



Lecture 11

Produced by Dr. Worldwide
Welcome to the 305

Ex: Give-back Weekend



- The Student Government Association (SGA) organizes a recurring event called “Give-Back Weekends” where teams are formed to work on projects for residents in the university community
- This event occurs over four consecutive Saturdays in April
- Coed teams are formed with 3 to 5 students from various dormitory groups, fraternities, sororities, clubs, and organizations
- Residents of the community fill out a form to describe work at their home that needs to be done
- Time to complete a project will vary between teams because of the different number of team members, skills of the team, and physical make-up of the team



Ex: Give-back Weekend



- Time estimates (in hours) submitted by the six teams available to work on 12 different projects for the first Saturday of the event

Team	Jobs											
	1	2	3	4	5	6	7	8	9	10	11	12
1	5	1.5	6	4	3.5	3	6	1.5	5	1	3	3.5
2	4	2	5	5	3	3	5.5	2	4	1.5	4	2.5
3	5	1.5	6.5	3.5	2.5	4	4.5	3	3.5	1	3.5	4
4	3.5	2	5.5	4	3.5	2.5	5	2.5	4	1.5	2.5	4
5	3.5	3	5	3	2	4	5	2	5	2	4	3
6	4	2.5	6	5	3	3	6	3	3	2	3	3.5

- The primary objective of SGA is to complete all 12 projects

Ex: Give-back Weekend



- Teams can work on multiple projects
- Teams cannot work more than 8 hours on Saturday
- Each team should work on at least one project
- Alternative Questions
 - Q1: How can we assign the 6 teams to the 12 projects to maximize the number of jobs completed on Saturday?
 - Q2: How can we assign the 6 teams to the 12 projects and minimize the total time required for all 6 teams to complete all projects?

Ex: Give-back Weekend



- Consider the jobs as the “sources” or “supply”
- Consider the teams as the “destinations” or demand
- Decision variables
 - $x_{ij} = \begin{cases} 1 & \text{if job } j \text{ is assigned to team } i \\ 0 & \text{otherwise} \end{cases}$
 - $i \in \{1, 2, 3, 4, 5, 6\}$
 - $j \in \{1, 2, 3, \dots, 12\}$

- Objective function for the number of completed jobs

$$Z = \sum_{i=1}^6 \sum_{j=1}^{12} x_{ij}$$

- Objective function for the amount of time for the teams to do all the jobs

$$Z = \sum_{i=1}^6 \sum_{j=1}^{12} t_{ij} x_{ij} \quad \text{where } t_{ij} = \text{time required for team } i \text{ to do job } j$$

Ex: Give-back Weekend



- Constraints

- Each team cannot work more than 8 hours

$$5x_{11} + 1.5x_{12} + 6x_{13} + 4x_{14} + 3.5x_{15} + 3x_{16} + 6x_{17} + 1.5x_{18} + 5x_{19} + x_{110} + 3x_{111} + 3.5x_{112} \leq 8 \quad (\text{Team } 1)$$

- Each project can only be assigned to at most one team, which adds a total of 12 constraints, one for each project $j \in \{1, 2, \dots, 12\}$

$$\sum_{i=1}^6 x_{ij} = x_{1j} + x_{2j} + x_{3j} + x_{4j} + x_{5j} + x_{6j} \leq 1 \quad (\text{Project } j)$$

- Each project is assigned to exactly one team

$$\sum_{i=1}^6 x_{ij} = x_{1j} + x_{2j} + x_{3j} + x_{4j} + x_{5j} + x_{6j} = 1 \quad (\text{Project } j)$$

- Each decision variable is binary

$$x_{ij} \in \{0, 1\}$$

Ex: Give-back Weekend



- Download [GiveBack.xlsx](#) from course website from link [Sheet 1](#)
- Solution for Q1 on sheet called [Max projects](#)
 - Maximize the number of projects that can be completed

Team	Jobs assigned	Time required to complete
1	8	1.5
2	3, 10	6.5
3	9, 12	7.5
4	7, 11	7.5
5	2, 4, 5	8
6	1,6	7
Total time:		38

- In general, there can be [multiple optimal solutions](#) in assignment problems
- Could have assigned job 4 to Team 1 instead of Team 5
- Important is the fact that an optimal solution exists to do all 12 jobs

Ex: Give-back Weekend



- Solution for Q2 on sheet called **Min time**
 - We are trying to identify if a faster way exists where all jobs will still be completed
 - Minimize the total time required

Team	Jobs assigned	Time required to complete
1	2, 8	3
2	3, 12	7.5
3	5, 7, 10	8
4	6, 11	5
5	1, 4	6.5
6	9	3
Total time:		33

- Q: Do you notice any differences in the optimal solution?
- Q: Which solution is better?

