

MATH 308 D200, Fall 2019

## 23. The transportation algorithm

(based on notes from Dr. J. Hales, Dr. L. Stacho, and Dr. L. Goddyn)

Dr. Masood Masjoody

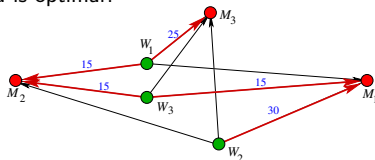
SFU Burnaby

# Transportation Algorithm

The goal is to convert a **feasible basic transportation tableau**, such as is produced by VAM, into an **optimal basic transportation tableau**. The **transportation algorithm** of the textbook mimics Phase 2 of the dual simplex algorithm.

How can we tell if a transportation tableau is optimal?

	$M_1$	$M_2$	$M_3$	
$W_1$	30	$\textcircled{20}^{15}$	$\textcircled{10}^{25}$	40
$W_2$	$\textcircled{15}^{30}$	30	25	30
$W_3$	$\textcircled{30}^{15}$	$\textcircled{20}^{15}$	15	30
	45	30	25	

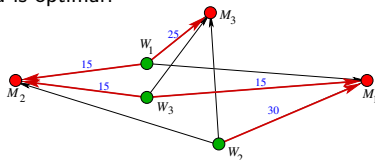


# Transportation Algorithm

The goal is to convert a **feasible basic transportation tableau**, such as is produced by VAM, into an **optimal basic transportation tableau**. The **transportation algorithm** of the textbook mimics Phase 2 of the dual simplex algorithm.

How can we tell if a transportation tableau is optimal?

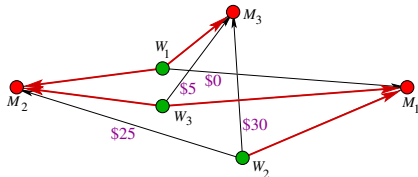
	$M_1$	$M_2$	$M_3$	
$W_1$	30	$\textcircled{20}^{15}$	$\textcircled{10}^{25}$	40
$W_2$	$\textcircled{15}^{30}$	30	25	30
$W_3$	$\textcircled{30}^{15}$	$\textcircled{20}^{15}$	15	30
	45	30	25	



The **dual slack variables**,  $s_{ij}$ , of the simplex tableau tell us that this one is optimal.

$x_{11}$	$x_{23}$	$x_{22}$	$x_{33}$	-1	
-1	1	1	1	15	$= -x_{32}$
0	1	1	0	30	$= -x_{21}$
1	-1	-1	0	15	$= -x_{31}$
1	-1	0	-1	15	$= -x_{12}$
0	1	0	1	25	$= -x_{13}$
0	-30	-25	-50	1750	$= f$

$= s_{11} = s_{23} = s_{22} = s_{33}$

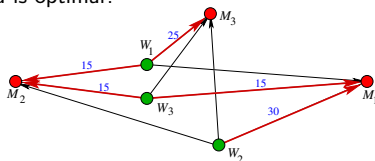


# Transportation Algorithm

The goal is to convert a **feasible basic transportation tableau**, such as is produced by VAM, into an **optimal basic transportation tableau**. The **transportation algorithm** of the textbook mimics Phase 2 of the dual simplex algorithm.

How can we tell if a transportation tableau is optimal?

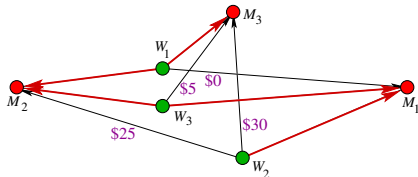
	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



The **dual slack variables**,  $s_{ij}$ , of the simplex tableau tell us that this one is optimal.

$x_{11}$	$x_{23}$	$x_{22}$	$x_{33}$	-1	
-1	1	1	1	15	$= -x_{32}$
0	1	1	0	30	$= -x_{21}$
1	-1	-1	0	15	$= -x_{31}$
1	-1	0	-1	15	$= -x_{12}$
0	1	0	1	25	$= -x_{13}$
0	-30	-25	-50	1750	$= f$

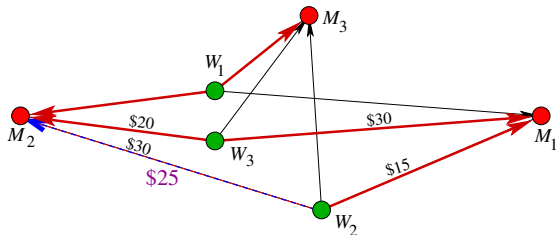
$$= s_{11} = s_{23} = s_{22} = s_{33}$$



We need to compute  $s_{ij}$  for every non-basic edge, and check whether  $s_{ij} \geq 0$ .

## Computing dual slack values

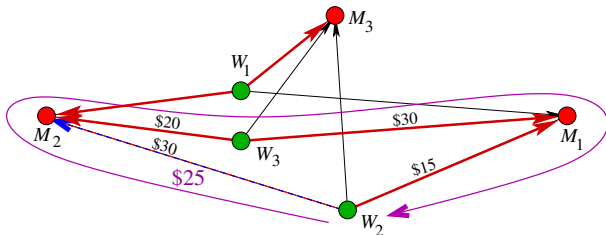
Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



To compute  $s_{22}$ :

## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



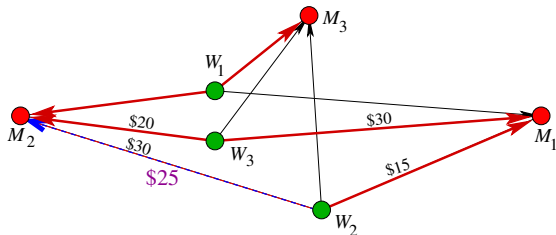
To compute  $s_{22}$ :

**Naive method:** Find the cycle using edge  $(W_2, M_2)$  and some tree edges. Then take an alternating sum of their costs.

$$s_{22} = 30 - 20 + 30 - 15 = \$25.$$

## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



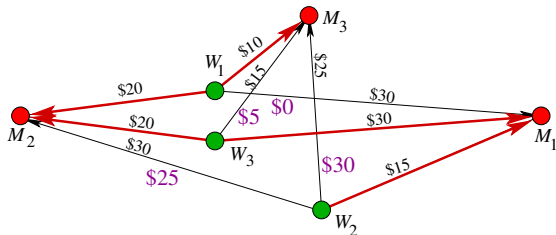
To compute  $s_{22}$ :

**Naive method:** Find the cycle using edge  $(W_2, M_2)$  and some tree edges. Then take an alternating sum of their costs.

$$s_{22} = 30 - 20 + 30 - 15 = \$25.$$

## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



To compute  $s_{22}$ :

**Naive method:** Find the cycle using edge  $(W_2, M_2)$  and some tree edges. Then take an alternating sum of their costs.

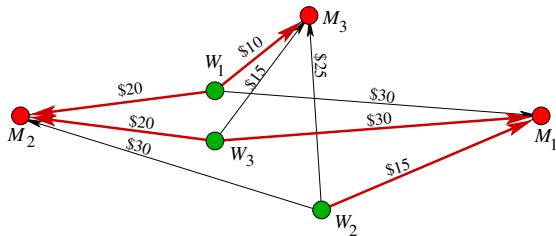
$$s_{22} = 30 - 20 + 30 - 15 = \$25.$$

Repeat this for every nonbasic edge.



## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.

