



Large Scale Optimization

Emmanouil Zachariadis, Assistant Professor

Department of Management Science and Technology, Athens University of Economics and Business

E: ezach@aueb.gr

A: Evelpidon 47A and Lefkados 33 street, 9th floor, Room 906, 11362, Athens, Greece

T: +30-210-8203 674

Cutting problems

- Consider a set of 20-m metal bars in stock
- **Orders:** Sets of bars ($<20\text{m}$)
 - Customer 1: 2 bars of 3m, 4 bars of 5m
 - Customer 2: 6 bars of 1m, 23 bars of 2 m
 - ...
- Cutting problem
- The objective is to minimize the spare quantity
 - Minimization of the metal which cannot be reprocessed



Example: Cutting problem

- In total, 11 bars have to be produced
- The length of each bar is seen in the following table:

K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_8	K_9	K_{10}	K_{11}
8	7	14	9	6	9	5	15	6	7	8

Example: Cutting problem

- **Objective:** Minimize the metal which cannot be reprocessed

- Minimize the number of 20m bars cut

- Bin Packing Problem:

- 20-m Bars: Bins

- Ordered Bars: Items

- **Solution Structure:**

Set of assignments between ordered bars and 20-m bars

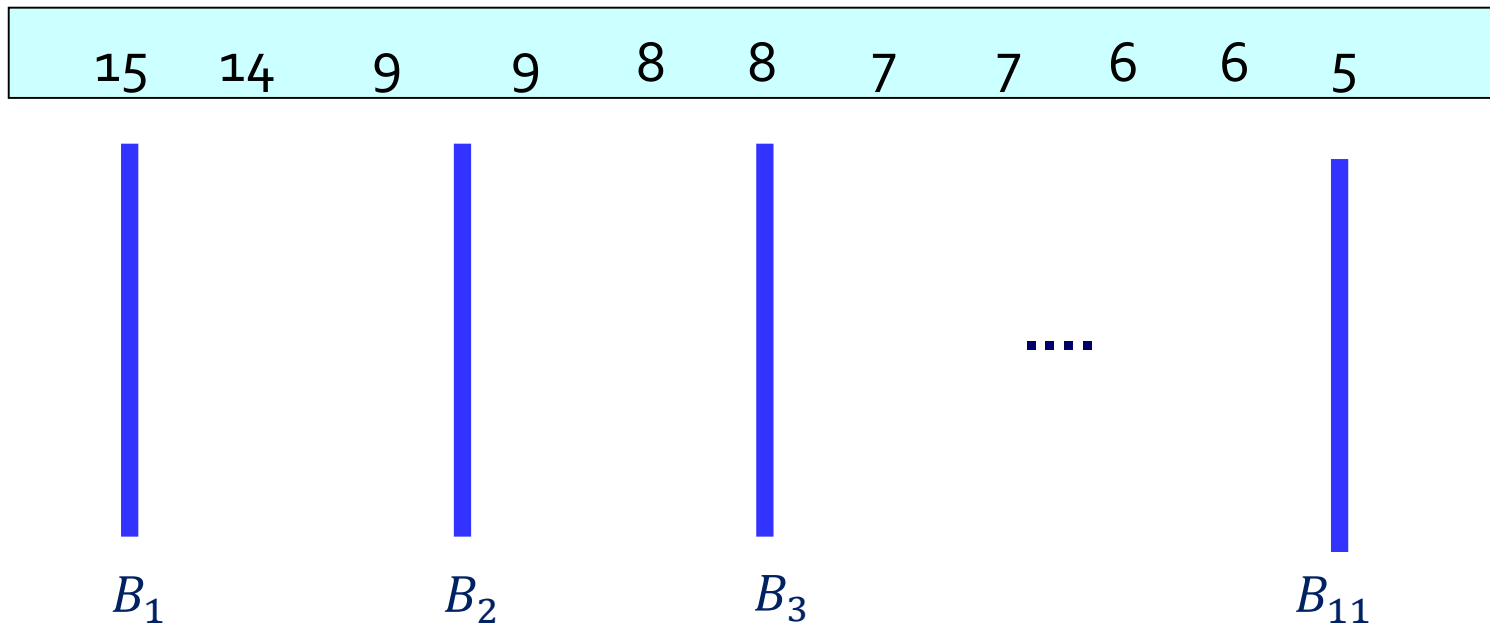
- **Solution Feature:**

An assignment of an ordered bar to a 20-m bar

Example: Cutting problem

First-Fit Algorithm:

- Consider and arrange 11 20-m bars as: B_1, B_2, \dots, B_{11}
- Arrange the ordered bars in decreasing order size
- Iteratively, assign the first ordered bar to the first 20-m bar with adequate size



Example: Cutting problem

The First-Fit Algorithm for the cutting problem is executed:

- Iteration 0. Partial Solution s is empty
- Iteration 1. Ordered Bar K_8 (size 15) is assigned to B_1
 $s: (K_8, B_1)$
- Iteration 2. Ordered Bar K_3 (size 14) is assigned to B_2
 $s: \{(K_8, B_1), (K_3, B_2)\}$
- Iteration 3. Ordered Bar K_4 (size 14) is assigned to B_3
 $s: \{(K_8, B_1), (K_3, B_2), (K_4, B_3)\}$

Example: Cutting problem

- Iteration 4. Ordered Bar K_6 (size 9) is assigned to B_3

$s: \{(K_8, B_1), (K_3, B_2), (K_4, B_3), (K_6, B_3)\}$

- Iteration 5. Ordered Bar K_1 (size 8) is assigned to B_4

$s: \{(K_8, B_1), (K_3, B_2), (K_4, B_3), (K_6, B_3), (K_1, B_4)\}$

- Iteration 6. Ordered Bar K_{11} (size 8) is assigned to B_4

$s: \{(K_8, B_1), (K_3, B_2), (K_4, B_3), (K_6, B_3), (K_1, B_4), (K_{11}, B_4)\}$

- Iteration 7. Ordered Bar K_2 (size 7) is assigned to B_5

$s: \{(K_8, B_1), (K_3, B_2), (K_4, B_3), (K_6, B_3), (K_1, B_4), (K_{11}, B_4), (K_2, B_5)\}$

Example: Cutting problem

- Iteration 8. Ordered Bar K_{10} (size 7) is assigned to B_5
 $s: \{(K_8, B_1), (K_3, B_2), (K_4, B_3), (K_6, B_3), (K_1, B_4), (K_{11}, B_4), (K_2, B_5), (K_{10}, B_5)\}$
- Iteration 9. Ordered Bar K_5 (size 6) is assigned to B_2
 $s: \{(K_8, B_1), (K_3, B_2), (K_4, B_3), (K_6, B_3), (K_1, B_4), (K_{11}, B_4), (K_2, B_5), (K_{10}, B_5), (K_5, B_2)\}$
- Iteration 10. Ordered Bar K_9 (size 6) is assigned to B_5
 $s: \{(K_8, B_1), (K_3, B_2), (K_4, B_3), (K_6, B_3), (K_1, B_4), (K_{11}, B_4), (K_2, B_5), (K_{10}, B_5), (K_5, B_2), (K_9, B_5)\}$
- Iteration 11. Ordered Bar K_7 (size 5) is assigned to B_1
 $s: \{(K_8, B_1), (K_3, B_2), (K_4, B_3), (K_6, B_3), (K_1, B_4), (K_{11}, B_4), (K_2, B_5), (K_{10}, B_5), (K_5, B_2), (K_9, B_5), (K_7, B_1)\}$

Example: Cutting problem

S :	s1:	K ₈	K ₇	
	s2:	K ₃	K ₅	
	s3:	K ₄	K ₆	
	s4:	K ₁	K ₁₁	
	s5:	K ₂	K ₁₀	K ₉

Facility Location Problem

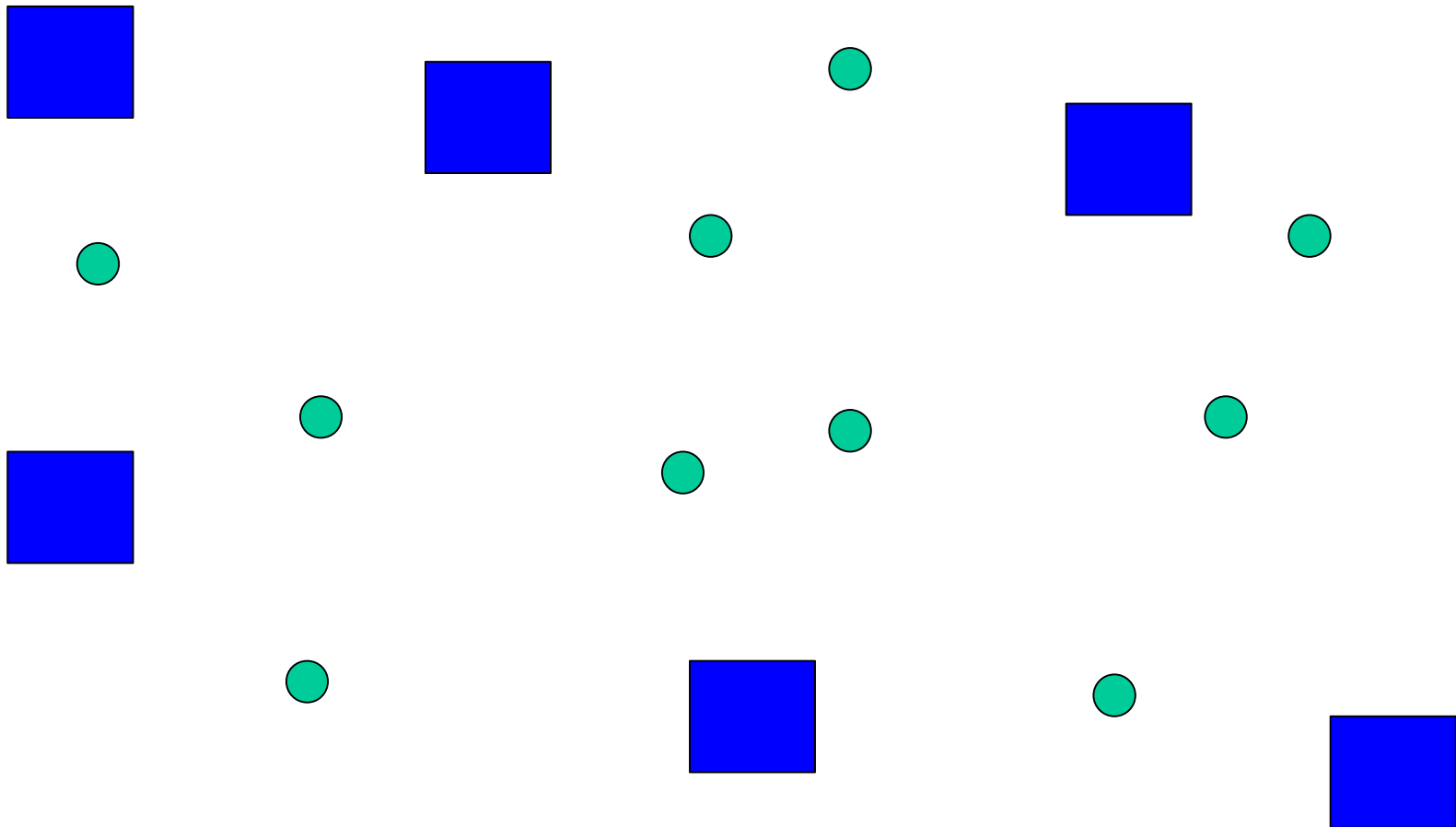
- It is decided that 3 firefighting facilities must be located in Mount Pelion
- The 3 firefighting units will accommodate 10 vehicles
- Each vehicle must leave the firefighting facility, go to a specific waiting point
- In total, 6 candidate location have been selected for the units
- Objective:

Minimization of total km travelled by all vehicles to move from their facilities to the waiting points

Facility Location Problem

- Constraints:
 - Every location can host a specific number of vehicles
 - Every vehicle is assigned to exactly one facility
- In the following figure:
 - Blue Squares: Candidate locations
 - Green Circles: Waiting Points

Facility Location Problem



Facility Location Problem

- Table: Maximum number of vehicles per candidate location

	Candidate Locations					
	A	B	C	D	E	F
Maximum Number of Vehicles	4	6	5	8	4	5

Facility Location Problem

- Distances for each (Location, Waiting Point) pair
 - If a vehicle is located at Location A, it must travel 89 km to get to waiting point 3

		Firefighter trucks waiting points									
		1	2	3	4	5	6	7	8	9	10
Locations	A	72	36	89	46	130	56	61	35	34	56
	B	31	140	79	63	78	31	25	92	128	29
	C	56	53	45	88	25	44	33	37	74	46
	D	78	68	28	42	141	72	45	78	28	89
	E	160	86	74	71	137	83	61	89	179	91
	F	95	41	34	122	39	52	54	22	45	23

Facility Location Problem

- In which category lies this problem
 - Permutation/Assignment/Selection
- Design a Greedy Algorithm for this problem
- Calculate the Objective of the final solution

Facility Location Problem – Greedy Algorithm

- Solution Structure:

A set of 10 (Location, Vehicle)=(Location, Waiting Point) assignments

- Solution Element:

A (Location, Vehicle) assignment

- Selection Criterion:

The selection criterion is to select the assignment minimizing the extra distance

Subject to:

Each vehicle is assigned to one location

Maximum three locations

Capacity of the location is respected

Facility Location Problem – Greedy Algorithm

- Objective Function:

The total Distance over the 10 assignments

The total distance travelled by the 10 vehicles to depart from their locations and transit to their waiting points

Facility Location Problem – Greedy Algorithm

Iteration 1: Assignment Vehicle 8 - Location F

$S = \{(8F)\}$

Iteration 2: Assignment Vehicle 10 - Location F

$S = \{(8F, 10F)\}$

Iteration 3: Assignment Vehicle 5 - Location C

$S = \{(8F, 10F, 5C)\}$

Iteration 4: Assignment Vehicle 7 - Location B

$S = \{(8F, 10F, 5C, 7B)\}$

Facility Location Problem – Greedy Algorithm

- At this point, the partial solution contains three locations
- Indirectly, we have selected the locations to be used
- All of the other locations are now inapplicable
- Thus, for the rest of the iterations only locations B,C,F will be taken into consideration

ΧΩΣΟΘΕΤΗΣΗ ΕΓΚΑΤΑΣΤΑΣΕΩΝ με ΧΣΗΣΗ ΡΣΕ

Iteration 5: Assignment Vehicle 1 - Location B

$S=\{(8F,10F,5C,7B,1B)\}$

Iteration 6: Assignment Vehicle 6 - Location B

$S=\{(8F,10F,5C,7B,1B,6B)\}$

Iteration 7: Assignment Vehicle 3 - Location F

$S=\{(8F,10F,5C,7B,1B,6B,3F)\}$

Iteration 8: Assignment Vehicle 2 - Location F

$S=\{(8F,10F,5C,7B,1B,6B,3F,2F)\}$

Iteration 9: Assignment Vehicle 9 - Location F

$S=\{(8F,10F,5C,7B,1B,6B,3F,2F,9F)\}$

Iteration 10: Assignment Vehicle 4 - Location B

$S=\{(8F,10F,5C,7B,1B,6B,3F,2F,9F,4B)\}$

Facility Location Problem – Greedy Algorithm

- The objective value of the final solution is 340km
- Comments
 - What is a weak point of the algorithm regarding the feasibility?
 - Is it safe that the algorithm will produce a feasible solution?
 - What is the weak point of the algorithm regarding the final solution quantity?
 - What table features might lead to an extremely poor solution?
 - Suggest another Greedy algorithm for such tables

PROJECT SCHEDULING PROBLEM (PSP)

- A project consists of a set of activities, that need to be executed with respect to a set of precedence constraints
- Every activity:
 - Has **predefined time duration** and
 - **Requires a set of resources in order to be implemented.** Financial resources, human resources, machinery, specialized equipment, energy, materials, storing facilities, are some examples of the types of resources that may be used for the completion of a project.

PSP Constraints

- Activity related constraints:
 - Activity precedence constraints
 - Duration between two activities constraints
- Resources Constraints
- Objective: Minimization of the total duration of the project, i.e., the project makespan



The Resource Constrained Project Scheduling Problem - RCPSP

RCPSP

- Let a set of activities $A = 1, \dots, n$ that need to be completed in order for the project to finish
- Every activity i has a specific duration for completion d_i
- Precedence constraints:
In order for an activity $i \in A$ to start, a set of activities P_i needs to be completed

e.g. $P_8 = \{7, 10, 12\}$: In order for activity 8 to start, activities 7, 10 and 12 must be completed

RCPSP

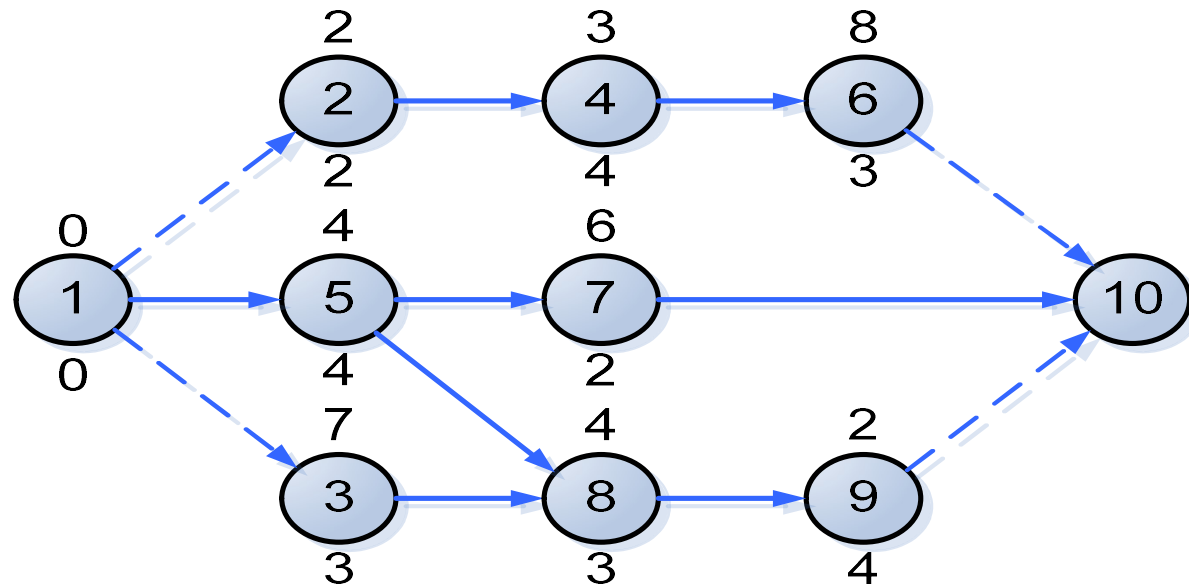
- Every activity has specific resource requirements
- Let a set of available resources $K = \{1, \dots, m\}$
- Every resource $k \in K$ has a capacity c_k , which is available for each time unit.
- Objective: minimization of the project timespan given that
 - The resource constraints are not violated
 - The capacity of each resource constraint is not violated for each time unit.

Example

- Assume a specific service/project that requires the completion of 8 activities in order to finish.
- There is only one resource type (human resource), with capacity of 8
- In other words, the activities that are executed each time unit cannot require more than 8 workers.

Example

- The duration (in time units) of each activity is given above the corresponding node, whereas the resource requirements are given below the node
- Design a Greedy Algorithm for makespan minimization



Solving via Greedy Algorithm

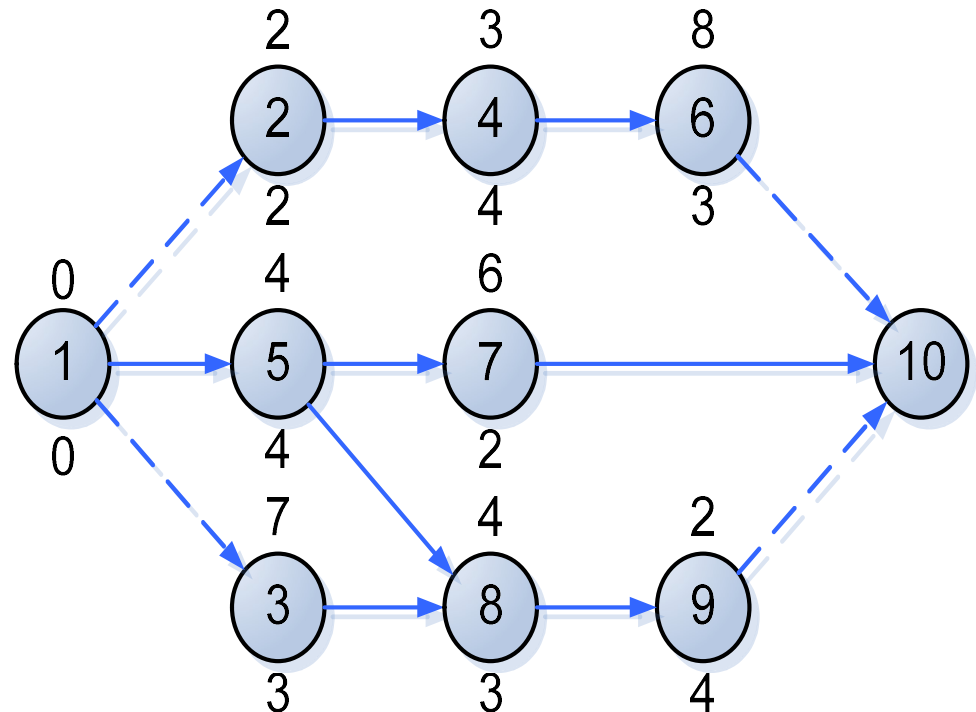
- Solution structure:
10 assignments of 10 activities to 10 time-periods (TP), where each TP is a range [BT , ET] with BT (Begin time) the earliest possible time for starting the activity and ET (End time) the latest possible completion time for this activity
- Solution element:
An assignment of an activity to a TP.
- Selection Criterion:
In each iteration, we choose the activity with the largest number of required resources. Then, we assign this activity to a TP [BT , ET]. Note that, the precedence and the resource constraints must be respected at all times.
- Objective function/Evaluation criterion:
The project time span is the maximum ET of any activity.