Network Flow Models

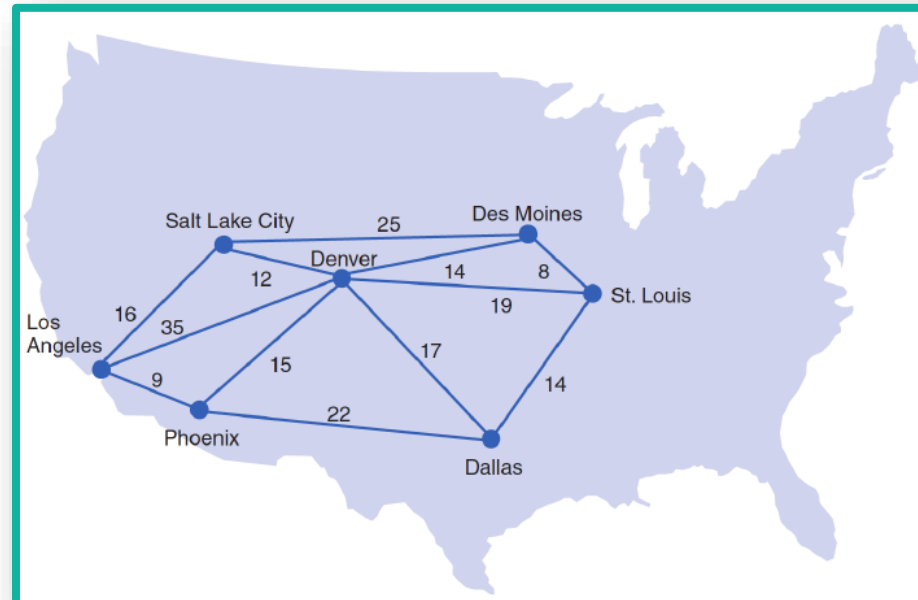


- A **network** is an arrangement of paths connected at various points
- A network has two main components
 - **Nodes** represent junction points
 - **Branches** or **edges** represent routes between nodes
- Diagrams can be utilized to visualize networks
 - Nodes represented by **circles**
 - Branches represented by **lines**
- Typical goals of transportation problems
 - Minimize transportation costs
 - Minimize total distance travelled
 - Maximize amount transported

Ex: Shortest Shipping Route



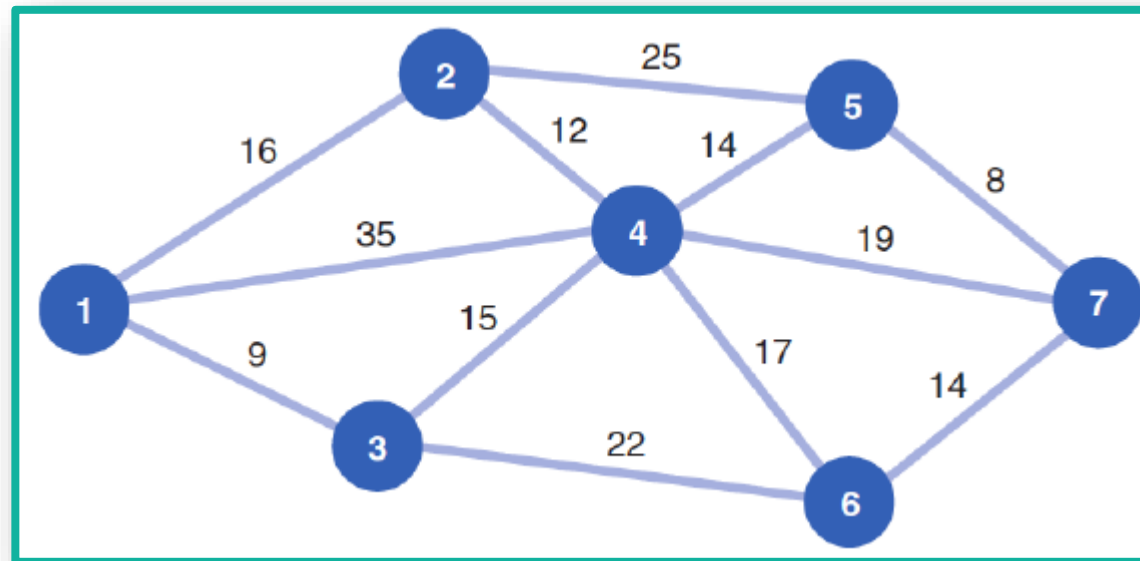
- The **shortest route problem** is to determine the shortest distance between an originating point and several destination points
- Stagecoach Shipping Company transports oranges from Los Angeles
 - Uses 6 trucks to travel to 6 different cities in the West and Midwest
 - Diagram of network with length of time (hours) between cities for transport