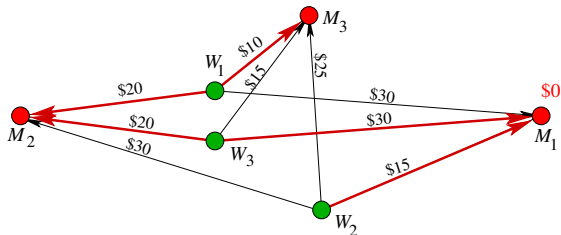


To compute  $s_{22}$ :

**Better method:** Compute *node prices*  $a_1, a_2, a_3, b_1, b_3, b_3$  for every node:

## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



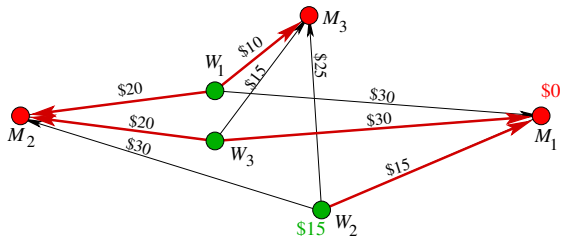
To compute  $s_{22}$ :

**Better method:** Compute *node prices*  $a_1, a_2, a_3, b_1, b_3, b_3$  for every node:

- Pick any node, and give it any price. (The textbook suggests setting  $b_1 = \$0$ , but often a higher price works better.)

## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



To compute  $s_{22}$ :

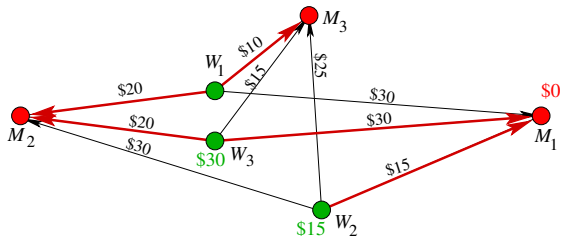
**Better method:** Compute **node prices**  $a_1, a_2, a_3, b_1, b_3, b_3$  for every node:

- Pick any node, and give it any price. (The textbook suggests setting  $b_1 = \$0$ , but often a higher price works better.)
- Find all the other node prices iteratively by using **tree edges** and the equation

$$c_{ij} = a_i + b_j, \text{ for every basic edge } (i, j).$$

## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



To compute  $s_{22}$ :

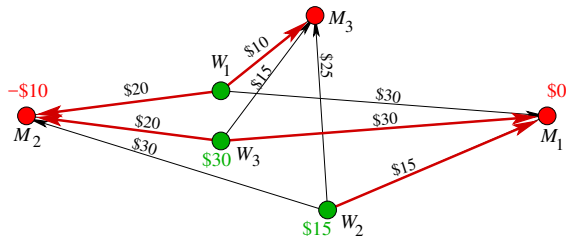
**Better method:** Compute **node prices**  $a_1, a_2, a_3, b_1, b_3, b_3$  for every node:

- Pick any node, and give it any price. (The textbook suggests setting  $b_1 = \$0$ , but often a higher price works better.)
- Find all the other node prices iteratively by using **tree edges** and the equation

$$c_{ij} = a_i + b_j, \text{ for every basic edge } (i, j).$$

## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



To compute  $s_{22}$ :

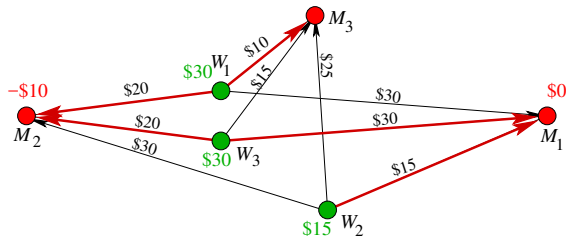
**Better method:** Compute *node prices*  $a_1, a_2, a_3, b_1, b_3, b_3$  for every node:

- Pick any node, and give it any price. (The textbook suggests setting  $b_1 = \$0$ , but often a higher price works better.)
- Find all the other node prices iteratively by using **tree edges** and the equation

$$c_{ij} = a_i + b_j, \text{ for every basic edge } (i, j).$$

## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



To compute  $s_{22}$ :

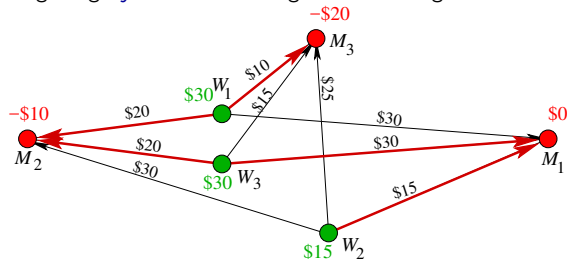
**Better method:** Compute *node prices*  $a_1, a_2, a_3, b_1, b_3, b_3$  for every node:

- Pick any node, and give it any price. (The textbook suggests setting  $b_1 = \$0$ , but often a higher price works better.)
- Find all the other node prices iteratively by using **tree edges** and the equation

$$c_{ij} = a_i + b_j, \text{ for every basic edge } (i, j).$$

## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



To compute  $s_{22}$ :

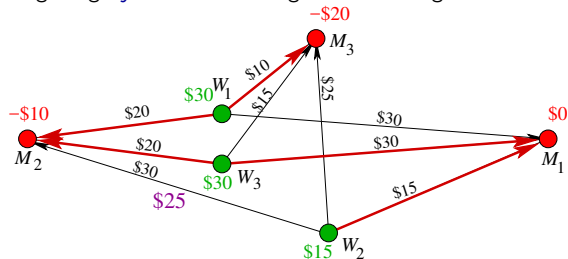
**Better method:** Compute *node prices*  $a_1, a_2, a_3, b_1, b_3, b_3$  for every node:

- Pick any node, and give it any price. (The textbook suggests setting  $b_1 = \$0$ , but often a higher price works better.)
- Find all the other node prices iteratively by using **tree edges** and the equation

$$c_{ij} = a_i + b_j, \text{ for every basic edge } (i, j).$$

## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



To compute  $s_{22}$ :

**Better method:** Compute *node prices*  $a_1, a_2, a_3, b_1, b_3, b_3$  for every node:

- Pick any node, and give it any price. (The textbook suggests setting  $b_1 = \$0$ , but often a higher price works better.)
- Find all the other node prices iteratively by using **tree edges** and the equation

$$c_{ij} = a_i + b_j, \text{ for every basic edge } (i, j).$$

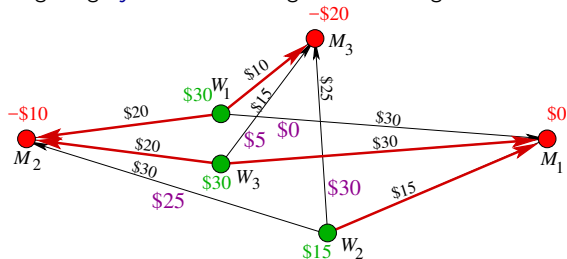
- For every non basic edge  $(i, j)$ , set

$$s_{ij} = c_{ij} - a_i - b_j.$$



## Computing dual slack values

Think of  $s_{ij}$  as a **price differential** or **reduced cost**: it is the extra cost of rerouting an item through edge  $ij$  instead of through the basic edges in the tree.



To compute  $s_{22}$ :

**Better method:** Compute *node prices*  $a_1, a_2, a_3, b_1, b_3, b_3$  for every node:

- Pick any node, and give it any price. (The textbook suggests setting  $b_1 = \$0$ , but often a higher price works better.)
- Find all the other node prices iteratively by using **tree edges** and the equation

$$c_{ij} = a_i + b_j, \text{ for every basic edge } (i, j).$$

- For every non basic edge  $(i, j)$ , set

$$s_{ij} = c_{ij} - a_i - b_j.$$

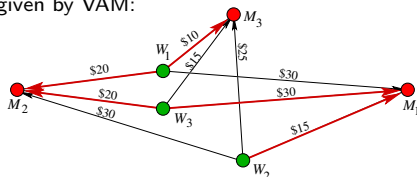
Big Advantage: Fewer calculations for each nonbasic edge.