Solving via Greedy Algorithm

The greedy algorithms is implemented as follows:

Iteration 1.

Activity 1 is the only one without precedence constraints, and therefore it is chosen and assigned to TP T1: [0,0].



Solving via Greedy Algorithm

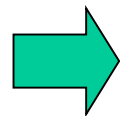
- Iteration 2.**

Since activity 5 requires the most resource units, it is chosen and assigned to TP T5:[0,4].

$$f(5)=4$$

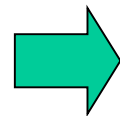
$$f(2)=2$$

$$f(3)=3$$



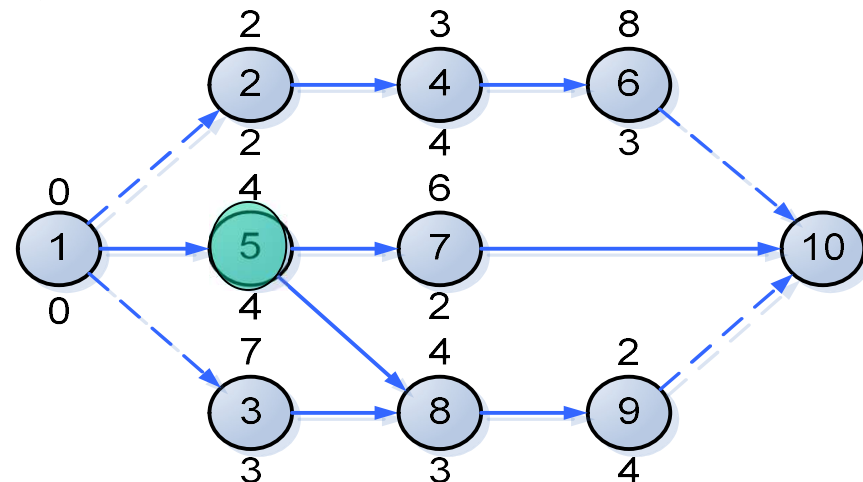
$$\text{Max } \{f(i)\}=4,$$

$$i=5$$

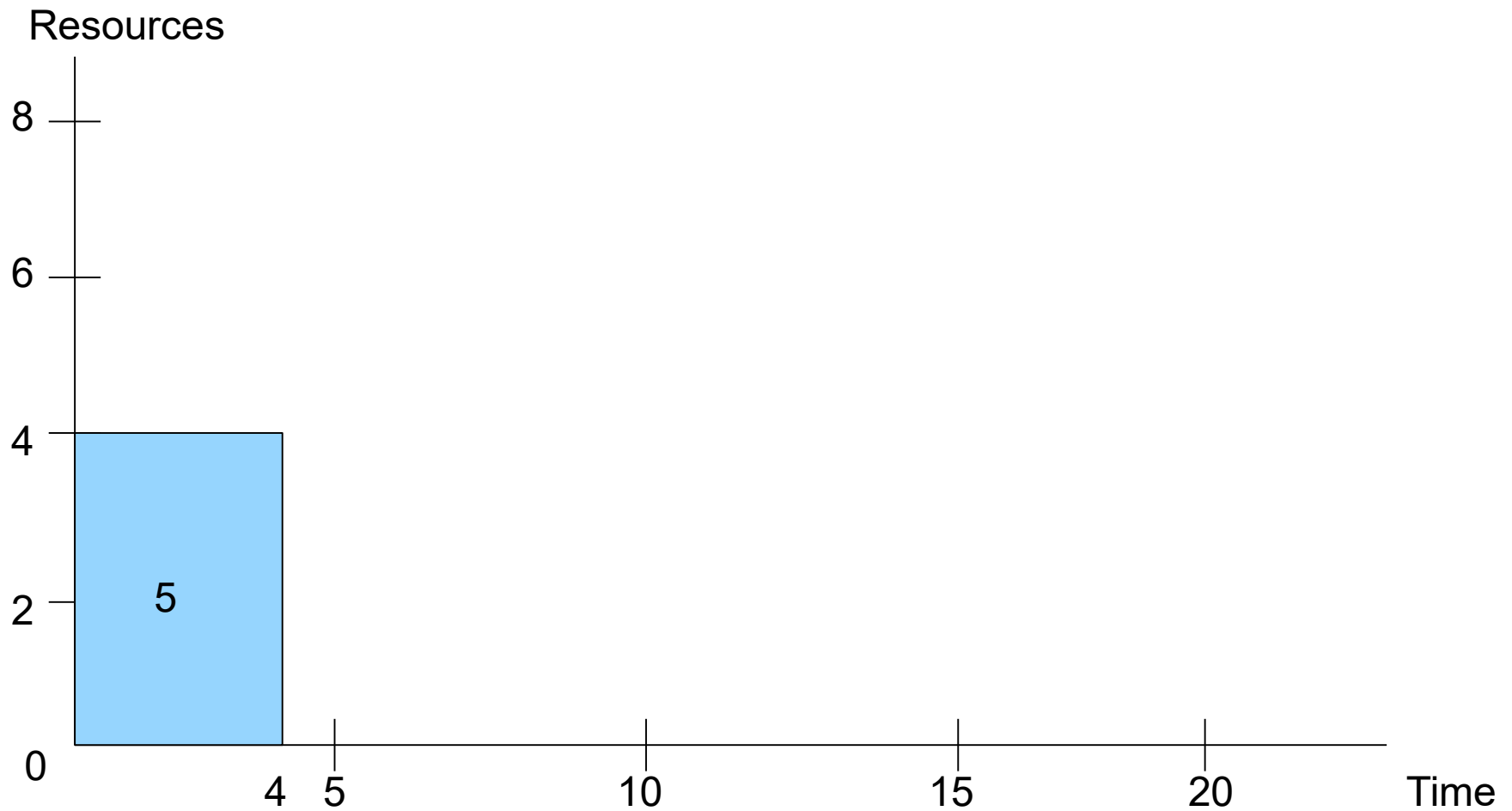


$$\text{Makespan} = 4$$

$$S = (1T1, 5T5)$$



Iteration 2 – Gantt Chart Illustration



Solving via Greedy Algorithm

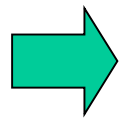
- Iteration 3.**

Since activity 3 requires the most resource units out of the remaining available units, it is chosen and assigned to TP T3:[0,7].