Ex: Shortest Shipping Route



- In this problem, we are looking for the shortest route based on **time**
- Simplified network

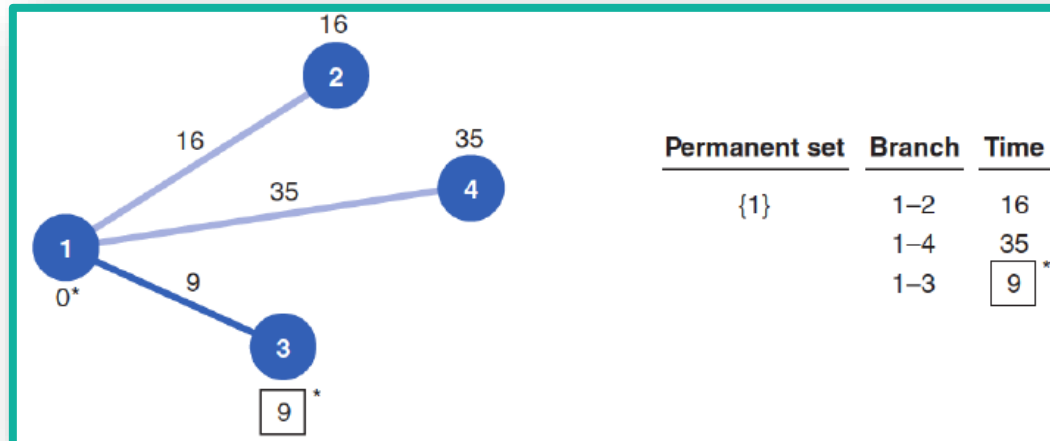


- Q: What is the shortest route from node 1 to each of the other nodes?

Ex: Shortest Shipping Route



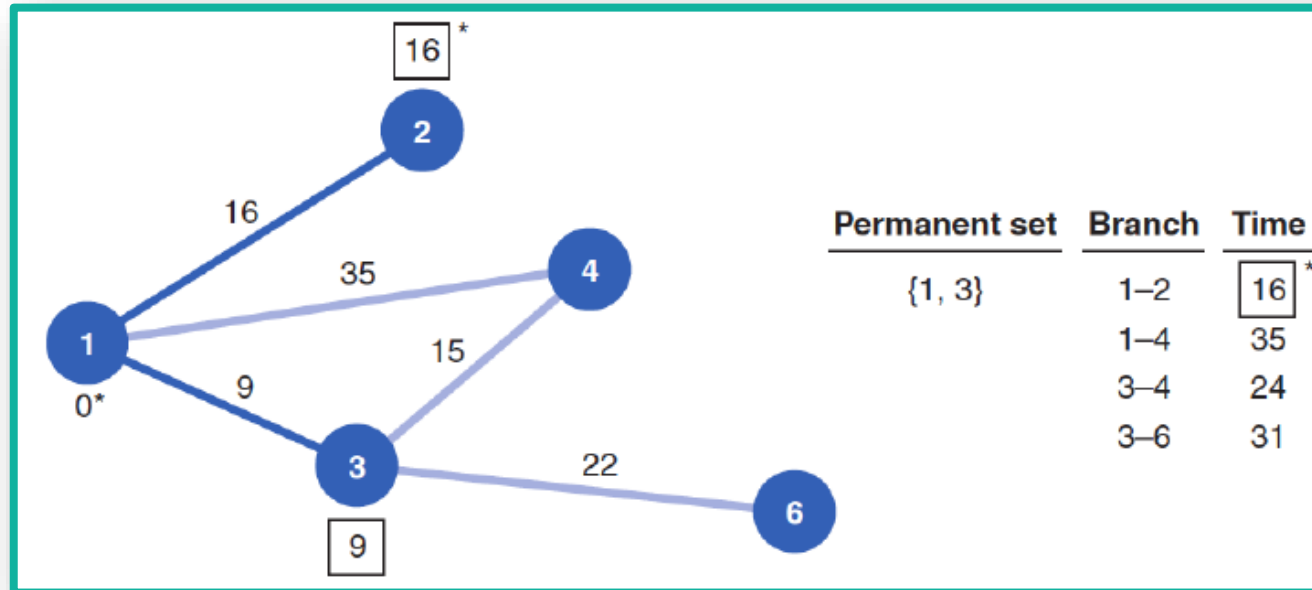
- We will keep track of the paths we choose by defining, and then updating a subset of the nodes called the **permanent set**
- To start, define the permanent set to be the **origin**, node 1
- The shortest path from node 1 to any of its adjacent nodes
- Node 3 is the closest to node 1
- Add node 3 to the permanent set



