Analysis and Design of Algorithms

$\sum_{i \in J} p_i$

$J = (\quad)$

n jobs → $d_i > 0$, $p_i > 0$

Job i

$p_1, p_2, p_3 \ldots p_n$

Job Sequencing with Deadlines Problem

$(4, 3, 1)$

$p_4 + p_3 + p_1$

value

Job Sequencing with Deadlines Problem

Insights

- General Example
- Job Sequence with Deadlines Problem
- Solution to the Problem
- Greedy Algorithm for Job Sequence with Deadlines Problem
- Algorithm & Time Complexity

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

Brinjal
D=3,
P=30

Spinach
D=1,
P=25

Tomato
D=2,
P=20

Sour spinach
D=1,
P=15

Potato
D=5,
P=10

DAY1	DAY2	DAY3

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

Brinjal
D=3,
P=30

Spinach
D=1,
P=25

Tomato
D=2,
P=20

Sour spinach
D=1,
P=15

Potato
D=5,
P=10

DAY2	DAY3

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

Brinjal D=3, P=30

Spinach D=1, P=25

Tomato D=2, P=20

Sour spinach D=1, P=15

Potato D=5, P=10

✓

DAY2 | DAY3

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

Brinjal D=3, P=30

Spinach D=1, P=25

✓

Tomato D=2, P=20

Sour spinach D=1, P=15

Potato D=5, P=10

DAY3

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

Brinjal
D=3,
P=30

Spinach
D=1,
P=25

Tomato
D=2,
P=20

Sour spinach
D=1,
P=15

Potato
D=5,
P=10

✓ ✗

DAY3

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

Brinjal
D=3,
P=30

Spinach
D=1,
P=25

✓

Tomato
D=2,
P=20

Sour spinach
D=1,
P=15

Potato
D=5,
P=10

DAY3

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

Brinjal
D=2,
P=30

Spinach
D=1,
P=25

Tomato
D=2,
P=2(

Sour spinach
D=1,
P=15

Potato
D=5,
P=10

✓ ✓

DAY3

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

Brinjal
D=2,
P=30

Spinach
D=1,
P=25

✓

Tomato
D=2,
P=20

✓

Sour spinach
D=1,
P=15

Potato
D=5,
P=10

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