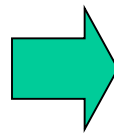
$$f(2)=2$$

$$f(3)=3$$

$$f(7)=2$$

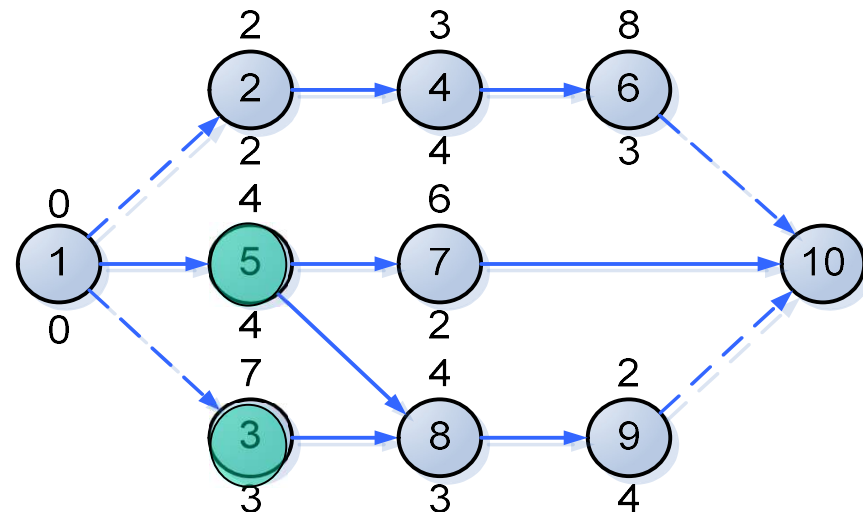


$$\text{Max } \{f(i)\}=3, \\ i=3$$

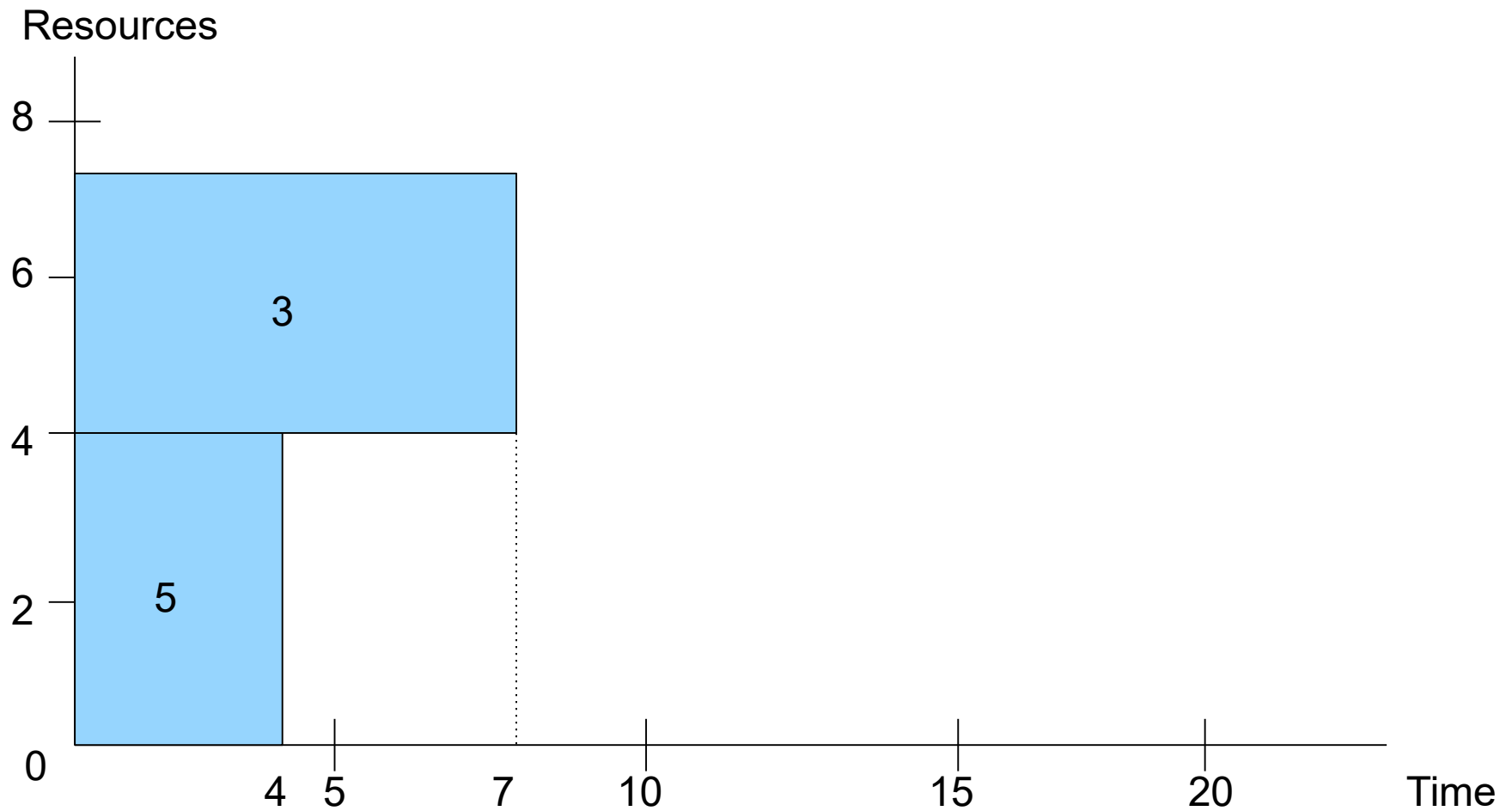


$$\text{Makespan} = 7$$

$$S = (1T1, 5T5, 3T3)$$



Iteration 3 – Gantt Chart Illustration



Solving via Greedy Algorithm

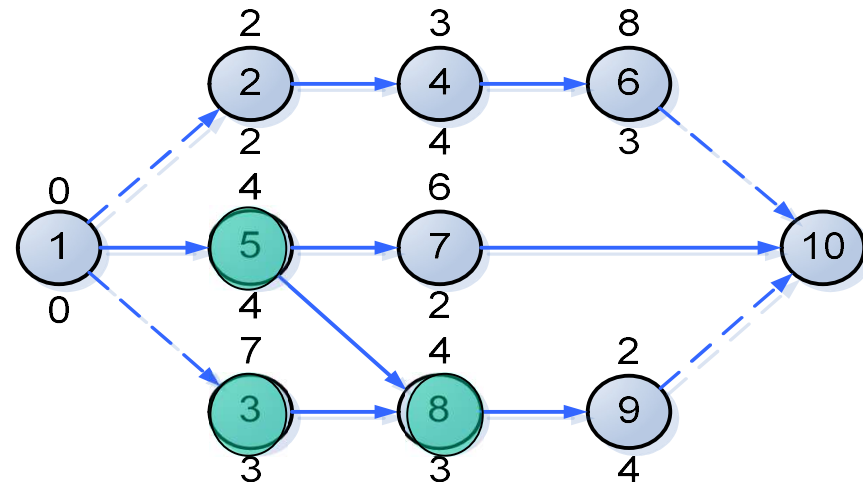
- Iteration 4.**

Since activity 8 requires the most resource units out of the remaining available units, it is chosen and assigned to TP T8:[7,11].

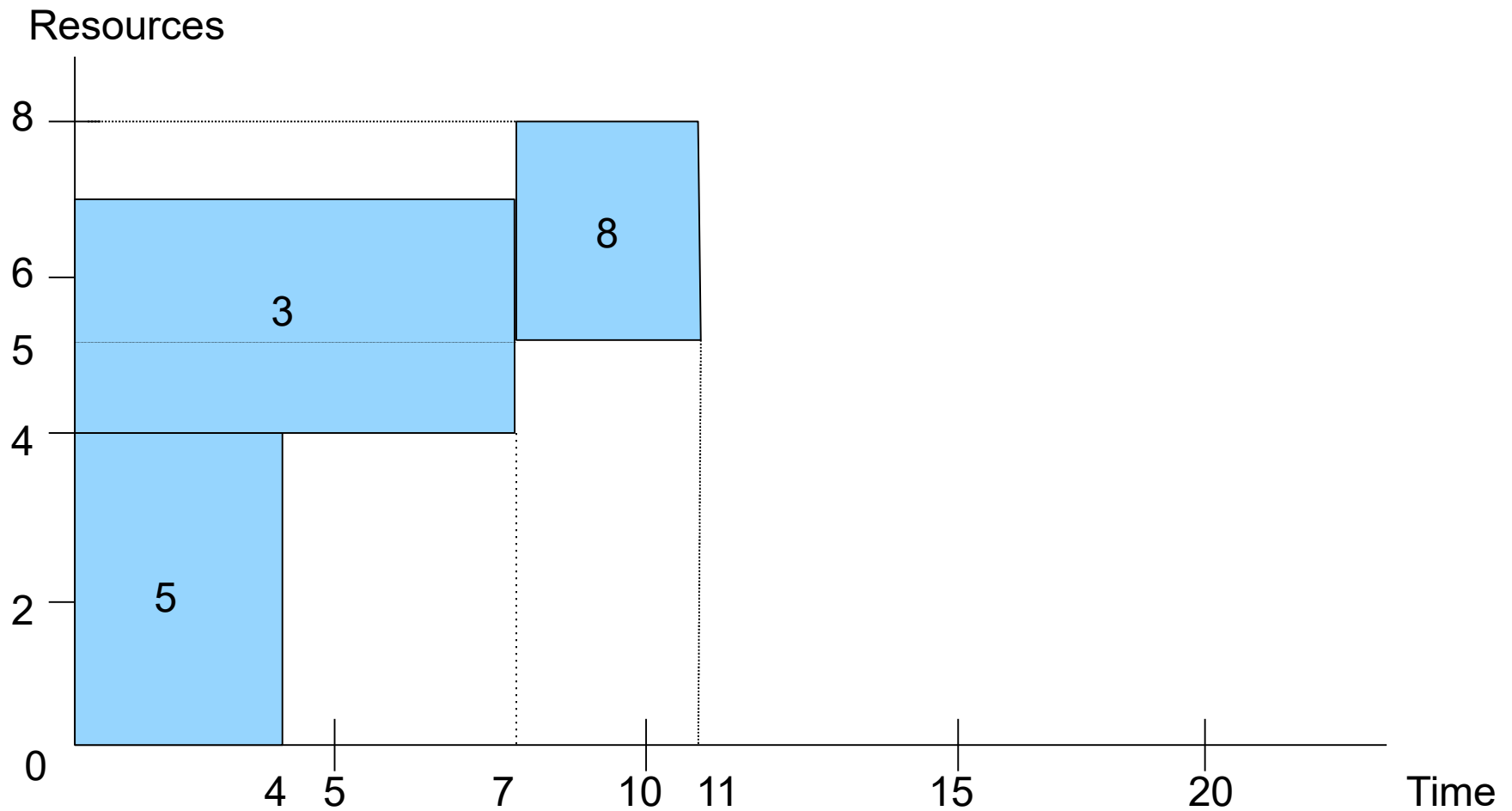
$f(2)=2$
 $f(7)=2$
 $f(8)=3$

$\Rightarrow \text{Max } \{f(i)\}=3, i=8 \Rightarrow \text{Makespan} = 11$

$S = (1T1, 5T5, 3T3, 8T8)$



Iteration 4 – Gantt Chart Illustration



Solving via Greedy Algorithm

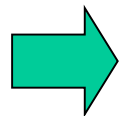
- Iteration 5.**

Since activity 9 requires the most resource units out of the remaining available units, it is chosen and assigned to TP T9:[11,13].