Small Disadvantage: The reduced costs  $a_i, b_j$  must be updated after each pivot.

# Transportation Algorithm

Using the feasible transportation tableau given by VAM:

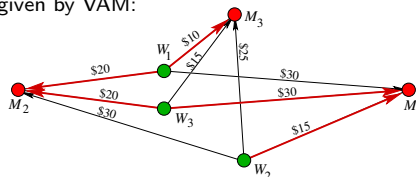
	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



# Transportation Algorithm

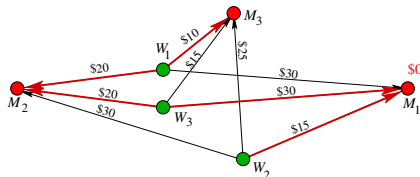
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



Find node prices, using  $c_{ij} = a_i + b_j$ :

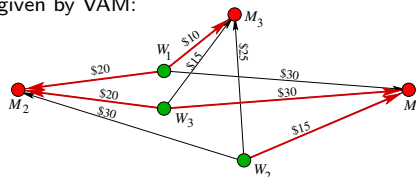
	0			
30	(20)	(10)		
(15)	30	25		
(30)	(20)	15		



# Transportation Algorithm

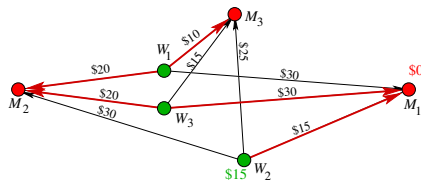
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



Find node prices, using  $c_{ij} = a_i + b_j$ :

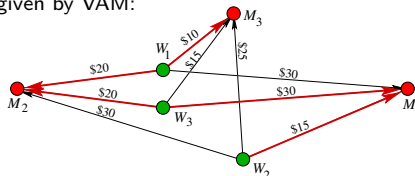
	0			
15	30	(20)	(10)	
	(15)	30	25	
	(30)	(20)	15	



# Transportation Algorithm

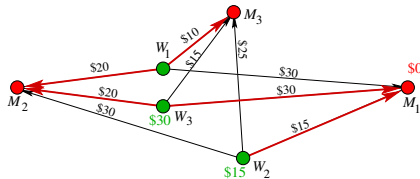
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



Find node prices, using  $c_{ij} = a_i + b_j$ :

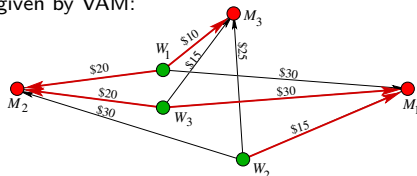
	0			
15	30	(20)	(10)	
30	(15)	30	25	
	(30)	(20)	15	



# Transportation Algorithm

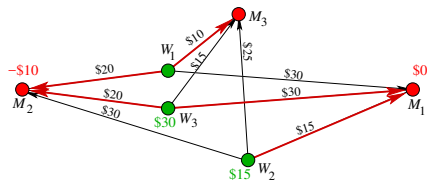
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



Find node prices, using  $c_{ij} = a_i + b_j$ :

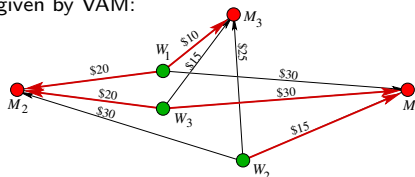
	0	-10	
15	30	(20)	(10)
30	(15)	30	25
	(30)	(20)	15



# Transportation Algorithm

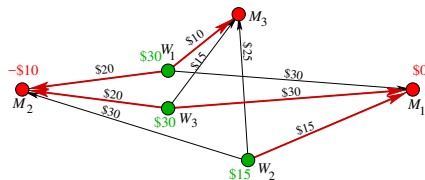
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



Find node prices, using  $c_{ij} = a_i + b_j$ :

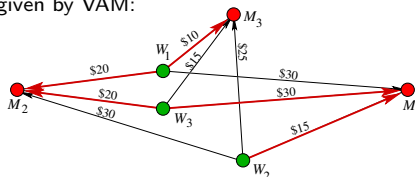
	0	-10	
30	30	(20)	(10)
15	(15)	30	25
30	(30)	(20)	15



# Transportation Algorithm

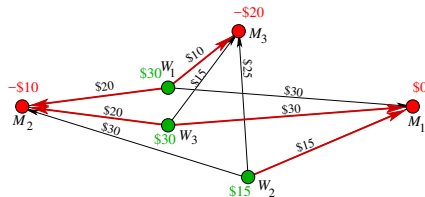
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



Find node prices, using  $c_{ij} = a_i + b_j$ :

	0	-10	-20
30	30	(20)	(10)
15	(15)	30	25
30	(30)	(20)	15

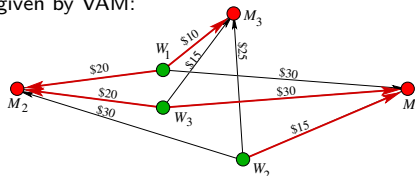




# Transportation Algorithm

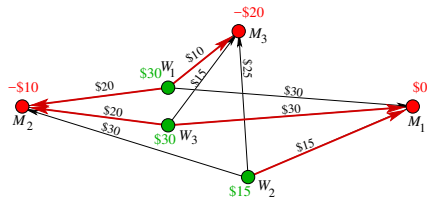
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



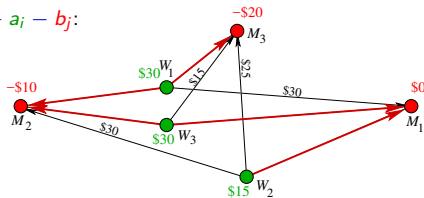
Find node prices, using  $c_{ij} = a_i + b_j$ :

	0	-10	-20
30	30	(20)	(10)
15	(15)	30	25
30	(30)	(20)	15



Find reduced prices, using  $s_{ij} = c_{ij} - a_i - b_j$ :

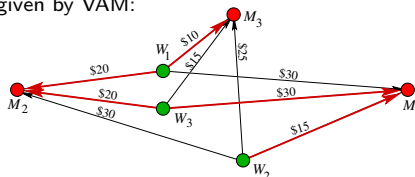
	0	-10	-20
30	30	(0)	(0)
15	(0)	30	25
30	(0)	(0)	15



# Transportation Algorithm

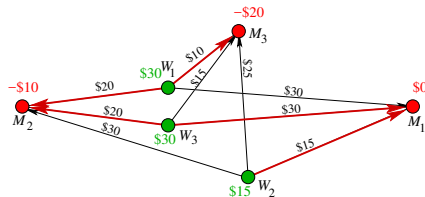
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



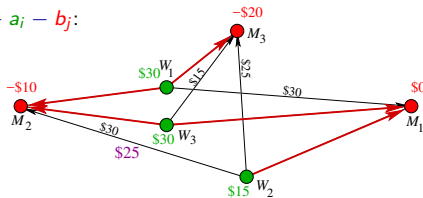
Find node prices, using  $c_{ij} = a_i + b_j$ :

	0	-10	-20
30	30	(20)	(10)
15	(15)	30	25
30	(30)	(20)	15



Find reduced prices, using  $s_{ij} = c_{ij} - a_i - b_j$ :