Ex: Shortest Shipping Route



- Next, explore all the nodes adjacent to nodes in the permanent set {1,3}

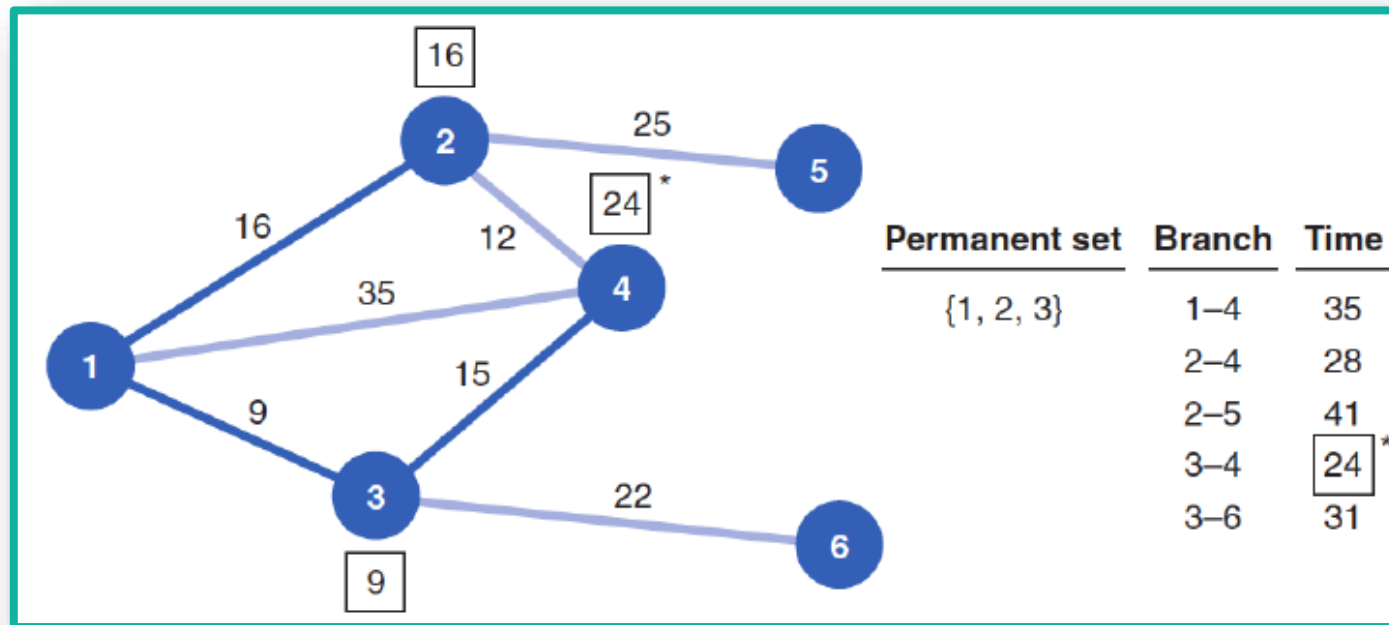


- Node 2 is the closest to the origin, so it is added to the permanent set

Ex: Shortest Shipping Route



- Next, explore all the nodes adjacent to nodes in the permanent set {1,2,3}

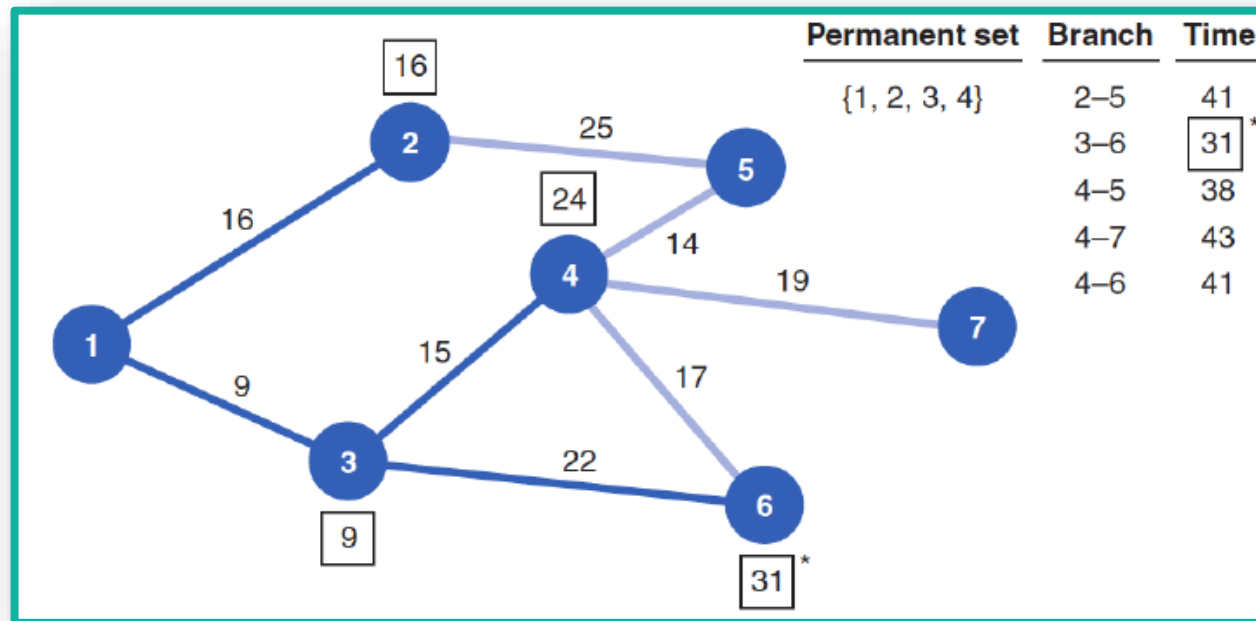