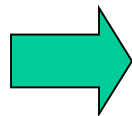
$$f(2)=2$$

$$f(7)=2$$

$$f(9)=4$$

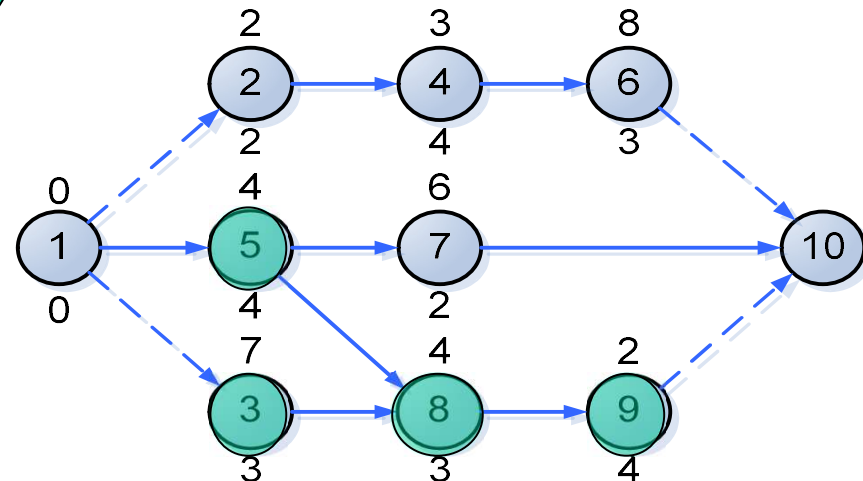


$$\text{Max } \{f(i)\}=3, \\ i=9$$

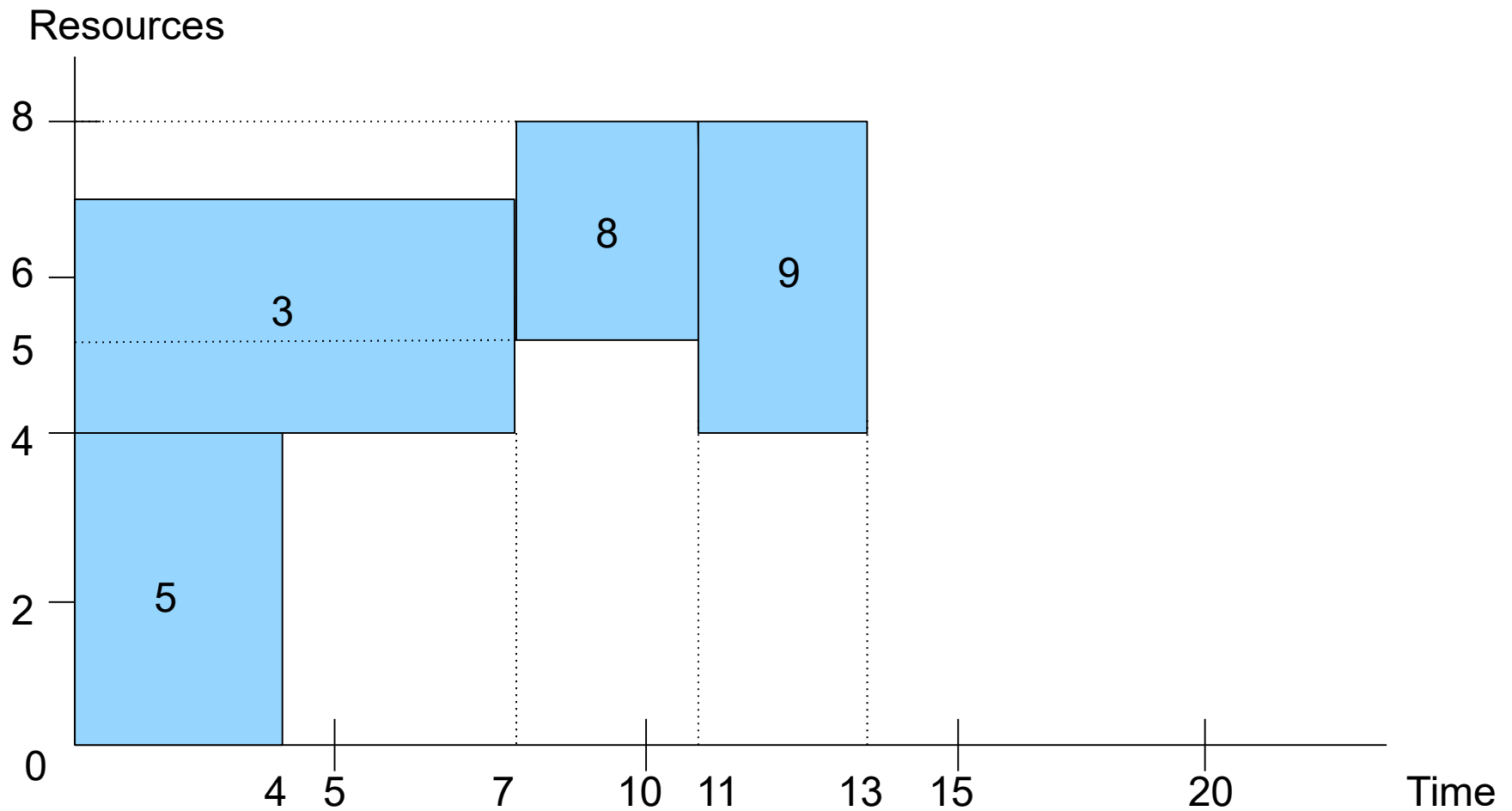


$$\text{Makespan} = 13$$

$$S = (1T1, 5T5, 3T3, 8T8, 9T9)$$



Iteration 5 – Gantt Chart Illustration

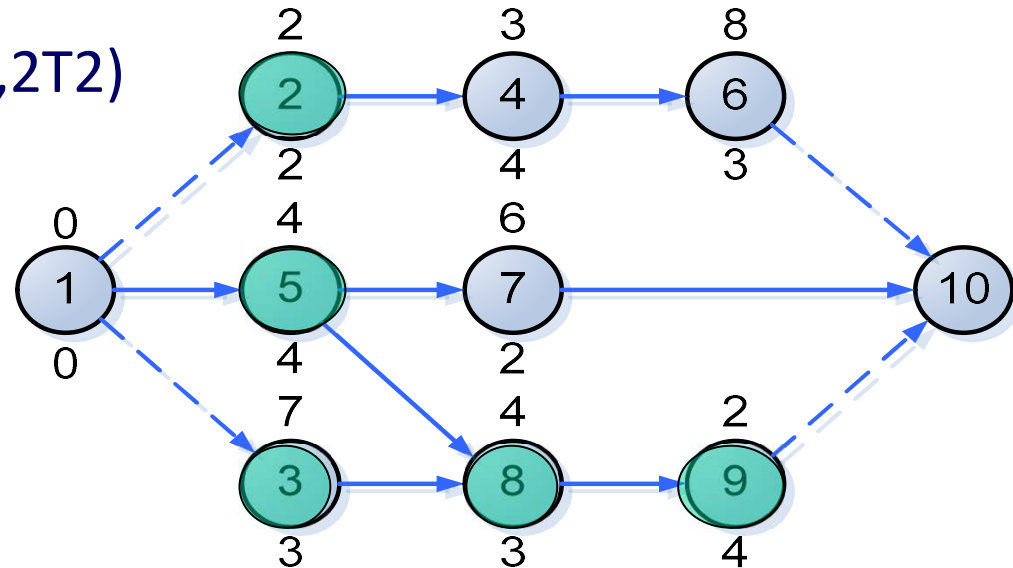


Solving via Greedy Algorithm

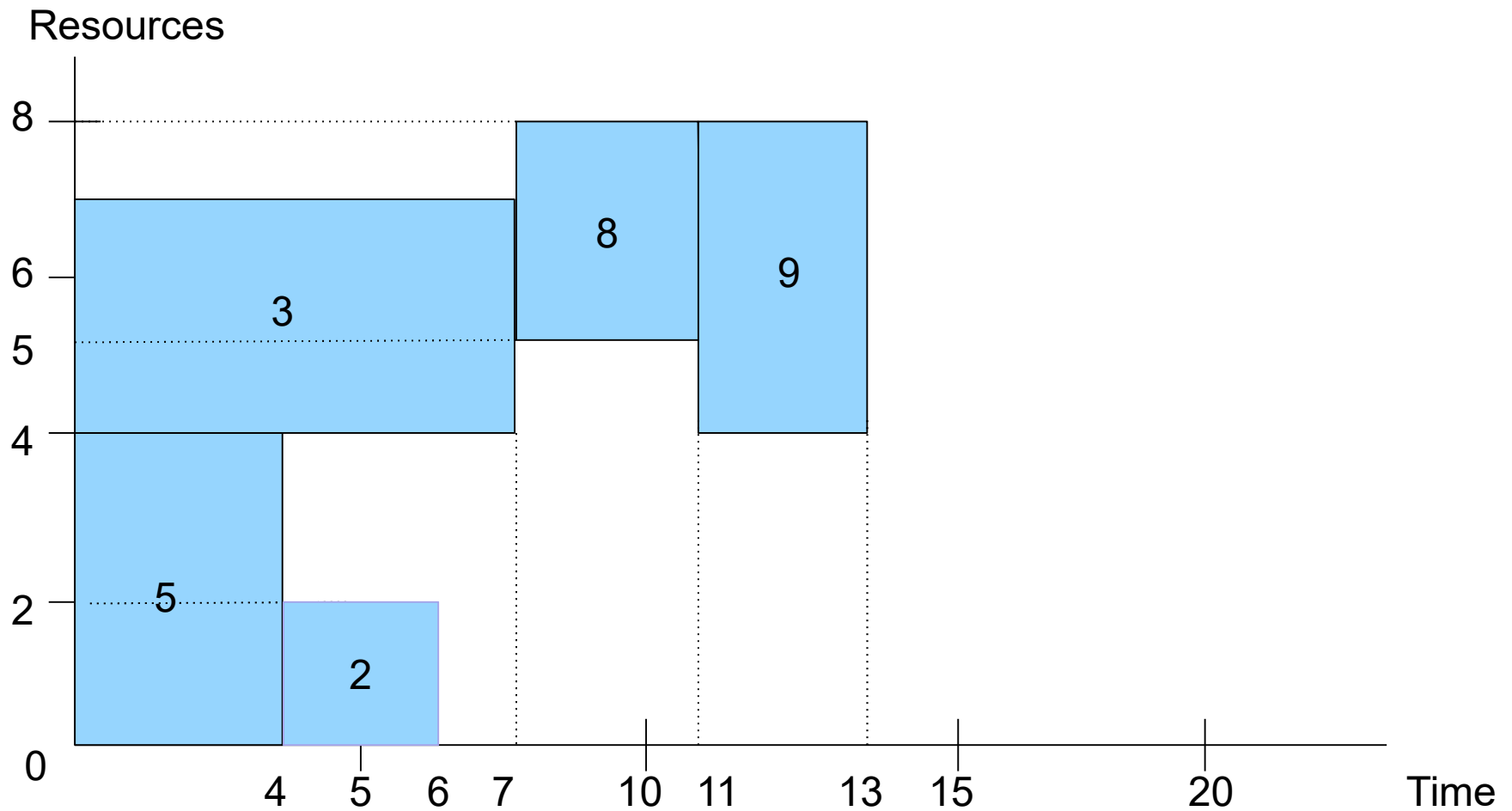
- Iteration 6.**

Since both activities 2 and 7 require the same number of resource units, we randomly select activity 2 and assign it to TP T2:[4,6] (As ET=4 is the earliest start time for this activity 4) and makespan remains 13 time units.

$S = (1T1, 5T5, 3T3, 8T8, 9T9, 2T2)$



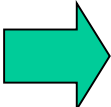

Iteration 6 – Gantt Chart Illustration



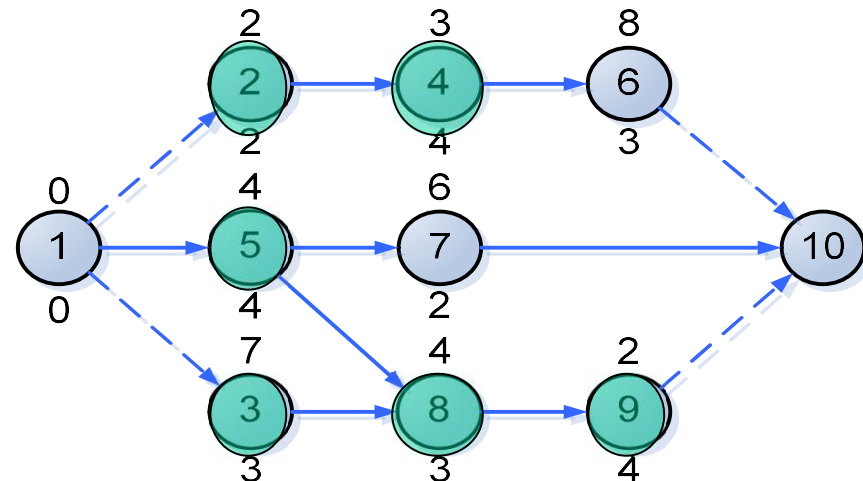
Solving via Greedy Algorithm

- Iteration 7.

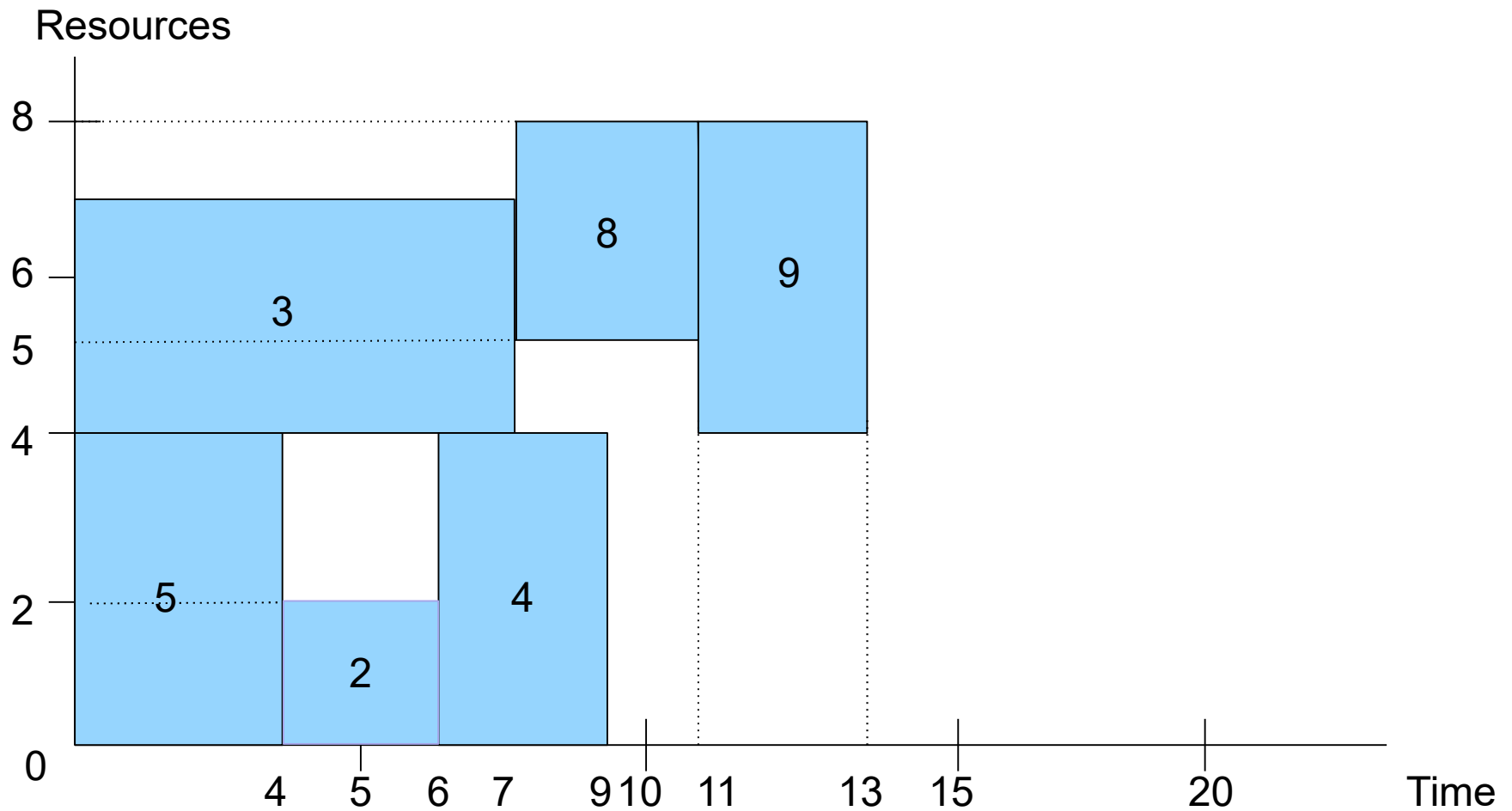
Since activity 4 requires the most resource units out of the remaining available units, it is chosen and assigned to TP T4:[6,9].