Brinjal
D=2,
P=30

Spinach
D=1,
P=25
✓

Tomato
D=2,
P=20
✓

Sour spinach
D=1,
P=15
✗

Potato
D=5,
P=10

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

✗

Brinjal
D=2,
P=30

Spinach
D=1,
P=25

Tomato
D=2,
P=20

Sour spinach
D=1,
P=15

Potato
D=5,
P=10

✓ ✓

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

✗

Brinjal
D=2,
P=30

Spinach
D=1,
P=25

Tomato
D=2,
P=20

Sour spinach
D ✗,
P=15

Potato
D=5,
P=10

✓ ✓

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

✗

✗

Brinjal
D=2,
P=30

Spinach
D=1,
P=25

Tomato
D=2,
P=20

Sour spinach
D=1,
P=15

Potato
D=5,
P=10

✓

✓

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

✗

✗

Brinjal
D=2,
P=30

Spinach
D=1,
P=25

Tomato
D=2,
P=20

Sour spinach
D=1,
P=15

Potato
D=5,
P=10

✓

✓

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

✗ ✗

Brinjal
D=2,
P=30

Spinach
D=1,
P=25

Tomato
D=2,
P=20

Sour spinach
D=1,
P=15

Potato
D=5,
P=10

✓ ✓ ✓

Job Sequencing with Deadlines Problem

Purchasing Vegetables for three days

Brinjal
D=3,
P=30

Spinach
D=1,
P=25

Tomato
D=2,
P=20

Sour spinach
D=1,
P=15

Potato
D=5,
P=10

DAY1	DAY2	DAY3

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU

JOB-1

D=2,
P=30

JOB-2

D=1,
P=25

JOB-3

D=2,
P=20

JOB-4

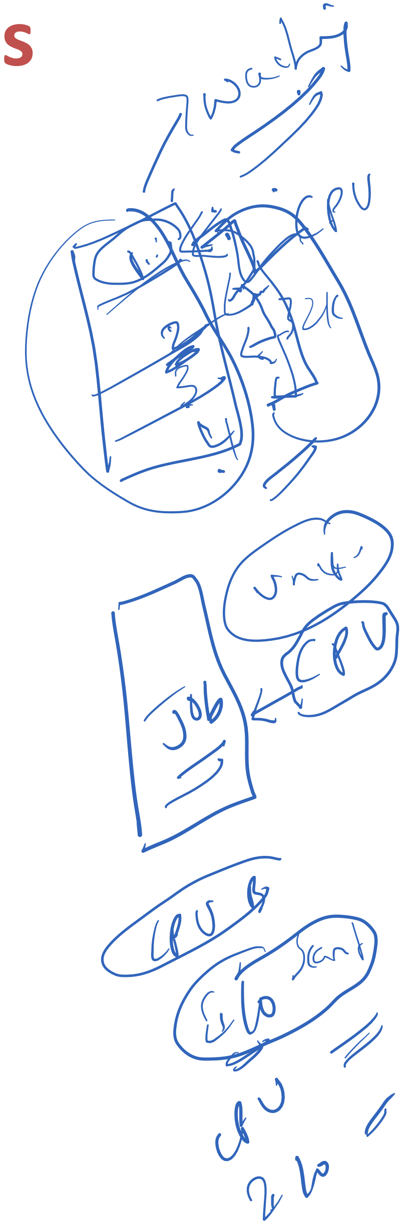
D=1,
P=15

JOB-5

D=5,
P=10

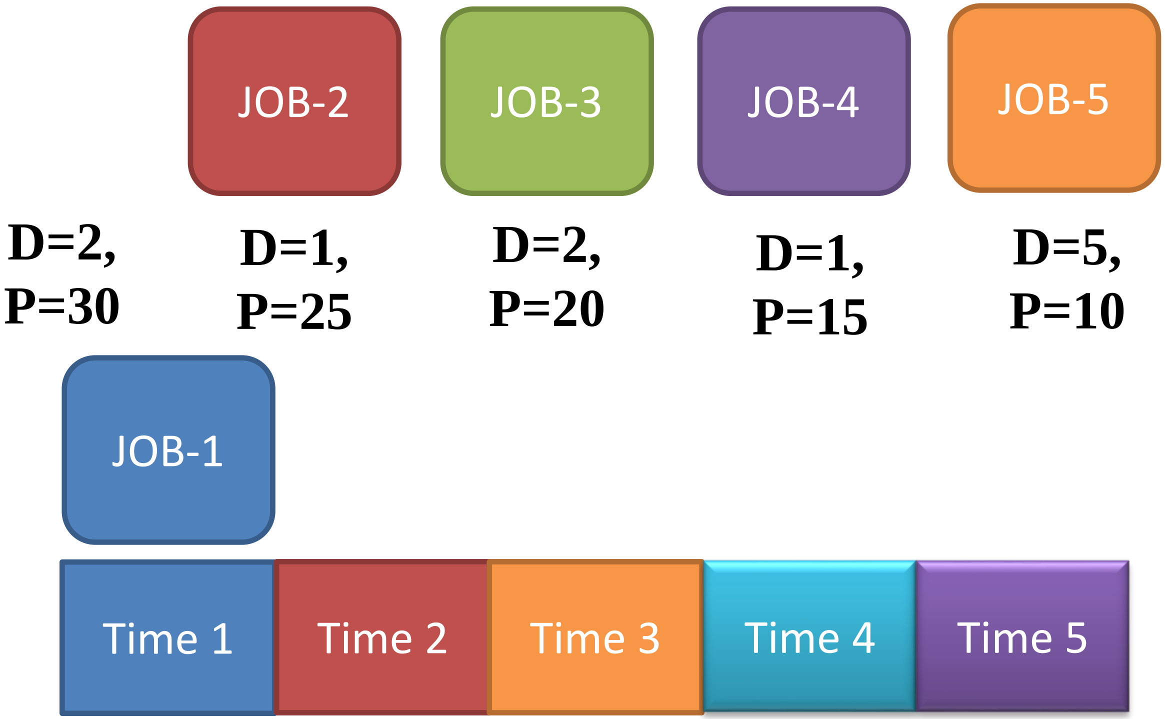
CPU Time Slots

Time 1 | Time 2 | Time 3 | Time 4 | Time 5



Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU



JOB-2

JOB-3

JOB-4

JOB-5

D=2, P=30

D=1, P=25

D=2, P=20

D=1, P=15

D=5, P=10

JOB-1

CPU Time Slots

Time 1

Time 2

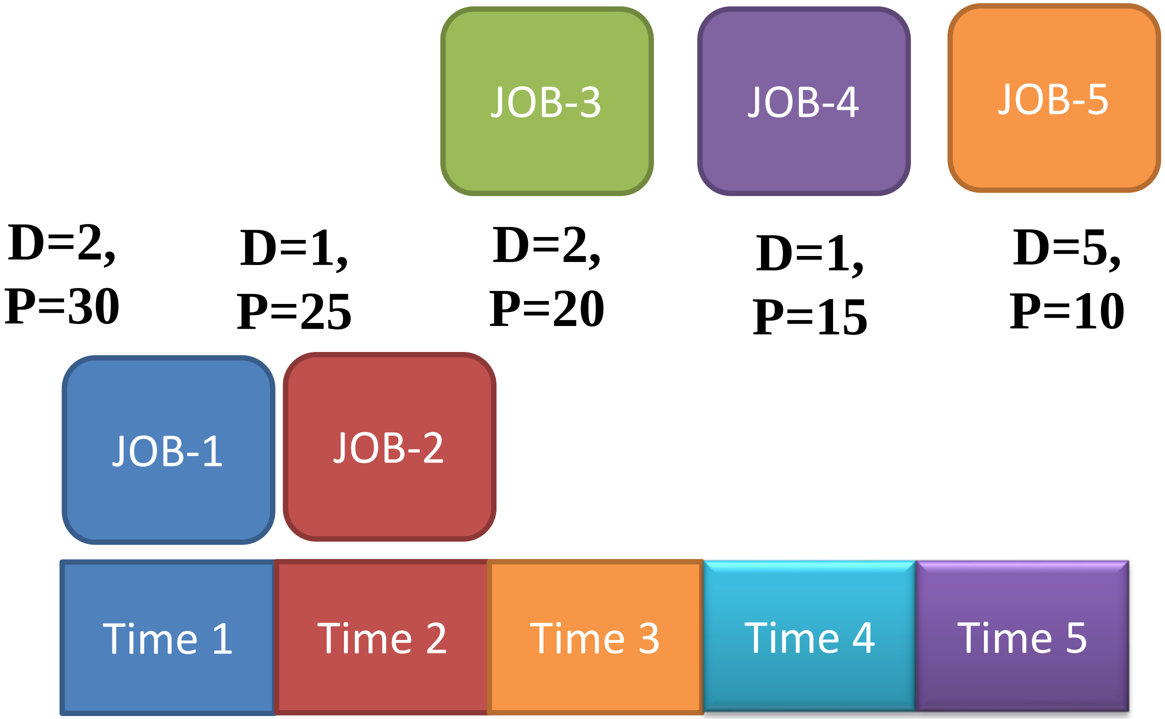
Time 3

Time 4

Time 5

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU



JOB-3

JOB-4

JOB-5

D=2, P=30

D=1, P=25

D=2, P=20

D=1, P=15

D=5, P=10

JOB-1