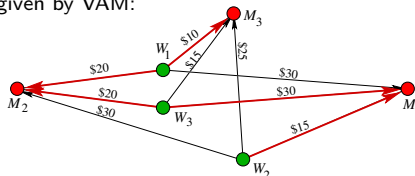
	0	-10	-20
30	30	(0)	(0)
15	(0)	25	25
30	(0)	(0)	15



# Transportation Algorithm

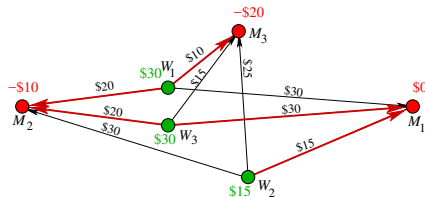
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



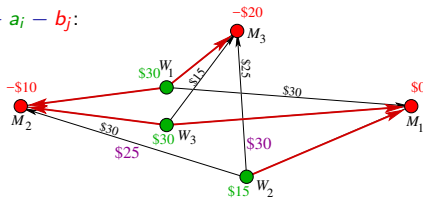
Find node prices, using  $c_{ij} = a_i + b_j$ :

	0	-10	-20
30	30	(20)	(10)
15	(15)	30	25
30	(30)	(20)	15



Find reduced prices, using  $s_{ij} = c_{ij} - a_i - b_j$ :

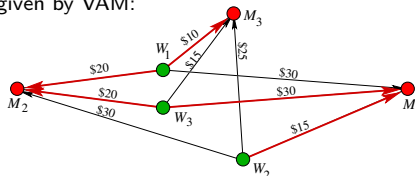
	0	-10	-20
30	30	(0)	(0)
15	(0)	25	30
30	(0)	(0)	15



# Transportation Algorithm

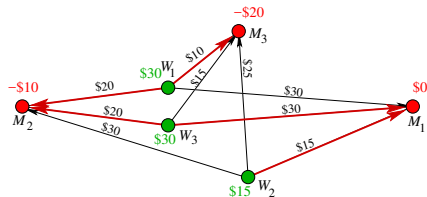
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



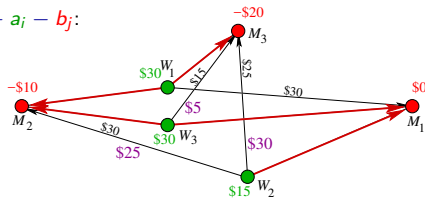
Find node prices, using  $c_{ij} = a_i + b_j$ :

	0	-10	-20
30	30	(20)	(10)
15	(15)	30	25
30	(30)	(20)	15



Find reduced prices, using  $s_{ij} = c_{ij} - a_i - b_j$ :

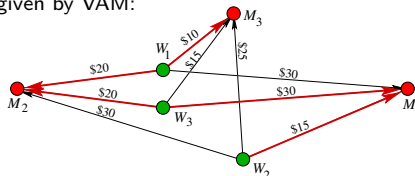
	0	-10	-20
30	30	(0)	(0)
15	(0)	25	30
30	(0)	(0)	5



# Transportation Algorithm

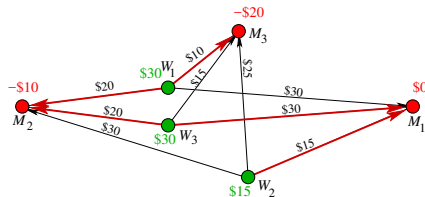
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



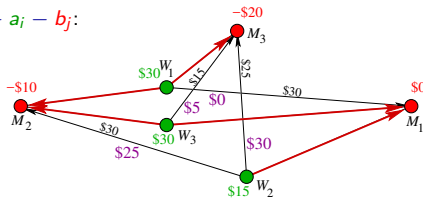
Find node prices, using  $c_{ij} = a_i + b_j$ :

	0	-10	-20
30	30	(20)	(10)
15	(15)	30	25
30	(30)	(20)	15



Find reduced prices, using  $s_{ij} = c_{ij} - a_i - b_j$ :

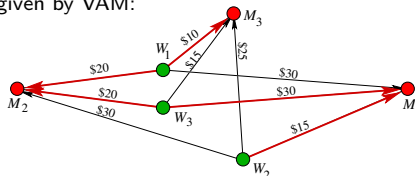
	0	-10	-20
30	0	(0)	(0)
15	(0)	25	30
30	(0)	(0)	5



# Transportation Algorithm

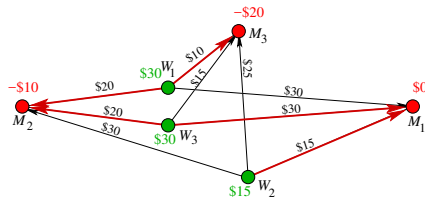
Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



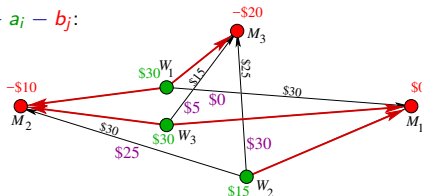
Find node prices, using  $c_{ij} = a_i + b_j$ :

	0	-10	-20
30	30	(20)	(10)
15	(15)	30	25
30	(30)	(20)	15



Find reduced prices, using  $s_{ij} = c_{ij} - a_i - b_j$ :

	0	-10	-20
30	0	(0)	(0)
15	(0)	25	30
30	(0)	(0)	5

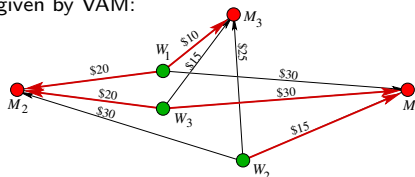


This is **optimal**, so **STOP**.

# Transportation Algorithm

Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	



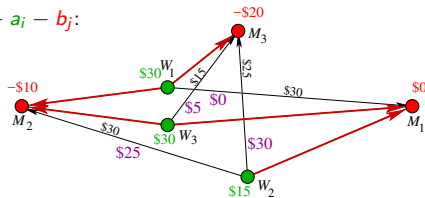
The initial node and price can be chosen, since  $c_{ij} - (a_i + t) - (b_j + t) = c_{ij} - a_i - b_j$ .

15

30	(20)	(10)
(15)	30	25
(30)	(20)	15

Find reduced prices, using  $s_{ij} = c_{ij} - a_i - b_j$ :

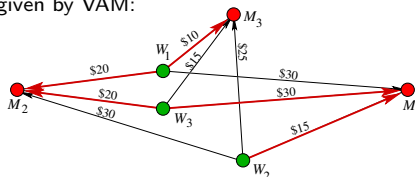
	0	-10	-20
30	0	0	0
15	0	25	30
30	0	0	5



# Transportation Algorithm

Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	

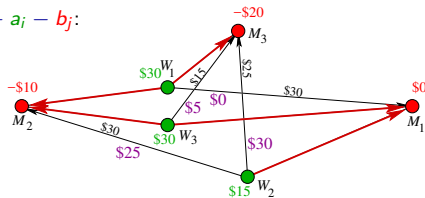


The initial node and price can be chosen, since  $c_{ij} - (a_i + t) - (b_j + t) = c_{ij} - a_i - b_j$ .