- Node 4 is the closest to the origin among {4,5,6} and added to the permanent set

Ex: Shortest Shipping Route



- Next, explore all the nodes adjacent to nodes in the permanent set {1, 2, 3, 4}

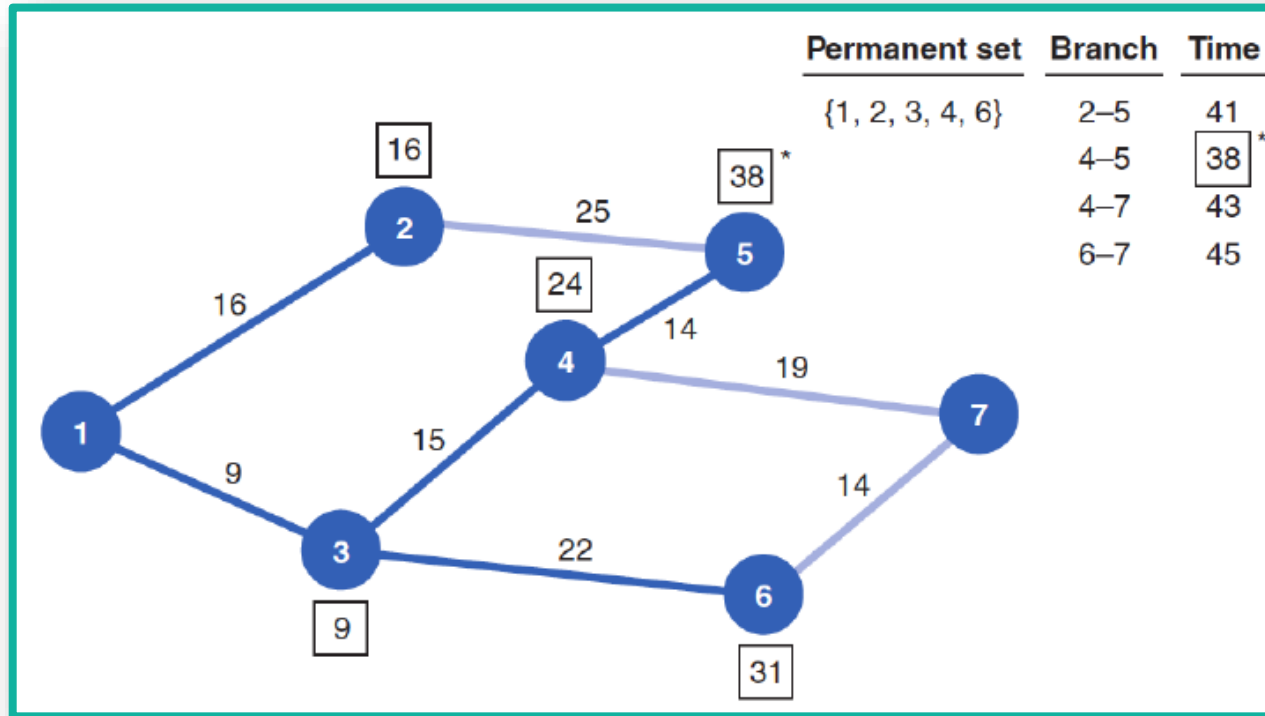


- Node 6 is the closest to the origin and added to the permanent set

Ex: Shortest Shipping Route



- Next, explore all the nodes adjacent to nodes in the permanent set {1, 2, 3, 4, 6}