JOB-2

CPU Time Slots

Time 1

Time 2

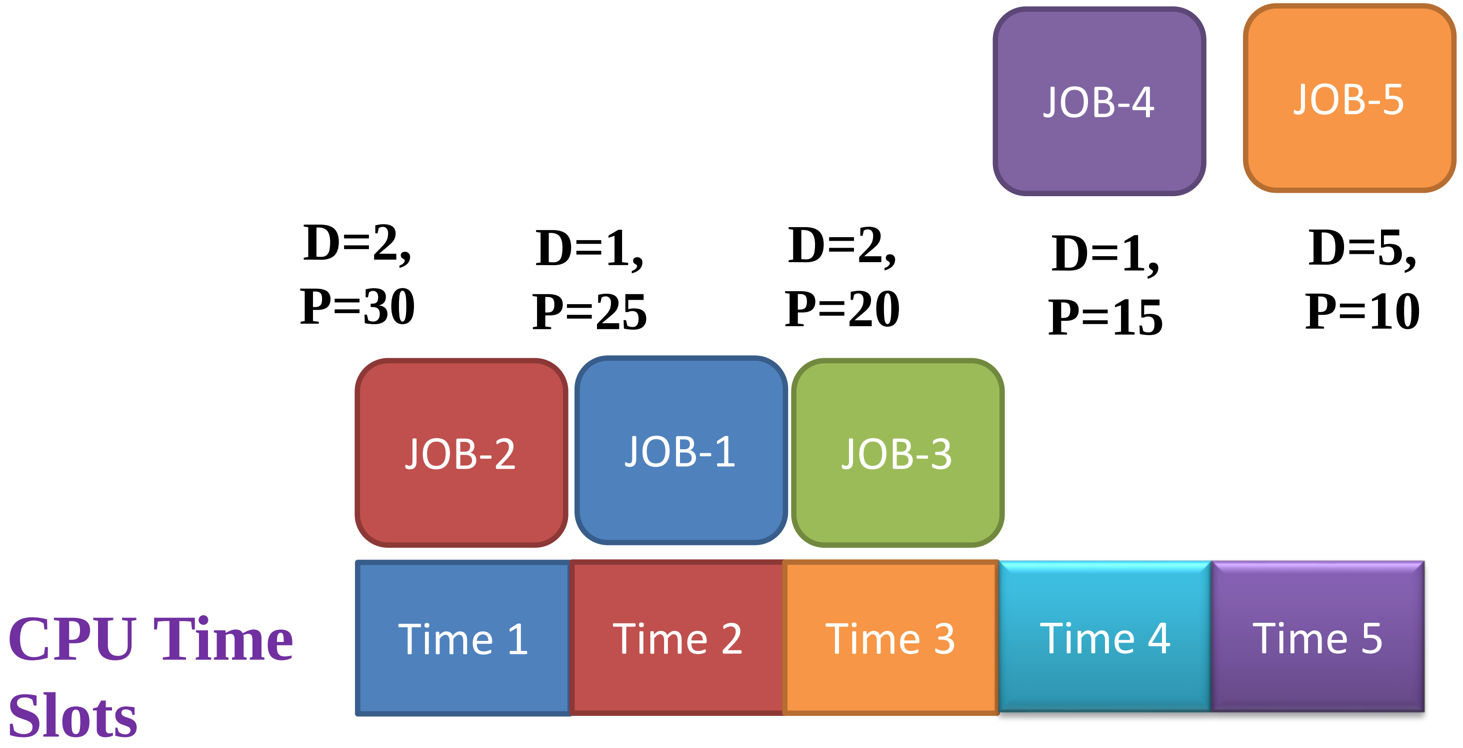
Time 3

Time 4

Time 5

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU



JOB-4

JOB-5

D=2, P=30

D=1, P=25

D=2, P=20

D=1, P=15

D=5, P=10

JOB-2

JOB-1

JOB-3

CPU Time Slots

Time 1

Time 2

Time 3

Time 4

Time 5

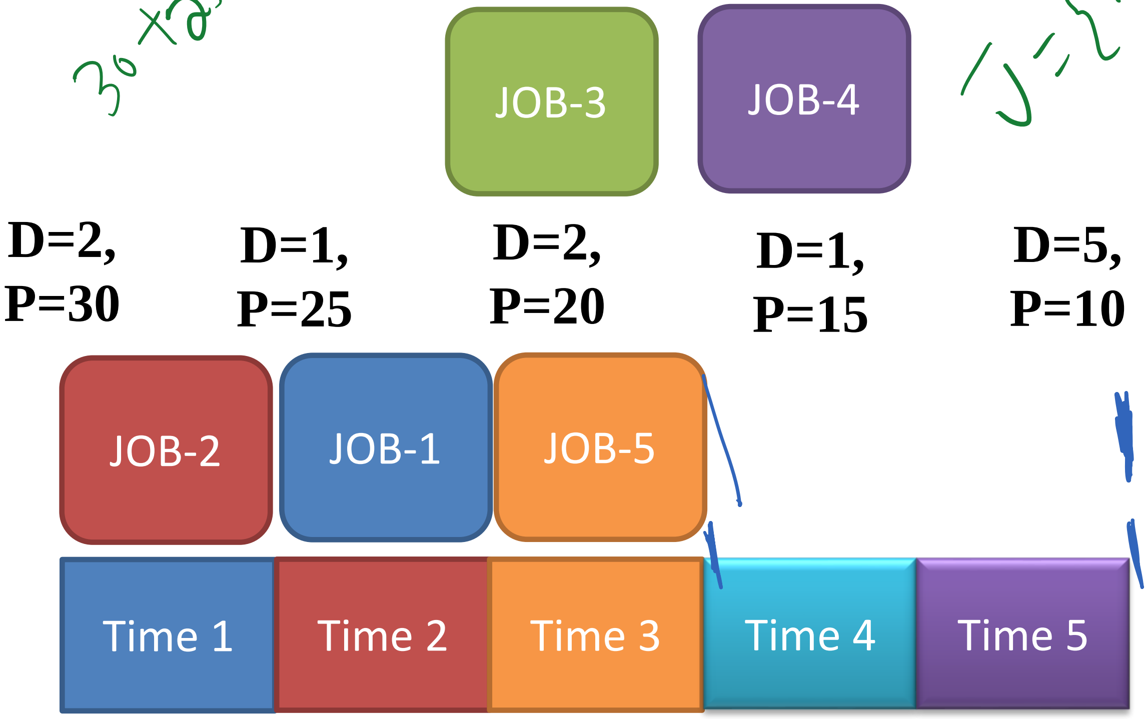
Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU

30+25+10

J={1, 2,5}

P1+P2+P5



CPU Time Slots

$n=4$

$(p_1, p_2, p_3, p_4) = (100, 10, 15, 27)$

$(d_1, \ldots, d_4) = (2, 1, 2, 1)$

$(1,2)$ (1)

$(1,3)$ (2)

$(1,4)$, (3)

$(2,3)$ (4)

$(3,4)$

$J = \{1,2\}$

$p_1 + p_2$

$p_2\ 7$

$J =$

```
Algorithm GreedyJob(d, J, n)
{
    $J = \{1\}$
    for $i = 2$ to $n$ do
    {
        if all Jobs in $J \cup \{i\}$ can be completed within their deadline
        then $J = J \cup \{i\}$
    }
}
```

Job Sequencing with Deadlines Problem – Algorithm

```
Algorithm JS(d, j, n)    {
// d[i] ≥ 1, 1 ≤ i ≤ n are the deadlines, n ≥ 1. The jobs are  ordered such that
p[1] ≥ p[2] …… ≥ p[n]
// j[i] is the i$^{th}$ job in the optimal solution, 1 ≤  i ≤  k , at termination d [ j[i]] ≤
d[j[i+1]], 1 ≤ i ≤  k
    d[0] := j[0] := 0;              // Initialize
    j[1] := 1;                      // Include job 1
    k := 1;
    for i := 2 to n do  {
        //Consider jobs in Descending order of p[i]. Find position for i and  check
feasibility of  insertion
        r := k;
        while( ( d[ j[r]]> d[i]  and ( d[j[r]] ≠ r )) do
            r :=  r - 1;
        if(  d[i]  > r )) then  {
         //  Insert i into  j[].
            for q = k to (r+1) do
```

J[0+1]

J[i] = i{1, 3, 5}

dead

2 4 5

d[J[i]]

dead

k = 3

d[J[3]] > d[i]

i = 2

0 > 1

J = {2, 1, 3, 5}

1 > 0

q = 3 to (1) do

q = 2

J[q+1] = J[3]

J[3] = J[2]

J[2] = J[1]