	15	5	-5
15	30	(20)	(10)
0	(15)	30	25
15	(30)	(20)	15

Find reduced prices, using  $s_{ij} = c_{ij} - a_i - b_j$ :

	0	-10	-20
30	0	(0)	(0)
15	(0)	25	30
30	(0)	(0)	5

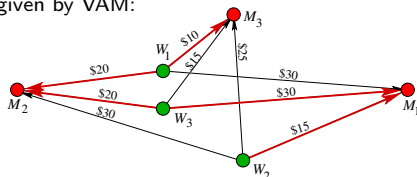




# Transportation Algorithm

Using the feasible transportation tableau given by VAM:

	$M_1$	$M_2$	$M_3$	
$W_1$	30	(20) <sup>15</sup>	(10) <sup>25</sup>	40
$W_2$	(15) <sup>30</sup>	30	25	30
$W_3$	(30) <sup>15</sup>	(20) <sup>15</sup>	15	30
	45	30	25	

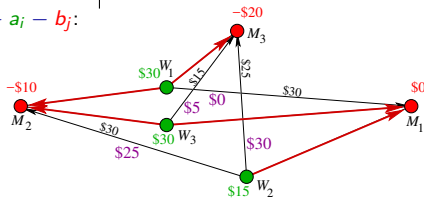


The initial node and price can be chosen, since  $c_{ij} - (a_i + t) - (b_j + t) = c_{ij} - a_i - b_j$ .

	$0 + 15$	$-10 + 15$	$-20 + 15$
30 - 15	30	(20)	(10)
15 - 15	(15)	30	25
30 - 15	(30)	(20)	15

Find reduced prices, using  $s_{ij} = c_{ij} - a_i - b_j$ :

	0	-10	-20
30	0	(0)	(0)
15	(0)	25	30
30	(0)	(0)	5



# Transportation Algorithm

Recall  $a_i$ ,  $b_j$  are just dual variables for the original LP:

$$(P) \min \quad C = \sum_{i=1}^m \sum_{j=1}^n x_{ij} c_{ij}$$

$$\text{s. t.} \quad \sum_{j=1}^n x_{ij} = s_i, \quad (i = 1, 2, \dots, m)$$

$$\sum_{i=1}^m x_{ij} = d_j, \quad (j = 1, 2, \dots, n)$$

$$x_{ij} \geq 0, \quad \text{for all } i, j$$

# Transportation Algorithm

Recall  $a_i$ ,  $b_j$  are just dual variables for the original LP:

$$(P) \min \quad C = \sum_{i=1}^m \sum_{j=1}^n x_{ij} c_{ij}$$

$$\text{s. t.} \quad \sum_{j=1}^n x_{ij} = s_i, \quad (i = 1, 2, \dots, m) \quad (a_i)$$

$$\sum_{i=1}^m x_{ij} = d_j, \quad (j = 1, 2, \dots, n) \quad (b_j)$$

$$x_{ij} \geq 0, \quad \text{for all } i, j$$

# Transportation Algorithm

Recall  $a_i$ ,  $b_j$  are just dual variables for the original LP:

$$(P) \min \quad C = \sum_{i=1}^m \sum_{j=1}^n x_{ij} c_{ij}$$

$$\text{s. t.} \quad \sum_{j=1}^n x_{ij} = s_i, \quad (i = 1, 2, \dots, m) \quad (a_i)$$

$$\sum_{i=1}^m x_{ij} = d_j, \quad (j = 1, 2, \dots, n) \quad (b_j)$$

$$x_{ij} \geq 0, \quad \text{for all } i, j$$

$$(D) \max \quad P = \sum_{i=1}^m s_i a_i + \sum_{j=1}^n d_j b_j$$

$$\text{s. t.} \quad a_i + b_j \leq c_{ij} \quad \text{for all } i, j \quad (x_{ij})$$
$$a_i, b_j \text{ unrestricted}$$

# The Transportation Algorithm

## Definition (Cycle in a Tableau)

A cycle  $C$  in a balanced transportation tableau  $T$  is a collection of cells of  $T$  such that each row and each column of  $T$  contains exactly zero or two cells of  $C$ .

Only horizontal and vertical movement is allowed to connect the cells...

# The Transportation Algorithm

## The Transportation Algorithm TA

0. Given an initial balanced transportation tableau.
1. Apply VAM to obtain a basic feasible solution and a corresponding basis.
2. Let  $b_1 = 0$  (or any number). Determine  $a_1, a_2, \dots, a_m$  and  $b_2, b_3, \dots, b_n$  uniquely such that  $a_i + b_j = c_{ij}$  for all **basis** cells.
3. For each  $i, j$ , replace cell costs  $c_{ij}$  by reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ ; the textbook calls  $s_{ij}$  the “new cell costs  $c_{ij}$ ”.)
4. If  $s_{ij} \geq 0$  for all  $i, j$ , then replace all cells with their original costs  $c_{ij}$ ; the current basic feasible solution is optimal. Otherwise, continue.
5. Choose  $s_{ij} < 0$ . To break ties use Bland’s anti-cycling rule: Choose  $s_{ij} < 0$  with smallest  $i$  and with respect to this with the smallest  $j$  (the Northwest-most negative cell). Label the “getter” cell with  $(\square^+)$ . Find the unique cycle  $C$  determined by the getter cell and some of the basis cells. Label cells of  $C$  starting from  $\square$  alternately “getter”  $(+)$  and “giver”  $(-)$ . Choose the “giver”  $(-)$  cell associated with the smallest flow of goods; break ties arbitrarily.
6. Adjust the flows  $x_{ij}$ : Add the squared cell of **Step 5**. to the basis, i.e., circle it in a new tableau. Remove the chosen “giver” from the basis, i.e., do not circle it in a new tableau. Add the amount of goods of this “giver” to amount of goods of all “getters” in  $C$  and subtract from the amount of goods of all “givers” in  $C$ . Go to **Step 2**.

Apply TA to the following (balanced) transportation problem

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

# Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

 $\xrightarrow{\text{VAM}}$ 

② <sup>20</sup>	1	② <sup>20</sup>	40
⑨ <sup>10</sup>	④ <sup>50</sup>	7	60
① <sup>10</sup>	2	9	10
40	50	20	110



## Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ .

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

 $\xrightarrow{\text{VAM}}$ 

② <sup>20</sup>	1	② <sup>20</sup>	40
⑨ <sup>10</sup>	④ <sup>50</sup>	7	60
① <sup>10</sup>	2	9	10
40	50	20	110

## Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ .

2	1	2	40		
9	4	7	60		
1	2	9	10		
40	50	20	110		

2

 $\xrightarrow{\text{VAM}}$ 
0

② <sup>20</sup>	1	② <sup>20</sup>	40		
⑨ <sup>10</sup>	④ <sup>50</sup>	7	60		
① <sup>10</sup>	2	9	10		
40	50	20	110		

Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ .

0

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

VAM

② <sup>20</sup>	1	② <sup>20</sup>	40
⑨ <sup>10</sup>	④ <sup>50</sup>	7	60
① <sup>10</sup>	2	9	10
40	50	20	110

Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ .

Diagram illustrating the VAM (Vogel's Approximation Method) step. The initial cost matrix is shown on the left, and the updated matrix is shown on the right.

**Initial Cost Matrix:**

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

**Updated Cost Matrix (after VAM step):**

2 <sup>20</sup>	1	2 <sup>20</sup>	40
9 <sup>10</sup>	4 <sup>50</sup>	7	60
1 <sup>10</sup>	2	9	10
40	50	20	110

The VAM step is indicated by a green arrow pointing from the initial matrix to the updated matrix. The updated matrix shows the selection of the minimum cost cell (1,1) with a value of 2, and the next minimum cost cell (1,3) with a value of 2. The total cost is updated to 100.

Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ .

Diagram illustrating the transformation of matrix  $A$  into matrix  $U$  using VAM (Row Summation Method).