$f(4)=4$
 $f(7)=2$

 $\text{Max } \{f(i)\}=4,$
 $i=4$

 $\text{Makespan} = 13$

$S = (1T1, 5T5, 3T3, 8T8, 9T9, 2T2, 4T4)$



Iteration 7 – Gantt Chart Illustration



Solving via Greedy Algorithm

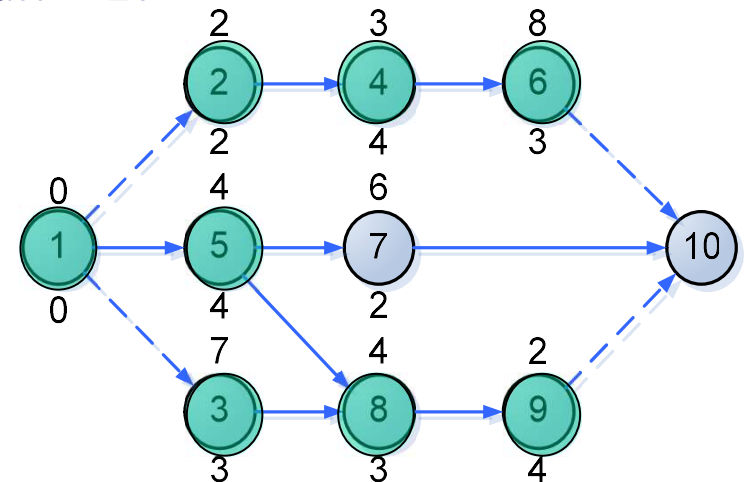
- Iteration 8.**

Since activity 6 requires the most resource units out of the remaining available units, it is chosen and assigned to TP T6:[9,17].

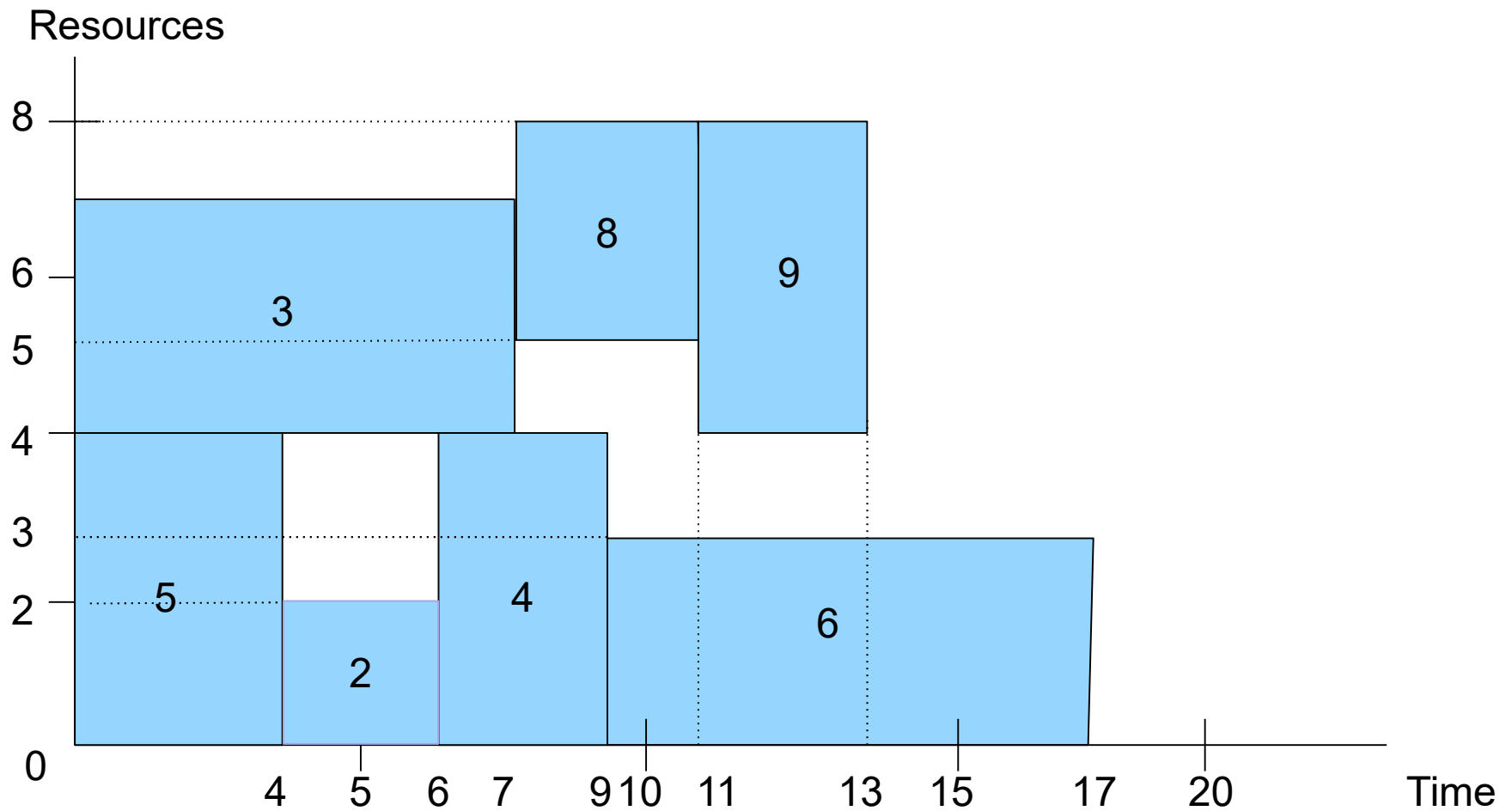
$f(6)=3$
 $f(7)=2$

$\Rightarrow \text{Max } \{f(i)\}=3, \quad i=6 \Rightarrow \text{Makespan} = 17$

$S = (1T1, 5T5, 3T3, 8T8, 9T9, 2T2, 4T4, 6T6)$



Iteration 8 – Gantt Chart Illustration



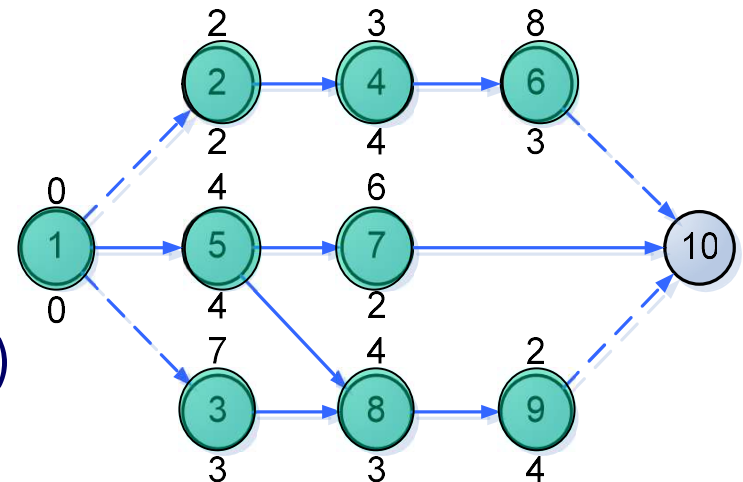
Solving via Greedy Algorithm

- Iteration 9.

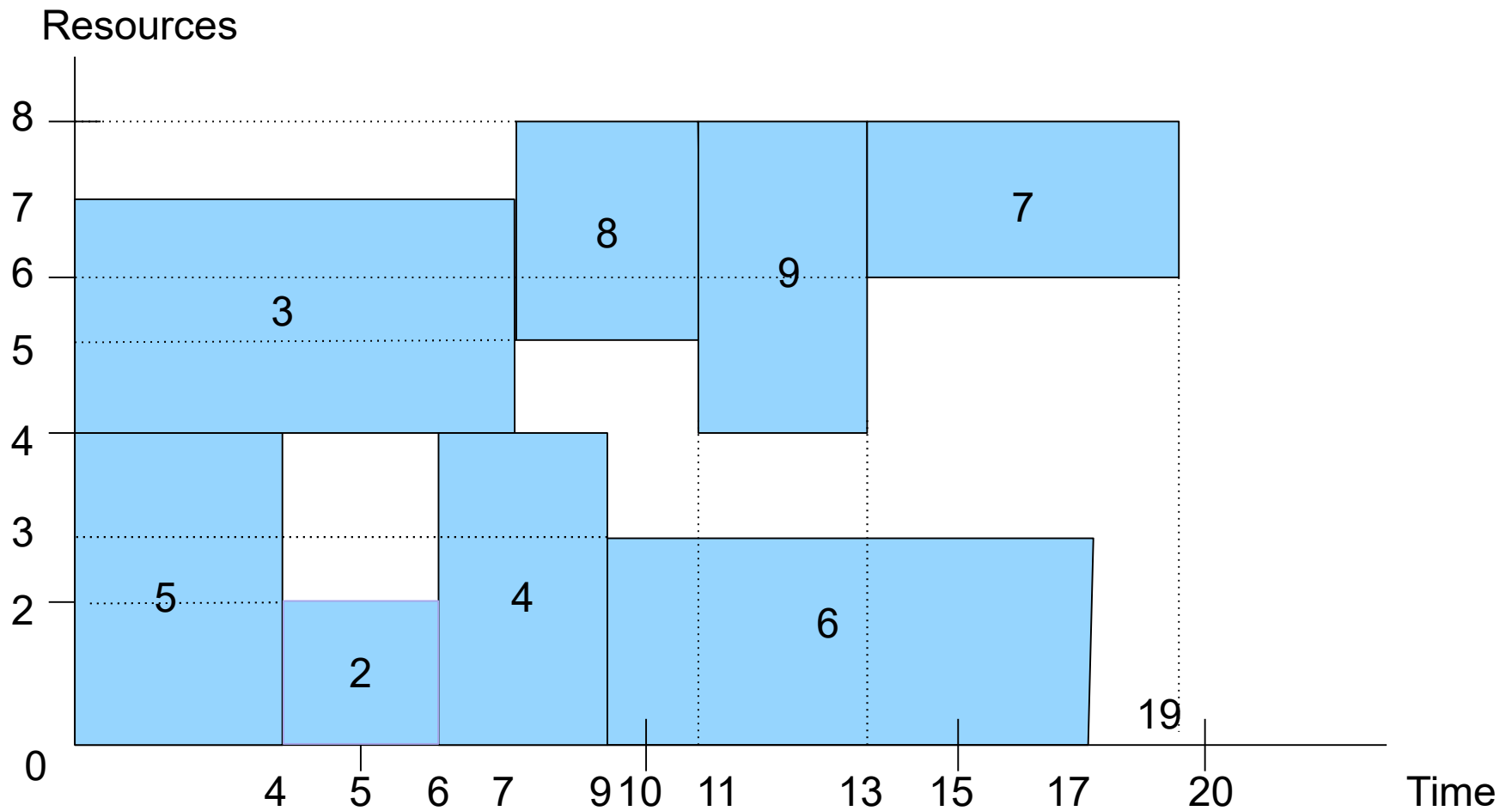
Since activity 7 requires the most resource units out of the remaining available units, it is chosen and assigned to TP T7:[13,19].