J[1] = i

Job Sequencing with Deadlines Problem – Algorithm

```
Algorithm JS(d, j, n)    {
    d[0] := j[0] := 0;                    // Initialize
    j[1] := 1;                           // Include job 1
    k := 1;
    for i := 2 to n do  {
       //Consider jobs in Descending order of p[i]. Find position for i and  check
feasibility of  insertion
        r := k;
        while( ( d[ j[r]]> d[i]  and ( d[j[r]] ≠ r )) do
             r :=  r - 1;
        if(  d[i]  > r )) then  {                //  Insert i into  j[].
           for q = k to (r+1) do
                j[q+1] = j[q];
           j[r+1] := i;
           k:=k+1;
 }  }
      return k;  }
```

Job Sequencing with Deadlines Problem – Algorithm

```
Algorithm JS(d, j, n)    {
d[0] := j[0] := 0;                    // Initialize
    j[1] := 1;                    // Include job 1
    k := 1;
      for i := 2 to n do  {
          //Consider jobs in Descending order of p[i]. Find position for i and  check
feasibility of  insertion
           r := k;
           while( ( d[ j[r] ]> d[i] ) and ( d[j[r]] ≠ r )) do
                 r :=  r - 1;
            if(  d[i]  > r )) then  {           //  Insert i into  j[].
               for q = k to (r+1) do
                     j[q+1] = j[q];
               j[r+1] := i;
               k:=k+1;
 }  }
      return k;  }
```

Job Sequencing with Deadlines Problem – Algorithm

```
Algorithm JS(d, j, n)    {

    d[0] := j[0] := 0;              // Initialize
    j[1] := 1;                               // Include job 1
    k := 1;
    for i := 2 to n do  {
        r := k;
        while( ( d[ j[r] ]> d[i] ) and ( d[j[r]] ≠ r )) do
            r :=  r - 1;
        if (  d[i]  > r ) then  {       //  Insert i into  j[].
            for q = k to (r+1) do
                j[q+1] = j[q];
            j[r+1] := i;
            k:=k+1;    }
    }// End of for  loop
    return k;  }
```

Job Sequencing with Deadlines Problem – Algorithm

```
Algorithm JS(d, j, n)    {
    d[0] := j[0] := 0;              // Initialize
    j[1] := 1;              // Include job 1
    k := 1;
    for i := 2 to n do  {
        r := k;
        while( ( d[ j[r] ]> d[i] ) and ( d[j[r]] ≠ r )) do
            r :=  r - 1;
         if (  d[i]  > r ) then  {       //  Insert i into  j[].
            for q = k to (r+1) do
                j[q+1] = j[q];
            j[r+1] := i;
            k:=k+1;
}  }
      return k;  }
```

Job Sequencing with Deadlines Problem – Algorithm

```
Algorithm JS(d, j, n)    {
d[0] := j[0] := 0;                      // Initialize
    j[1] := 1;                          // Include job 1
    k := 1;
    for i := 2 to n do  {
        r := k;
        while( ( d[ j[r] ]> d[i] ) and ( d[j[r]] ≠ r )) do
            r :=  r - 1;
         if (  d[i]  > r ) then  {       //  Insert i into  j[].
            for q = k to (r+1) do
                j[q+1] = j[q];
            j[r+1] := i;
            k:=k+1;
 }  }
     return k;  }
```

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU

JOB-1
JOB-2
JOB-3
JOB-4
JOB-5

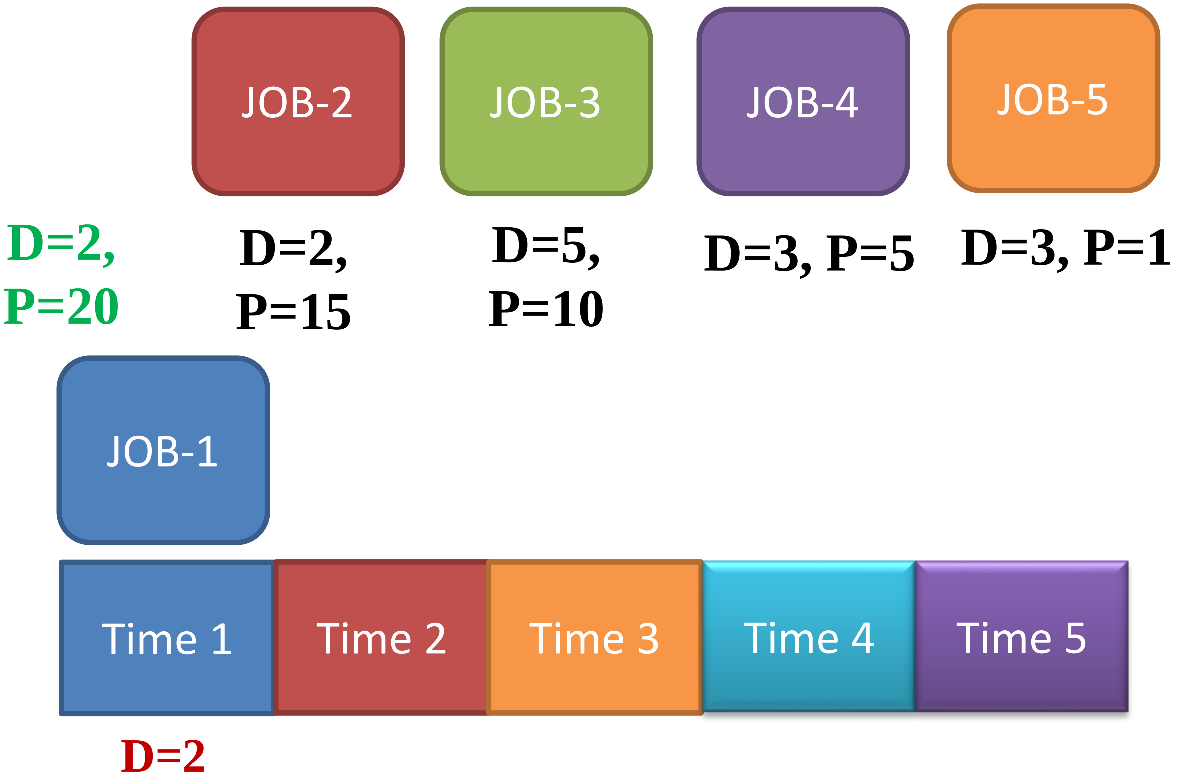
JOB-1	JOB-2	JOB-3	JOB-4	JOB-5
D=2, P=20	**D=2, P=15**	**D=5, P=10**	**D=3, P=5**	**D=3, P=1**