Matrix  $A$  (Initial Matrix):

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

Matrix  $U$  (Upper Triangular Matrix):

$2^{20}$	1	$2^{20}$	40
$9^{10}$	$4^{50}$	7	60
$1^{10}$	2	9	10
40	50	20	110

The transformation is indicated by the arrow labeled VAM.

Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ .

Diagram illustrating the transformation of matrix  $A$  into matrix  $U$  using VAM (Row Summation Method).

Matrix  $A$  (Initial Matrix):

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

Matrix  $U$  (Upper Triangular Matrix):

0	-5	0	
$2^{20}$	1	$2^{20}$	40
$9^{10}$	$4^{50}$	7	60
$1^{10}$	2	9	10
40	50	20	110

The transformation is indicated by the arrow labeled VAM.

Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .

The diagram illustrates the VAM (Vogel's Approximation Method) process for finding an initial feasible solution to a transportation problem. It shows three stages of the process:

- Initial Cost Matrix:** A 3x3 matrix with costs: (1,1)=2, (1,2)=1, (1,3)=2, (2,1)=9, (2,2)=4, (2,3)=7, (3,1)=1, (3,2)=2, (3,3)=9. Row penalties are 0, -5, 0. Column penalties are 0, -5, 0.
- Allocation:** The lowest cost cell (1,1) is selected. 20 units are allocated to it. The matrix is updated, and the row and column are crossed out. The new row penalties are 2, 9, 1. The new column penalties are 0, -5, 0.
- Reduced Costs:** The final allocation is shown, and the reduced cost matrix is calculated. The reduced costs are: (1,2)=20, (1,3)=20, (2,1)=10, (2,2)=50, (2,3)=7, (3,1)=10, (3,2)=2, (3,3)=9. The total cost is 110.

Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .



Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .

Initial Cost Matrix:
 

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

Opportunity Costs (VAM):
 

0	-5	0
2 <sup>20</sup>	1	2 <sup>20</sup>
9 <sup>10</sup>	4 <sup>50</sup>	7
1 <sup>10</sup>	2	9
40	50	20

Reduced Cost Matrix:
 

0 <sup>20</sup>	4	0 <sup>20</sup>
0 <sup>10</sup>	0 <sup>50</sup>	
0 <sup>10</sup>		

 Reduced costs  $s_{ij}$

Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .

Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .

The diagram illustrates the steps of the VAM algorithm:

- Initial cost matrix:

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

- Applying VAM (selecting the cell with the largest difference between the highest and second-highest costs in a row or column):

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

- Resulting reduced cost matrix (with the selected cell circled and its cost reduced to zero):

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .

Diagram illustrating the reduction of a cost matrix using VAM (Vertical Allocation Method).

The initial cost matrix is shown on the left, with row and column totals. The VAM step is indicated by an arrow.

The reduced cost matrix is shown on the right, with the reduced costs  $s_{ij}$  highlighted in pink.

Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\boxed{+}$ .

The diagram illustrates the VAM (Vogel's Approximation Method) process for finding an initial feasible solution to a transportation problem. It shows three stages of the process:

**Stage 1: Initial Cost Matrix and Penalties**

2	1	2	40	2
9	4	7	60	9
1	2	9	10	1
40	50	20	110	

Row penalties: 0, -5, 0

**Stage 2: Matrix after Allocation**

Allocations: 20 units to (1,1), 10 units to (1,2), 50 units to (2,2).

2 <sup>20</sup>	1	2 <sup>20</sup>	40	2
9 <sup>10</sup>	4 <sup>50</sup>	7	60	9
1 <sup>10</sup>	2	9	10	1
40	50	20	110	

**Stage 3: Final Reduced Cost Matrix**

Reduced costs  $s_{ij}$ :

0 <sup>20</sup>	4	0 <sup>20</sup>
0 <sup>10</sup>	0 <sup>50</sup>	-2
0 <sup>10</sup>	6	8

Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\boxed{+}$ .

The diagram illustrates the VAM (Vogel's Approximation Method) process for finding an initial feasible solution. It shows three stages of the process:

- Initial Cost Matrix:** A 3x3 matrix with costs:
 

2	1	2	40
9	4	7	60
1	2	9	10
40	50	20	110

 Above the matrix are the row penalties: 0, -5, 0. To the right are the column penalties: 2, 9, 1.
- Matrix after one iteration:** The matrix after allocating 20 units to cell (1,1) and 10 units to cell (2,1). The updated costs are:
 

2 <sup>20</sup>	1	2 <sup>20</sup>	40
9 <sup>10</sup>	4 <sup>50</sup>	7	60
1 <sup>10</sup>	2	9	10
40	50	20	110

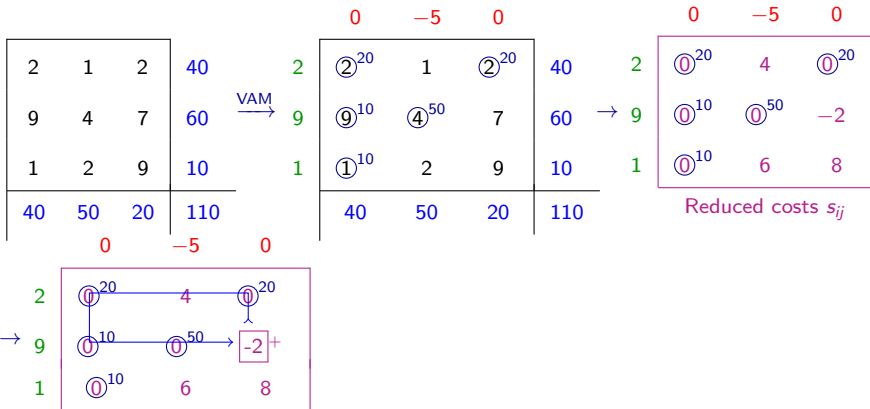
 The row penalties are 0, -5, 0. The column penalties are 2, 9, 1.
- Reduced Cost Matrix:** The matrix after subtracting the row and column penalties from the original costs. The reduced costs are:
 

0 <sup>20</sup>	4	0 <sup>20</sup>
0 <sup>10</sup>	0 <sup>50</sup>	-2
0 <sup>10</sup>	6	8

 The reduced costs are labeled as  $s_{ij}$ .

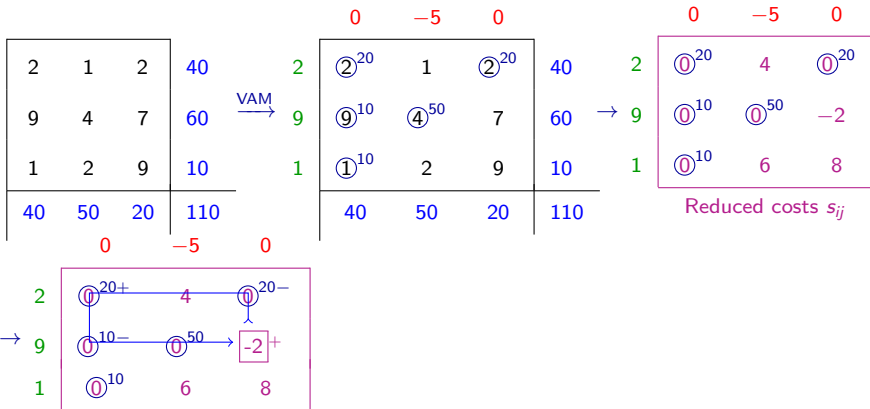
## Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\square^+$ .
4. Find the cycle of basic cells containing the getter cell.