$f(7)=2 \Rightarrow \text{Max } \{f(i)\}=2, \Rightarrow \text{Makespan} = 19$
 $i=7$

$S = (1T1, 5T5, 3T3, 8T8, 9T9, 2T2, 4T4, 6T6, 7T7)$



Iteration 9 – Gantt Chart Illustration



Solving via Greedy Algorithm

- Iteration 10.

Activity 10 is assigned to TP T10:[19,19],

Therefore, the makespan = 19

$S = (1T1, 5T5, 3T3, 8T8, 9T9, 2T2, 4T4, 6T6, 7T7, 10T10)$

Exams Scheduling Problem

- The Athens University of Economics and Business must schedule the exams for the Department of Management Science and Technology. Assuming that the capacity is not a problem, one of the basic constraints is to not schedule simultaneously courses of the same year, courses that the same students are examined on and courses of the same professor.
- Given that the exams take place during 2-hour time periods (i.e., 8-10, 10-12, 12-2, 2-4, etc.), the **objective is to minimize the total time that the exams last** (this solves the capacity problem up to a point). Suggest a Greedy Algorithm for solving the problem.
- The next slide contains **the courses/subjects that cannot be examined at the same time** (e.g., exam 1 cannot be held at the same time period with exams 5 and 6, exam 2 cannot be held at the same time period with exam 3 και 5, etc.)

Exams Scheduling Problem

- Exam 1 \rightarrow 5,6
- Exam 2 \rightarrow 3,5
- Exam 3 \rightarrow 2,6,7
- Exam 4 \rightarrow 7
- Exam 5 \rightarrow 1,2,6,8,9
- Exam 6 \rightarrow 1,3,5,9,10
- Exam 7 \rightarrow 3,4,10,11,12
- Exam 8 \rightarrow 5
- Exam 9 \rightarrow 5,6,10
- Exam 10 \rightarrow 6,7,9
- Exam 11 \rightarrow 7
- Exam 12 \rightarrow 7,14
- Exam 13 \rightarrow 14,15
- Exam 14 \rightarrow 12,13,15
- Exam 15 \rightarrow 13,14

Exams Scheduling Problem – Greedy Algorithm

Solution structure: 15 assignments of an exam to a time period.

Solution element: An assignment of an exam to a time period

Exams Scheduling Problem – Greedy Algorithm

Selection Criterion:

The motivation of the selection criterion is the following: It will be attempted to schedule the **most exams possible** to the **less possible time periods**, aiming to minimize the total time consumed to complete all exams

Exams Scheduling Problem – Greedy Algorithm

Selection Criterion :

- Remember that for the bin packing problem, we have chosen to sort the items in descending size order.
- The idea was to firstly pick the bulky object, in order to then take advantage of the small empty spaces for the smaller items.
- The same idea may be applied to the exam scheduling problem
- Which are the “hardest” to schedule exams?
(Hardest to fit objects)
- Exams with the most conflicting exams that cannot be scheduled simultaneously
(Bulky objects)

Exams Scheduling Problem – Greedy Algorithm

Selection Criterion :

- Sort the exams in descending order of number of conflicting exams
- In each iteration choose the first exam in the sorted list
- Schedule/Assign the exam to the **first available** time period that the exam can be scheduled, without violating any conflicting exams constraints

Exams Scheduling Problem – Greedy Algorithm

Exams sorting in descending order of the number of conflicting exams

1.	Exam 5	1,2,6,8,9		Exam 2	3,5
	Exam 6	1,3,5,9,10		Exam 12	7,14
	Exam 7	3,4,10,11,12		Exam 13	14,15
2.	Exam 3	2,6,7		Exam 15	13,14
	Exam 9	5,6,10	4.	Exam 4	7
	Exam 14	12,13,15		Exam 8	5
	Exam 10	6,7,9		Exam 11	7
3.	Exam 1	5,6			

Exams Scheduling Problem – Greedy Algorithm

Iteration 1

Exam 5 is selected.

It is assigned to the first available time period X_1

Partial solution **S:(5X₁)**

Iteration 2

Exam 6 is selected.

It is assigned to the first available time period X_2

Partial solution **S:(5X₁, 6X₂)**

(The exam could not be scheduled to X_1 because it is conflicting with exam 5)

Exams Scheduling Problem – Greedy Algorithm

Iteration 3

Exam 7 is selected.

It is assigned to the first available time period X_1 (no conflicting constraints are violated)

Partial solution $S:(5X_1, 6X_2, 7X_1)$

Iteration 4

Exam 3 is selected.

It is assigned to the first available time period X_3

(Exam 3 cannot be scheduled to X_1 due to conflict with exam 7, and also cannot be scheduled to X_2 as it conflicts with exam 6)

Partial solution $S:(5X_1, 6X_2, 7X_1, 3X_3)$

Exams Scheduling Problem – Greedy Algorithm

Iteration 5

Exam 9 is selected.

It is assigned to the first available time period X_3

(Exam 9 cannot be scheduled to X_1 due to conflict with exam 5, and also cannot be scheduled to X_2 as it conflicts with exam 6)

Partial solution **S:(5X₁, 6X₂, 7X₁, 3X₃, 9X₃)**

Iteration 6

Exam 14 is selected.

It is assigned to the first available time period X_1

Partial solution **S:(5X₁, 6X₂, 7X₁, 3X₃, 9X₃, 14X₁)**

Exams Scheduling Problem – Greedy Algorithm

Iteration 7

Exam 10 is selected.

It is assigned to the first available time period X_4

(X_1 : conflict with 7, X_2 : conflict with 6, X_3 : conflict 9)

Partial solution $S:(5X_1, 6X_2, 7X_1, 3X_3, 9X_3, 14X_1, 10X_4)$

Iteration 8

Exam 1 is selected.

It is assigned to the first available time period X_3

(X_1 : conflict with 5, X_2 : conflict with 6)

Partial solution $S:(5X_1, 6X_2, 7X_1, 3X_3, 9X_3, 14X_1, 10X_4, 1X_3)$

Exams Scheduling Problem – Greedy Algorithm

Iteration 9

Exam 2 is selected.

It is assigned to the first available time period X₂

(X₁: conflict with 5)

Partial solution S:(5X₁, 6X₂, 7X₁, 3X₃, 9X₃, 14X₁, 10X₄, 1X₃, 2X₂)

Iteration 10

Exam 12 is selected.

It is assigned to the first available time period X₂

(X₁: conflict with 7)

Partial solution S:(5X₁, 6X₂, 7X₁, 3X₃, 9X₃, 14X₁, 10X₄, 1X₃, 2X₂, 12 X₂)

Exams Scheduling Problem – Greedy Algorithm

Iteration 11

Exam 13 is selected.

It is assigned to the first available time period X₂

(X₁: conflict with 14)

Partial solution :

S:(5X₁, 6X₂, 7X₁, 3X₃, 9X₃, 14X₁, 10X₄, 1X₃, 2X₂, 12 X₂, 13X₂)

Iteration 12

Exam 15 is selected.

It is assigned to the first available time period X₃

(X₁: conflict with 14, X₂: conflict with 13)

Partial solution :

S:(5X₁, 6X₂, 7X₁, 3X₃, 9X₃, 14X₁, 10X₄, 1X₃, 2X₂, 12 X₂, 13X₂, 15X₃)

Exams Scheduling Problem – Greedy Algorithm

Iteration 13

Exam 4 is selected.

It is assigned to the first available time period X₂

(X₁: conflict with 7)

Partial solution :

S:(5X₁, 6X₂, 7X₁, 3X₃, 9X₃, 14X₁, 10X₄, 1X₃, 2X₂, 12 X₂, 13X₂, 15X₃, 4X₂)

Iteration 14

Exam 8 is selected.

It is assigned to the first available time period X₂

(X₁: conflict with 5)

Partial solution :

S:(5X₁, 6X₂, 7X₁, 3X₃, 9X₃, 14X₁, 10X₄, 1X₃, 2X₂, 12 X₂, 13X₂, 15X₃, 4X₂, 8X₂)

Exams Scheduling Problem – Greedy Algorithm

Iteration 15

Exam 11 is selected.

It is assigned to the first available time period X2

(X1: conflict with 7)

Final solution:

S:(5X1, 6X2, 7X1, 3X3, 9X3, 14X1, 10X4, 1X3, 2X2, 12 X2, 13X2, 15X3, 4X2, 8X2, 11X2)

Exams Scheduling Problem – Greedy Algorithm

Therefore, the solution is

**S:(5X₁, 6X₂, 7X₁, 3X₃, 9X₃, 14X₁, 10X₄, 1X₃,
2X₂, 12 X₂, 13X₂, 15X₃, 4X₂, 8X₂, 11X₂)**

- The first time slot (8-10) will host exams 5 ,7, 14
- The second time slot (10-12) will host exams 2, 4, 6, 8,11, 12, 13
- The third time slot (12-2) will host exams 1, 3, 9, 15
- The fourth time slot (2-4) will host exams 10

Therefore, 8 hours (i.e., 4 slots) are required to schedule all exams of Department of Management Science and Technology.