CPU Time Slots

Time 1
Time 2
Time 3
Time 4
Time 5

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU



JOB-2

JOB-3

JOB-4

JOB-5

D=2, P=20

D=2, P=15

D=5, P=10

D=3, P=5

D=3, P=1

JOB-1

CPU Time Slots

Time 1

Time 2

Time 3

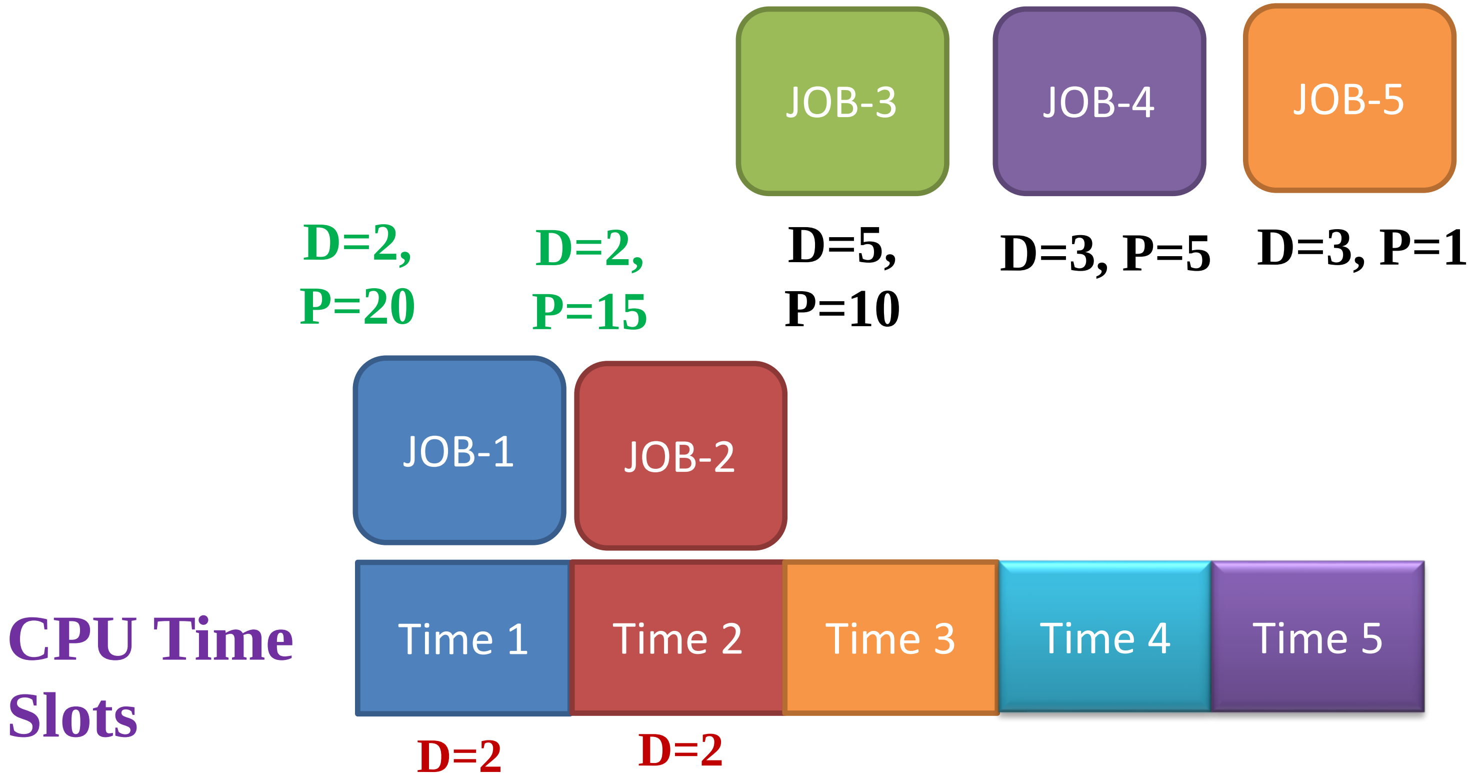
Time 4

Time 5

D=2

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU



JOB-3

JOB-4

JOB-5

D=2, P=20

D=2, P=15

D=5, P=10

D=3, P=5

D=3, P=1

JOB-1

JOB-2

CPU Time Slots

Time 1

Time 2

Time 3

Time 4

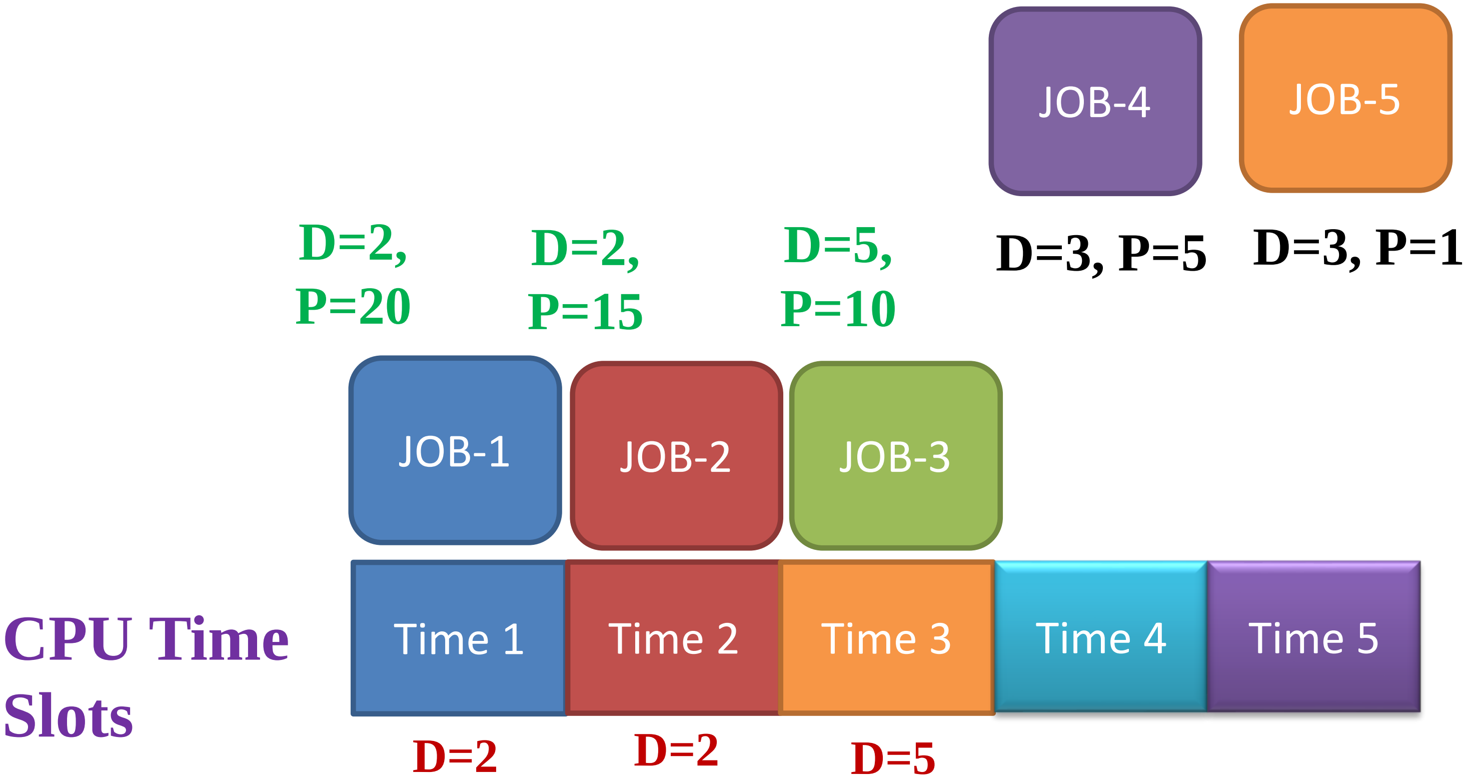