## Apply TA to the following (balanced) transportation problem

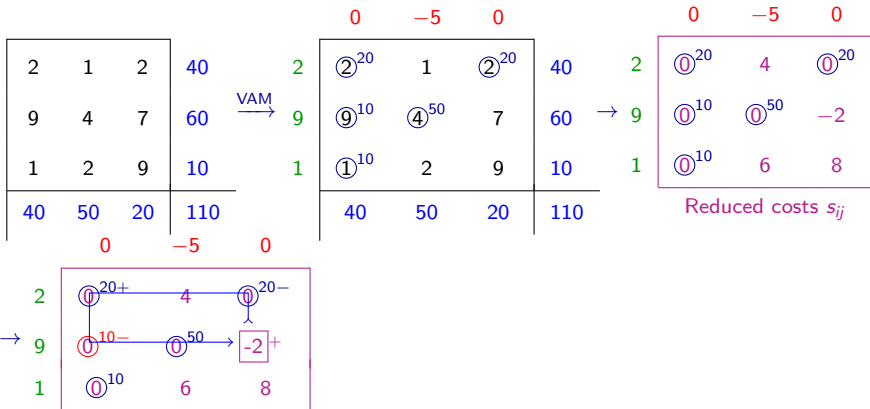
1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\square^+$ .
4. Find the cycle of basic cells containing the getter cell.
5. Mark cells in the cycle alternating as givers  $(-)$  and getters  $(+)$ .





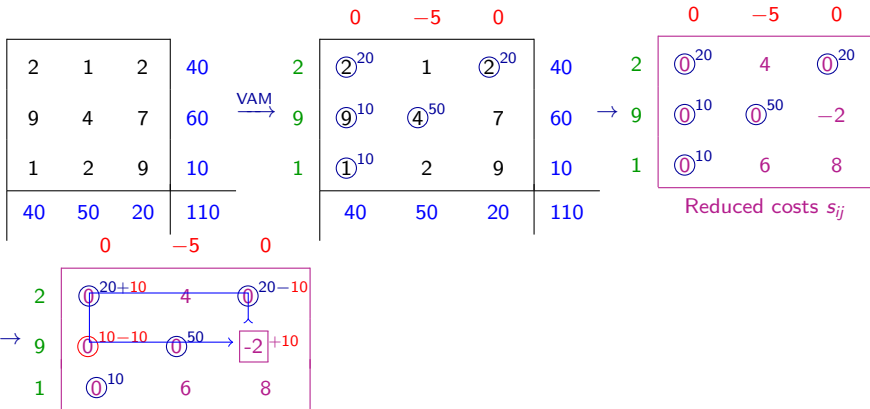
## Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\square^+$ .
4. Find the cycle of basic cells containing the getter cell.
5. Mark cells in the cycle alternating as givers ( $-$ ) and getters ( $+$ ).
6. The giver with the smallest flow will leave the basis.



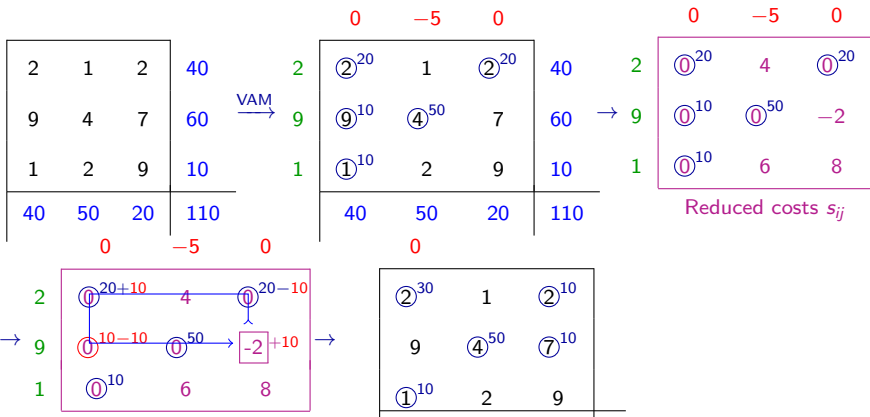
## Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\square^+$ .
4. Find the cycle of basic cells containing the getter cell.
5. Mark cells in the cycle alternating as givers ( $-$ ) and getters ( $+$ ).
6. The giver with the smallest flow will leave the basis. Adjust the flows.



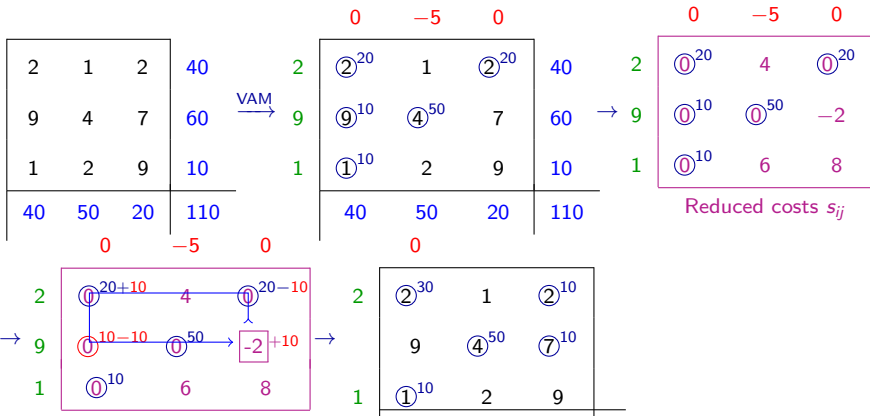
## Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\square^+$ .
4. Find the cycle of basic cells containing the getter cell.
5. Mark cells in the cycle alternating as givers  $(-)$  and getters  $(+)$ .
6. The giver with the smallest flow will leave the basis. Adjust the flows.
7. Update node values and reduced costs. Some of them will change by  $\pm 2$ .



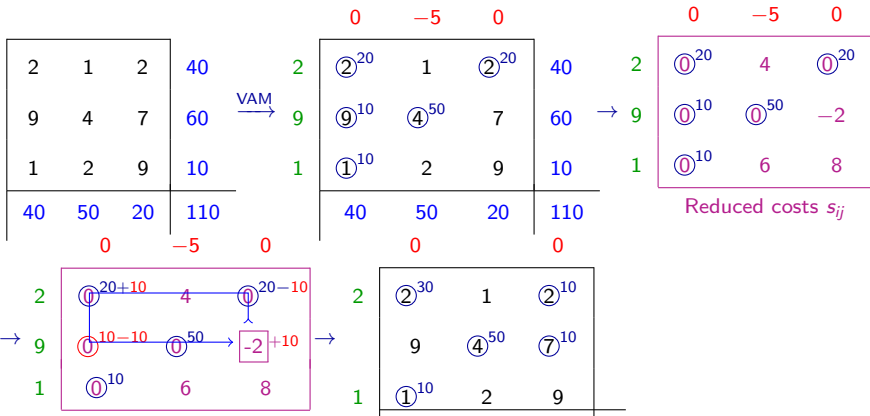
## Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\square^+$ .
4. Find the cycle of basic cells containing the getter cell.
5. Mark cells in the cycle alternating as givers ( $-$ ) and getters ( $+$ ).
6. The giver with the smallest flow will leave the basis. Adjust the flows.
7. Update node values and reduced costs. Some of them will change by  $\pm 2$ .