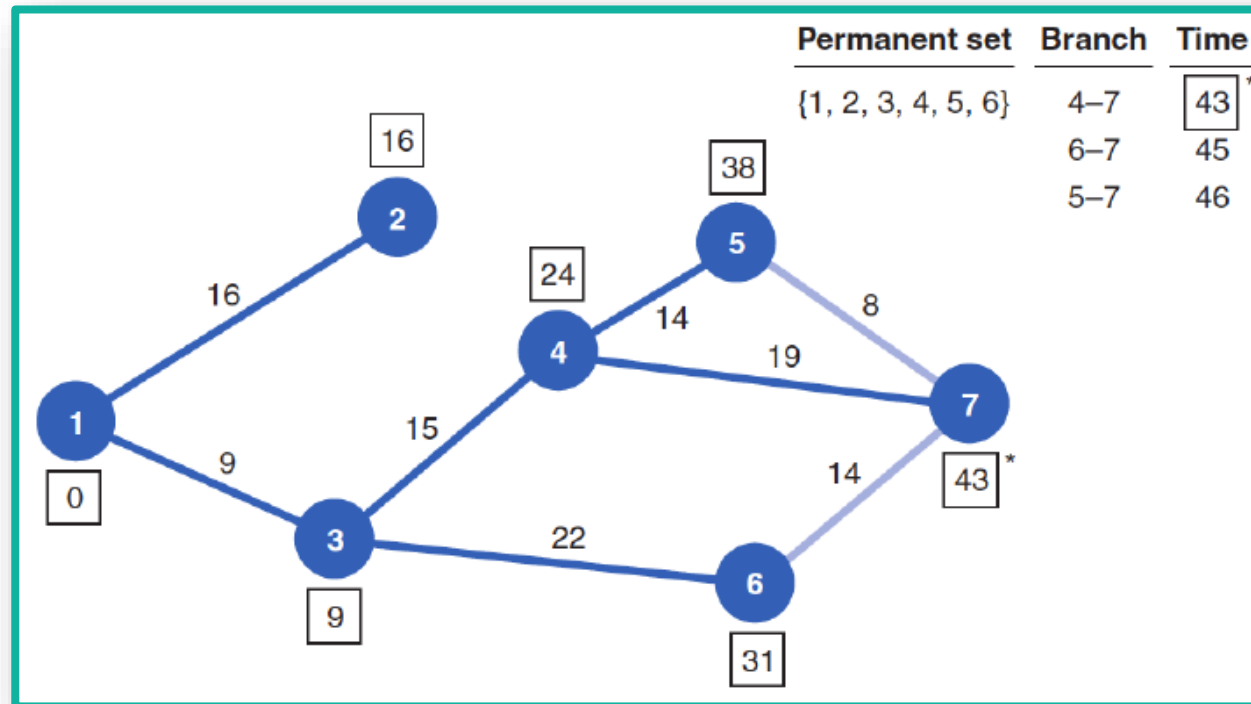


- Node 5 is the closest to the origin and added to the permanent set

Ex: Shortest Shipping Route



- Last, explore all the nodes adjacent to nodes in the permanent set {1, 2, 3, 4, 6, 5}