Time 5

D=2

D=2

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU



JOB-4

JOB-5

D=3, P=5

D=3, P=1

D=2, P=20

D=2, P=15

D=5, P=10

JOB-1

JOB-2

JOB-3

CPU Time Slots

Time 1

Time 2

Time 3

Time 4

Time 5

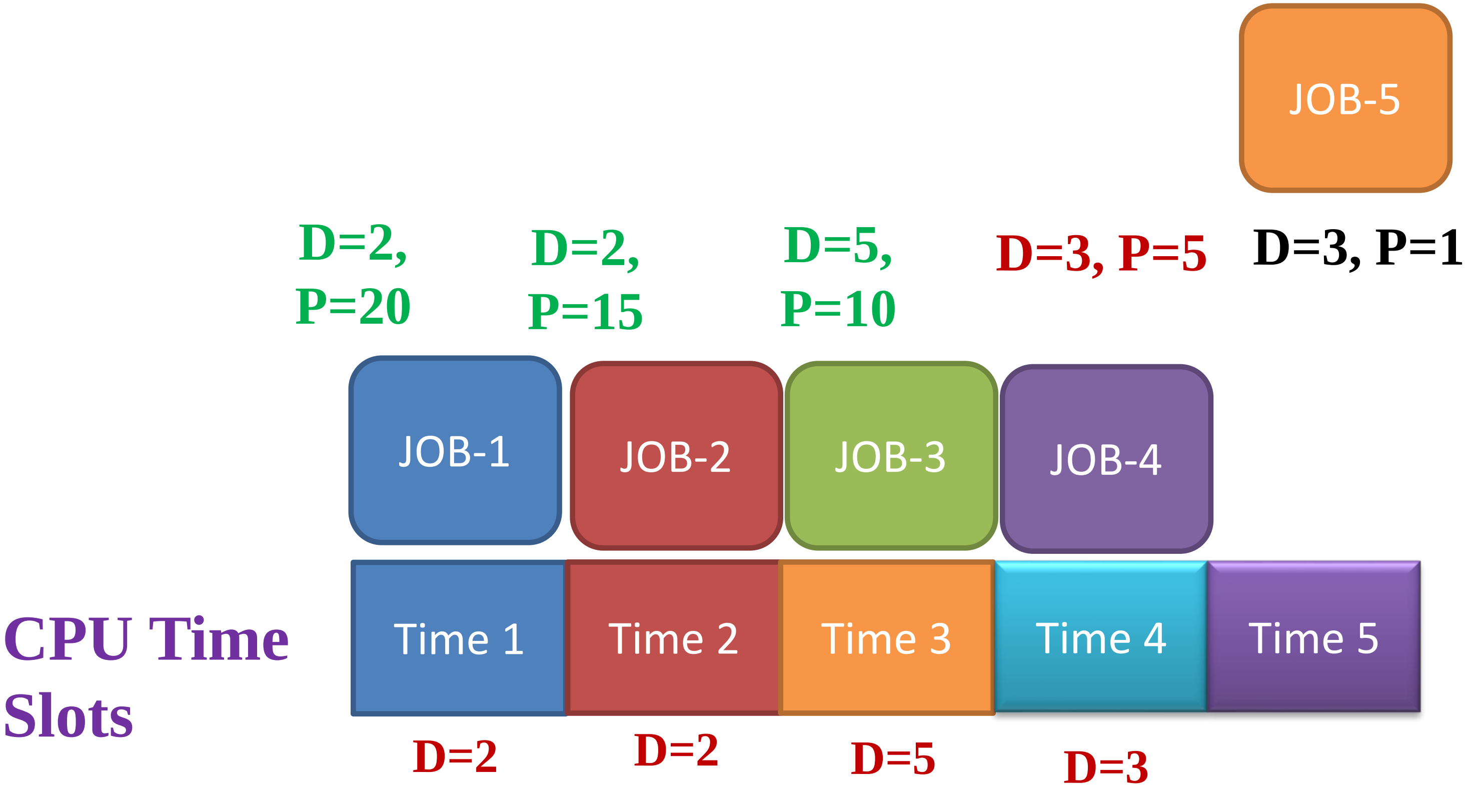
D=2

D=2

D=5

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU



JOB-5

D=3, P=1

D=2, P=20

D=2, P=15

D=5, P=10

D=3, P=5

JOB-1

JOB-2

JOB-3

JOB-4

CPU Time Slots

Time 1

Time 2

Time 3

Time 4

Time 5

D=2

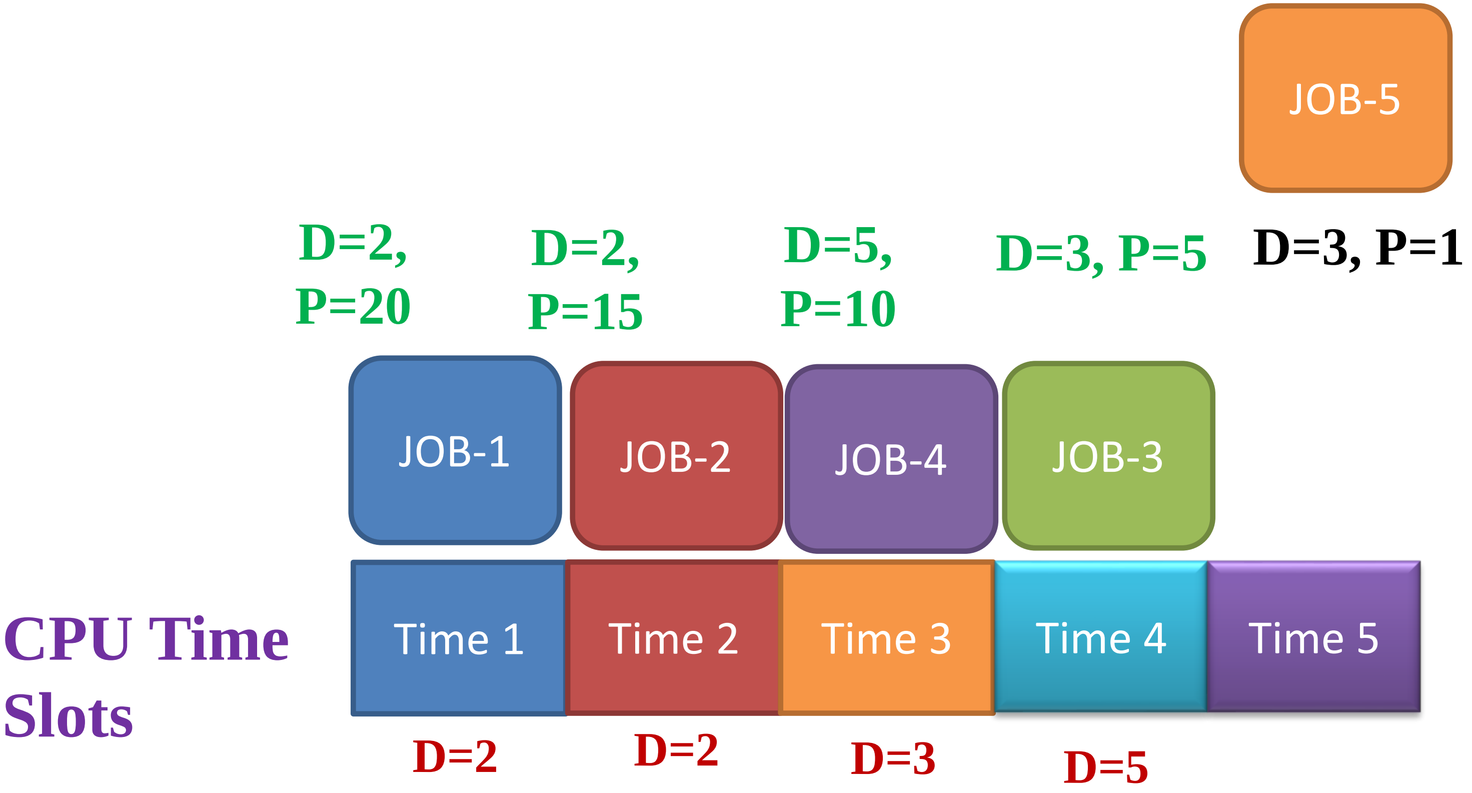