Graph Coloring Problem

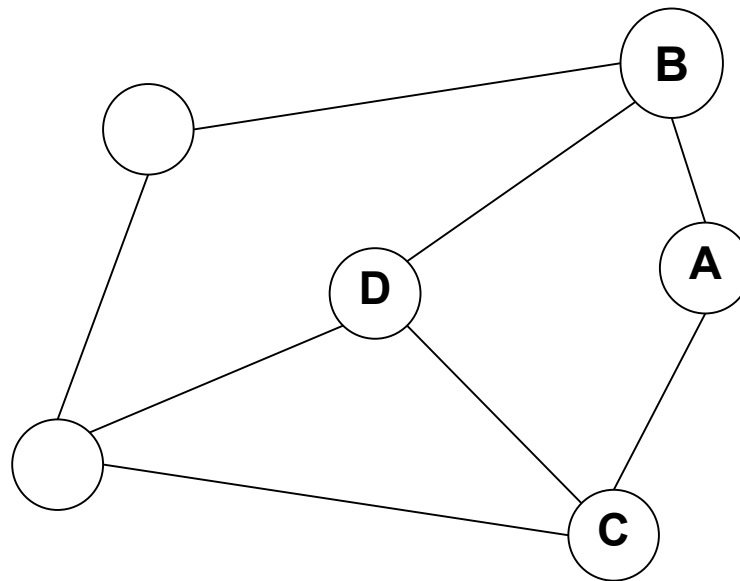
- Let a graph $G = \{V, E\}$
 - Set V : nodes of the network/graph
 - Set E : connections/edges of the graph nodes
- Coloring of a graph:
 - Assignment of every graph node to a color
 - Constraint: Two neighboring nodes (nodes connected with an edge) cannot be assigned to the same color
- Graph Coloring Problem:
 - Decision Problem: Is it possible to color the graph with only k colors;
 - Optimization Problem: Which is the minimum number of colors that must be used to color the graph;
- The Decision Problem is closely connected to the Optimization Problem
 - Obviously, if we can answer the question of the decision problem for every k , we can also solve the optimization problem

Graph Coloring Problem

- Optimization problem objective: Color number
- Objective function: the number of different colors required
- Objective function minimization = color number

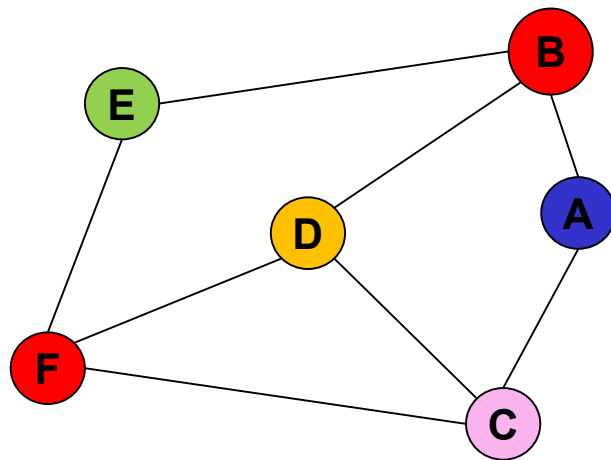
Graph Coloring Problem

- Node degree: The number of edges starting (ending) from (to) this node
 - Degree of A = 2
 - Degree of D = 3

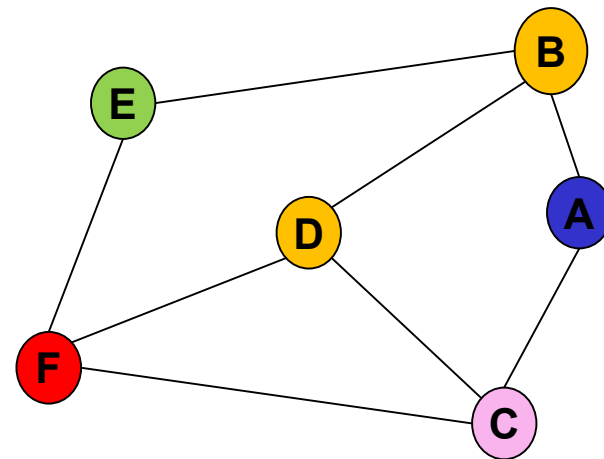


Graph Coloring Problem – Coloring conflicts

Valid coloring



Invalid Coloring:
Connected nodes B and D
share the same color



Graph Coloring Problem

- How is the Graph Coloring problem connected to the Exam Scheduling Problem of the Athens University of Economics and Business?

Graph Coloring Problem – Greedy Algorithms

- The basic scheme of Greedy Algorithms for the Graph Coloring Problem
 - Given a specific color order, iteratively assign a color to a node
 - Color selection:
 - The first of the already existing colors that results in a valid non violating coloring
 - The color that minimizes the conflicts for the already existing colors
 - Obviously, if there is no already existing color available, a new color is selected
 - Order selection
 - Basic scheme (initial node order)
 - Welsh Powell algorithm (descending node degree order)

Product shelf distribution

- The extremely competitive global market has changed the rules. Therefore, retailers seek to diversify and secure competitive advantage. The adoption of an effective product distribution system to the shelves is the key to to it.
- The Shelf-Space Allocation Problem – SSAP examines the optimal distribution of products to shelves with respect to maximizing the profit, the customer satisfaction and the out-of-stock cases.
- At the same time, the spot that the products are placed and the number of units affect the customer choice.
- For all above reasons, retailers have a great interest for effective product distribution to shelved.

Shelf-space allocation problem

The Shelf-Space Allocation problem can be described as follows: Assume a retailer, e.g. a supermarket. Assume also the following:

- k : a specific shelf, $k = 1, 2, \dots, m$
- m : the number of available shelves
- T_k : the length of shelf k , $k = 1, 2, \dots, m$
- i : a specific product, $i = 1, 2, \dots, n$
- α_i : the length of every unit of a product i
- L_i : the minimum number of units of product i that must be placed on the shelf
- U_i : the maximum number of units of product i that must be placed on the shelf
- A_i : The total available units of product i
- P_i : The per unit profit of product i in a shelf

Shelf-space allocation problem

- The length of all products that are placed on the shelf must not exceed the total shelf length.
- The number of units of a product i in a shelf k must be within the minimum and maximum number of units of the product i .
- The number of units of a product i in a shelf k receives integer values.
- The aim of the problem is the optimal distribution of different products to the specified shelves of the retailer (supermarket).
- The Shelf-Space Allocation problem's objective function can be selected with respect to the aim: profit maximization, shelf utilization, product diversification, etc.

Shelf-space allocation problem: Example

- $m = 2$ available shelves
- The length of each shelf $T_1 = 200$ και $T_2 = 200$
- i = a specific product, with: $i = 1, 2, \dots, 6$
- k = a specific shelf, with: $k = 1, 2$
- The per unit profit of the retailer for product i in shelf k is the same independently of the shelf.
- Below are the information for products $i = 1, 2, \dots, 6$:

Product	Units (A_i)	α_i	P_i	L_i	U_i
No.1	3	20	100	0	2
No.2	4	40	800	0	2
No.3	7	30	300	0	4
No.4	2	50	700	0	2
No.5	5	20	200	0	4
No.6	4	50	400	0	3

T_k : the length of each shelf k

α_i : the length of each unit of a specific product

P_i : the retailer's profit for each product unit i in some shelf

L_i = the minimum number of units of product i that must be available to a specific shelf

U_i = the maximum number of units of product i that must be available to a specific shelf

Shelf-space allocation problem: Example

The objective is to find a solution that maximizes the profit, via a Greedy Algorithm

Shelf-space allocation problem: Example

- **Solution structure:**
 - The solutions structure is a number of assignments of product units to the 2 shelves.
- **Solution element:**
 - An assignment of a product unit to a shelf
- **Selection criterion** of the candidate solution element:
 - Calculate the fraction P_i/α_i for each product and order the products with respect these fractions (descending order). In each iteration add to the partial solution a valid assignment of a non already placed unit of a product i to a shelf. For each iteration, the unit of product i is the product with the largest fraction P_i/α_i .
- **Objective function:**
 - The total per unit profit of the retailer from products distribution to shelves.

Shelf-space allocation problem: Example

- Calculate fractions P_i/α_i and sort them in descending order.

Order	Product	Units	P_i / α_i	L_i	U_i
1	No.2	4	20	0	2
2	No.4	2	14	0	2
3	No.3	7	10	0	4
4	No.5	5	10	0	4
5	No.6	4	8	0	3
6	No.1	3	5	0	2

Remaining capacity in shelf 1 after positioning the first unit of product No.2

Iteration 1:

Shelf 1 (160) ←	P2(40)
Shelf 2 (200)	

S: (s1p2)

Iteration 2 :

Shelf 1 (120)	P2(40)	P2(40)
Shelf 2 (200)		

S: (s1P2,s1P2)

Iteration 3:

Shelf 1 (120)	P2(40)	P2(40)
Shelf 2 (160)	P2(40)	

S: (s1P2,s1P2,s2P2)

Iteration 4:

Shelf 1 (120)	P2(40)	P2(40)
Shelf 2 (120)	P2(40)	P2(40)

S: (s1P2,s1P2,s2P2,s2P2)

Iteration 5:

Shelf 1(70)	P2(40)	P2(40)	P4(50)
Shelf 2(120)	P2(40)	P2(40)	

S: (s1P2,s1P2,s2P2,s2P2,s1P4)

Iteration 6:

Shelf 1(20)	P2(40)	P2(40)	P4(50)	P4(50)
Shelf 2(120)	P2(40)	P2(40)		

S: (s1P2,s1P2,s2P2,s2P2,s1P4,s1P4)

Iteration 7:

Shelf 1(20)	P2(40)	P2(40)	P4(50)	P4(50)
Shelf 2(90)	P2(40)	P2(40)	P3(30)	

S: (s1P2,s1P2,s2P2,s2P2,s1P4,s1P4,s2P3)

Iteration 8:

Shelf 1(20)	P2(40)	P2(40)	P4(50)	P4(50)
Shelf 2(60)	P2(40)	P2(40)	P3(30)	P3(30)

S: (s1P2,s1P2,s2P2,s2P2,s1P4,s1P4,s2P3,s2P3)

Iteration 9:

Shelf 1(20)	P2(40)	P2(40)	P4(50)	P4(50)	
Shelf 2(30)	P2(40)	P2(40)	P3(30)	P3(30)	P3(30)

S: (s1P2,s1P2,s2P2,s2P2,s1P4,s1P4,s2P3,s2P3, s2P3)

Iteration 10:

Shelf 1(20)	P2(40)	P2(40)	P4(50)	P4(50)		
Shelf 2(0)	P2(40)	P2(40)	P3(30)	P3(30)	P3(30)	P3(30)

S: (s1P2,s1P2,s2P2,s2P2,s1P4,s1P4,s2P3,s2P3,s2P3,s2P3)

Iteration 11:

Shelf 1 (0)	P2(40)	P2(40)	P4(50)	P4(50)	P5(20)	
Shelf 2 (0)	P2(40)	P2(40)	P3(30)	P3(30)	P3(30)	P3(30)

S: (s1P2,s1P2,s2P2,s2P2,s1P4,s1P4,s2P3,s2P3, s2P3,s2P3,s1P5)

$$\begin{aligned} z(S) &= 4 \cdot 800 + 2 \cdot 700 + 4 \cdot 300 + 1 \cdot 200 = \\ &= 3200 + 1400 + 1200 + 200 = 6000 \end{aligned}$$