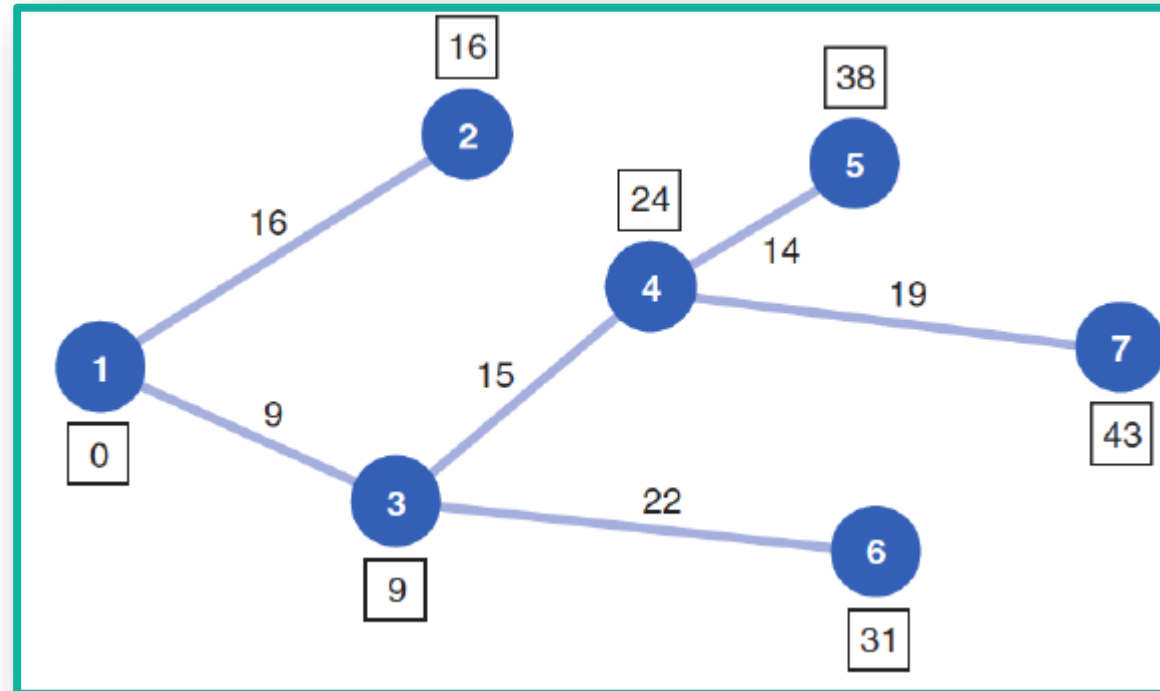


- Node 7 is the last one added
- Minimum time from origin to node 7 is 43 (19+15+9)

Ex: Shortest Shipping Route



- Final solution represented graphically using a subset of the edges



- Minimum time travelled from node 1 to all other nodes

$$9 + 16 + 24 + 31 + 38 + 43 = 161$$

Ex: Shortest Shipping Route



- Optimal routes to reach each of the 7 cities

From L.A. to:	Route	Total hours
Salt Lake City (node 2)	1-2	16
Phoenix (node 3)	1-3	9
Denver (node 4)	1-3-4	24
Des Moines (node 5)	1-3-4-5	38
Dallas (node 6)	1-3-6	31
St. Louis (node 7)	1-3-4-7	43

Routes can be expressed using
an ordered list of nodes



Shortest Route Problem



- Dijkstra's algorithm for identifying the shortest route
- Step 1: Select the node with the shortest route from the origin
- Step 2: Create a permanent set that includes the origin and the node chosen in the 1st step
- Step 3: Identify all nodes that are adjacent to the nodes in the permanent set
- Step 4: Select the node with the shortest route from the group of nodes adjacent to the nodes in the permanent set. Add the chosen node to the permanent set
- Step 5: Repeat the 3rd and 4th steps until all nodes are in the permanent set

Shortest Route Problem



- We want to reformulate the problem as a linear program
- The origin in the shortest route problem can be thought of as a single **supply** node
- The other nodes can be thought of as **demand** nodes
- Source has supply equal to the number of nodes in the graph minus one (itself)
- Each demand node requires a single unit
- Distance between nodes corresponds to transportation cost of that edge
- To reduce the number of variables, we assume units only flow in the direction of a higher node number

Ignore x_{ij} if $i \geq j$

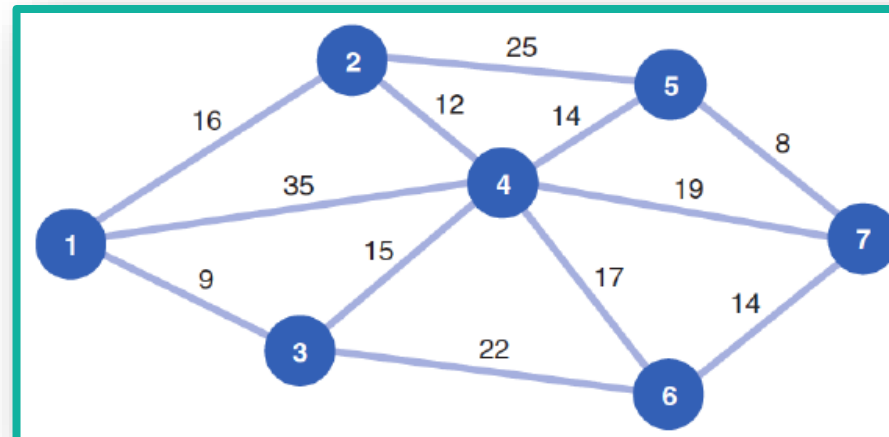
Ex: Shortest Shipping Route



- Decision variables
 - x_{ij} = number of trucks transported along edge (i,j)
 - $i = \{1,2,3,\dots,7\}$
 - $j = \{1,2,3,\dots,7\}$
 - $i < j$

