)边



产生一个覆盖：b, c, d, e, f, g



最小覆盖为b, d, e