D=2

D=5

D=3

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU



JOB-5

D=3, P=1

D=2, P=20

D=2, P=15

D=5, P=10

D=3, P=5

JOB-1

JOB-2

JOB-4

JOB-3

CPU Time Slots

Time 1

Time 2

Time 3

Time 4

Time 5

D=2

D=2

D=3

D=5

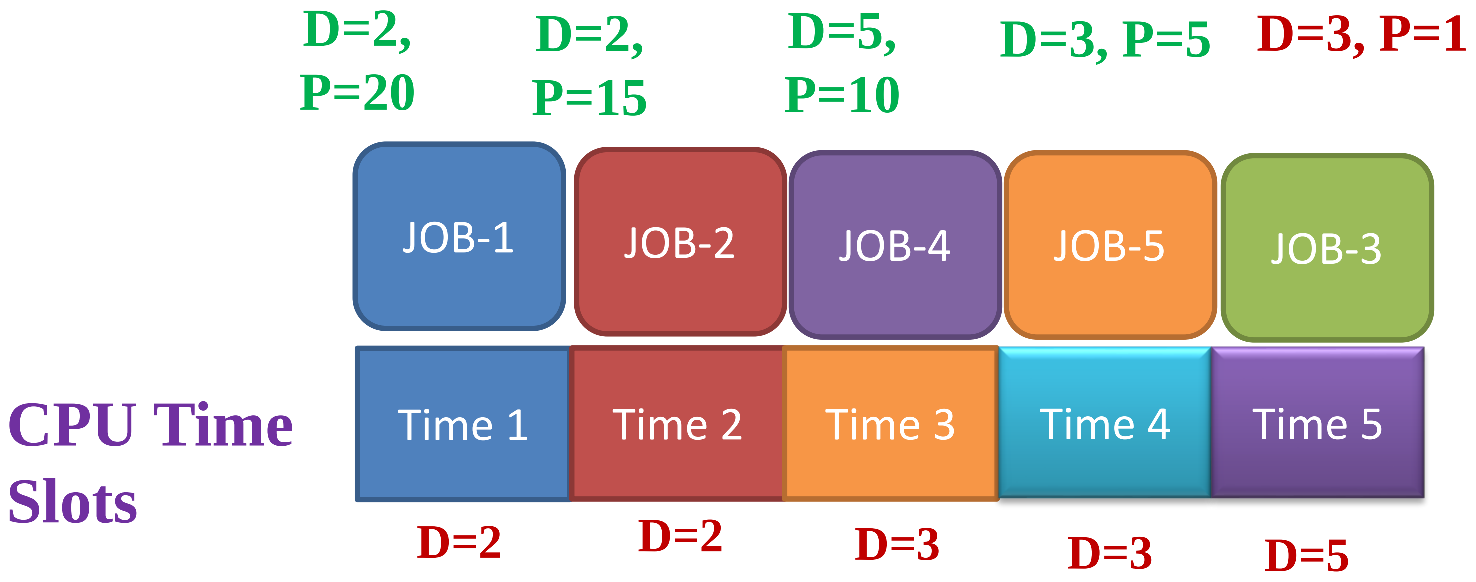
Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU

	D=2, P=20	**D=2, P=15**	**D=5, P=10**	**D=3, P=5**	**D=3, P=1**
	JOB-1	JOB-2	JOB-4	JOB-3	JOB-5
CPU Time Slots	Time 1	Time 2	Time 3	Time 4	Time 5
	D=2	**D=2**	**D=3**	**D=5**	**D=3**