- Objective function

$$Z = 16x_{12} + 35x_{14} + 9x_{13} + 12x_{24} + 15x_{34} + 25x_{25} + 14x_{45} + 17x_{46} + 8x_{57} + 19x_{47} + 22x_{36} + 14x_{67}$$



Ex: Shortest Shipping Route



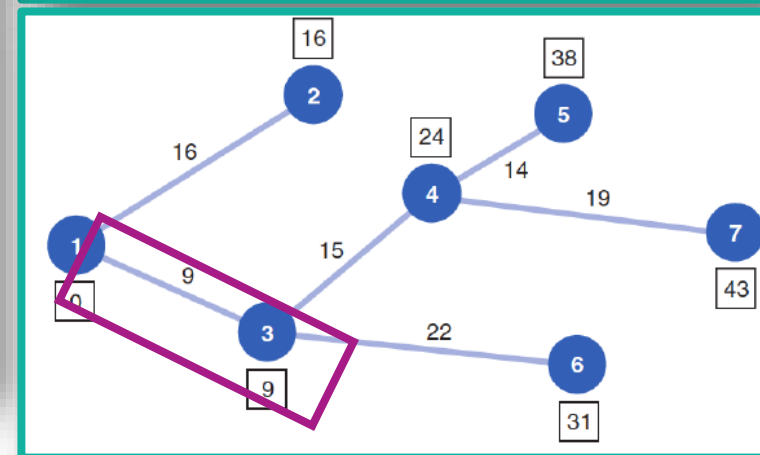
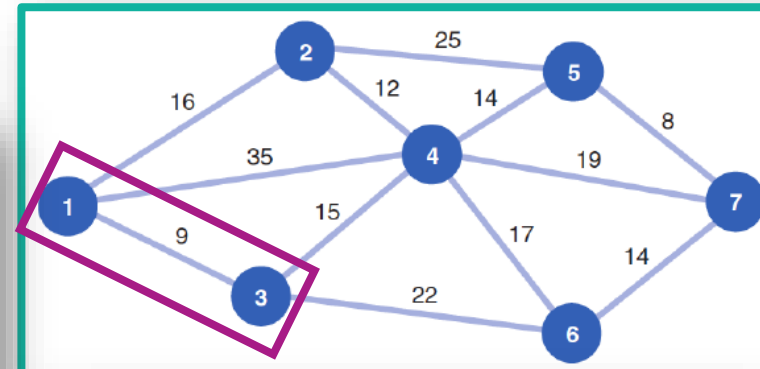
- Whatever gets into a node leaves the node is known as **flow conservation**
- Origin produces 1 unit of flow and the node with largest index must get 1 unit
- Constraints
 - $x_{12} + x_{13} + x_{14} = 6$ (Out of node 1)
 - $x_{12} = x_{24} + x_{25} + 1$ (Through node 2)
 - $x_{13} = x_{34} + x_{36} + 1$ (Through node 3)
 - $x_{14} + x_{24} + x_{34} = x_{45} + x_{46} + x_{47} + 1$ (Through node 4)
 - $x_{25} + x_{45} = x_{57} + 1$ (Through node 5)
 - $x_{36} + x_{46} = x_{67} + 1$ (Through node 6)
 - $x_{47} + x_{57} + x_{67} = 1$ (Into node 7)
- Download **ShortestRoute.xlsx** from course website from link **Sheet 2** and focus on tab named **One-to-All**

Ex: Shortest Shipping Route



- Solution to shipping problem

	A	B	C	D	E	F
2	One source to all other nodes					
3						
4						
5	Units shipped	Node	City	Node	City	Distance (hours)
6	1	1	Los Angeles	2	Salt Lake City	16
7	5	1	Los Angeles	3	Phoenix	9
8	0	1	Los Angeles	4	Denver	35
9	0	2	Salt Lake City	4	Denver	12
10	0	2	Salt Lake City	5	Des Moines	25
11	3	3	Phoenix	4	Denver	15
12	1	3	Phoenix	6	Dallas	22
13	1	4	Denver	5	Des Moines	14
14	0	4	Denver	6	Dallas	17
15	1	4	Denver	7	St. Louis	19
16	0	5	Des Moines	7	St. Louis	8
17	0	6	Dallas	7	St. Louis	14
18				Total		161





The End



Dale