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU



D=2, P=20

D=2, P=15

D=5, P=10

D=3, P=5

D=3, P=1

JOB-1

JOB-2

JOB-4

JOB-5

JOB-3

CPU Time Slots

Time 1

Time 2

Time 3

Time 4

Time 5

D=2

D=2

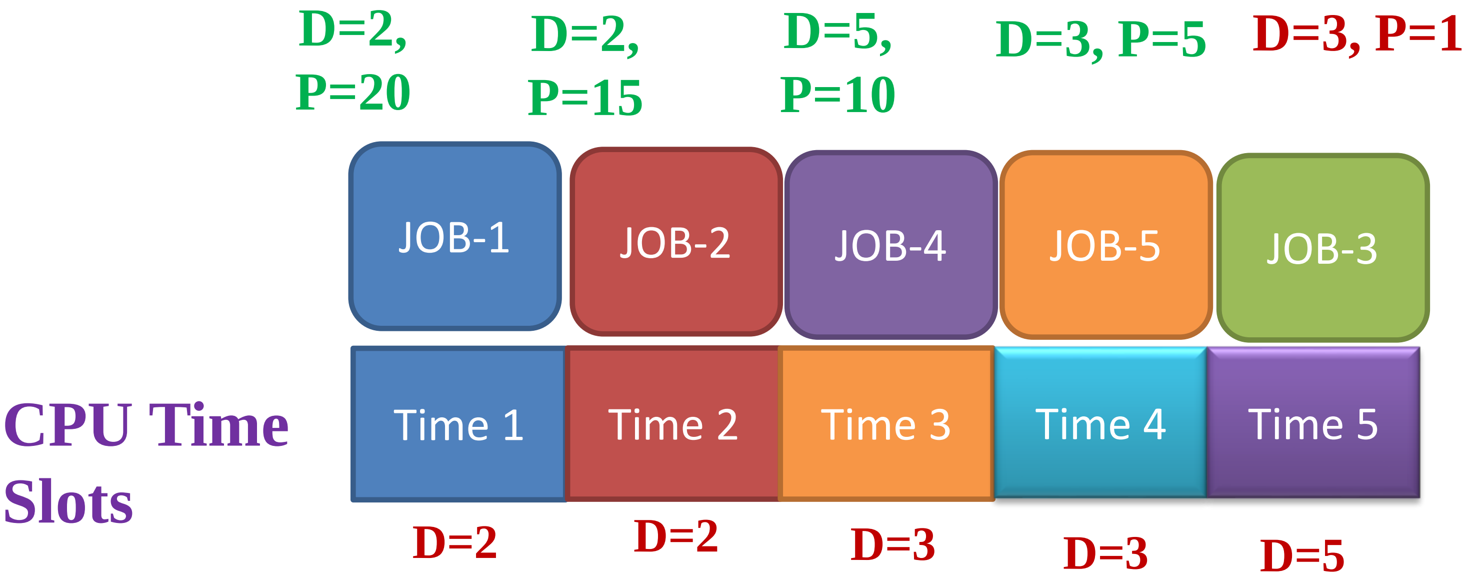
D=3

D=3

D=5

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU



D=2, P=20
D=2, P=15
D=5, P=10
D=3, P=5
D=3, P=1

JOB-1
JOB-2
JOB-4
JOB-5
JOB-3

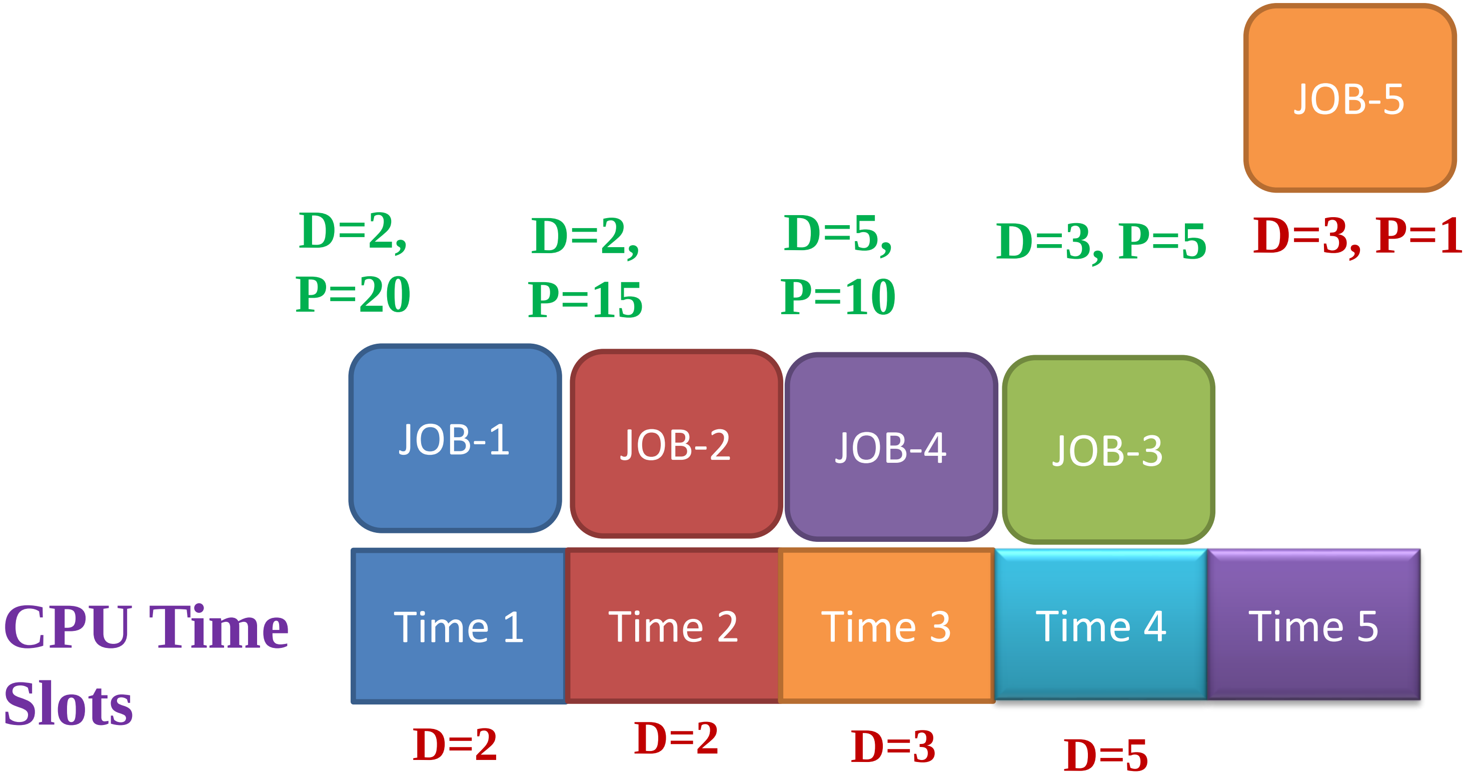
CPU Time Slots

Time 1
Time 2
Time 3
Time 4
Time 5

D=2
D=2
D=3
D=3
D=5

Job Sequencing with Deadlines Problem

scheduling jobs on a Single CPU



JOB-5

D=2, P=20

D=2, P=15

D=5, P=10

D=3, P=5

D=3, P=1

JOB-1

JOB-2

JOB-4

JOB-3

CPU Time Slots

Time 1

Time 2

Time 3

Time 4

Time 5

D=2

D=2

D=3

D=5

Job Sequencing with Deadlines Problem – Algorithm

```
Algorithm JS(d, j, n)   {
d[0] := j[0] := 0;                    // Initialize
    j[1] := 1;                        // Include job 1
    k := 1;
    for i := 2 to n do  {
        r := k;
        while( ( d[ j[r] ]> d[i] ) and ( d[j[r]] ≠ r )) do
            r :=  r - 1;
         if (  d[i]  > r ) then  {      //  Insert i into  j[].
            for q = k to (r+1) do
                j[q+1] = j[q];
            j[r+1] := i;
            k:=k+1;
 }  }
      return k;  }
```

In this algorithm, we are using two loops, one is within another. Hence, the **Time Complexity** of this algorithm is **O(n^2).**