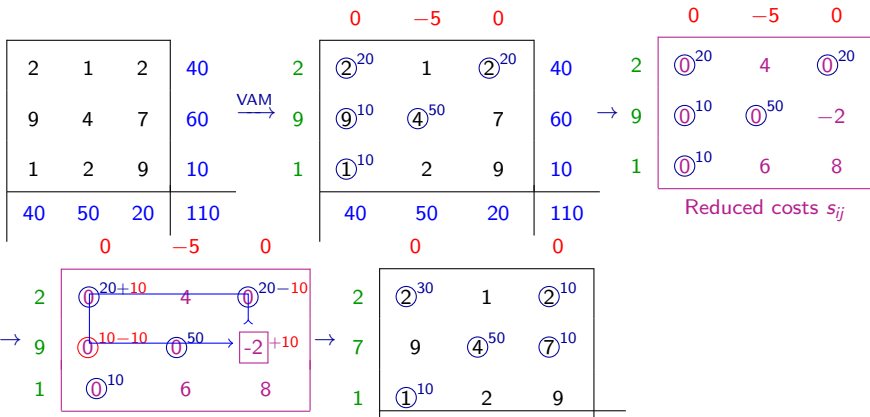
## Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\square^+$ .
4. Find the cycle of basic cells containing the getter cell.
5. Mark cells in the cycle alternating as givers ( $-$ ) and getters ( $+$ ).
6. The giver with the smallest flow will leave the basis. Adjust the flows.
7. Update node values and reduced costs. Some of them will change by  $\pm 2$ .



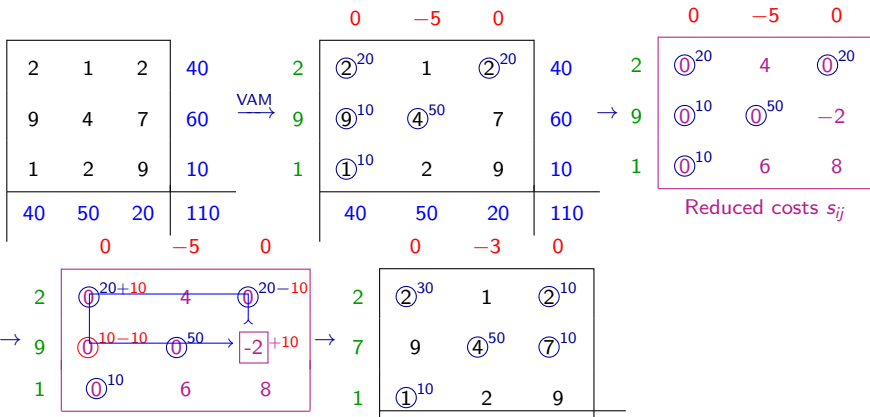
## Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\square^+$ .
4. Find the cycle of basic cells containing the getter cell.
5. Mark cells in the cycle alternating as givers  $(-)$  and getters  $(+)$ .
6. The giver with the smallest flow will leave the basis. Adjust the flows.
7. Update node values and reduced costs. Some of them will change by  $\pm 2$ .



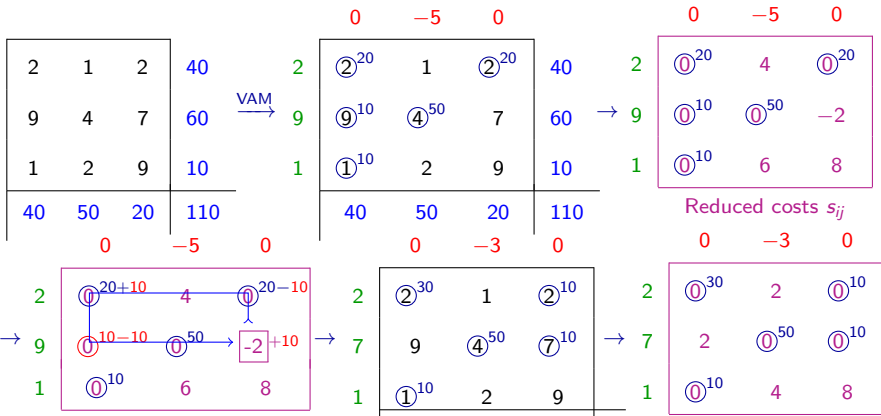
## Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\square^+$ .
4. Find the cycle of basic cells containing the getter cell.
5. Mark cells in the cycle alternating as givers  $(-)$  and getters  $(+)$ .
6. The giver with the smallest flow will leave the basis. Adjust the flows.
7. Update node values and reduced costs. Some of them will change by  $\pm 2$ .



## Apply TA to the following (balanced) transportation problem

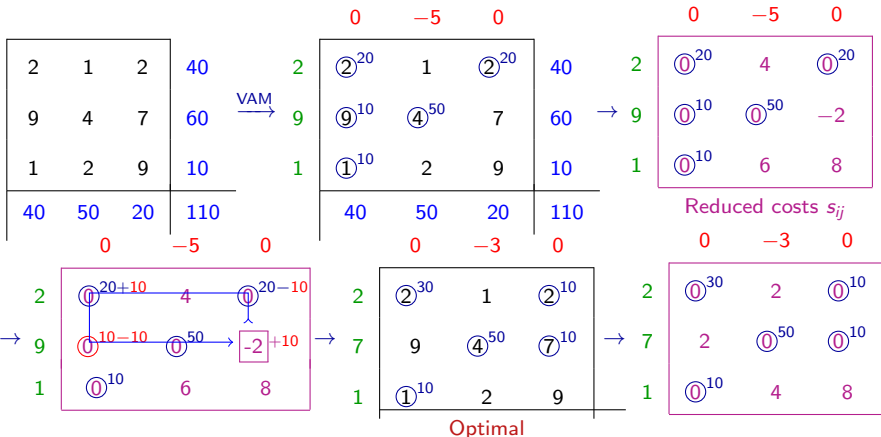
1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\square^+$ .
4. Find the cycle of basic cells containing the getter cell.
5. Mark cells in the cycle alternating as givers  $(-)$  and getters  $(+)$ .
6. The giver with the smallest flow will leave the basis. Adjust the flows.
7. Update node values and reduced costs. Some of them will change by  $\pm 2$ .





## Apply TA to the following (balanced) transportation problem

1. Find a BFS by applying VAM. Costs are  $c_{ij}$ . Supply/Demands are blue.
2. Compute node values, start with, say,  $b_1 = 0$ . Compute reduced costs  $s_{ij} = c_{ij} - a_i - b_j$ .
3. Not yet optimal. Select a "getter" cell  $s_{ij} < 0$  to enter basis, and mark it with  $\square^+$ .
4. Find the cycle of basic cells containing the getter cell.
5. Mark cells in the cycle alternating as givers  $(-)$  and getters  $(+)$ .
6. The giver with the smallest flow will leave the basis. Adjust the flows.
7. Update node values and reduced costs. Some of them will change by  $\pm 2$ .
8. All no negative reduced costs. Flow is optimal.



Solve the following BTP

5	12	8	50	26
11	4	10	8	20
14	50	1	9	30
15	20	26	15	

$5^{15}$	$12^0$	8	$50^{11}$	26
11	$4^{20}$	10	8	20
14	50	$1^{26}$	$9^4$	30
15	20	26	15	

5	12	8	50	26
11	4	10	8	20
14	50	1	9	30
15	20	26	15	

# Variations of the Transportation Problem

**Negative costs**

**Maximization**

**Forbidden routes**

**Unbalanced Transportation**

# Variations of the Transportation Problem

## Negative costs

Add a big constant  $M$ , such as  $M = -\min_{i,j} c_{ij}$ , to *all* the edge costs. Now solve the new problem. Notice the new objective function  $C'$  is just a shift of the old one  $C$ , so both problems have the same optimal solutions  $(x_{ij})$ .

$$C' = \sum_{i=1}^m \sum_{j=1}^n (c_{ij} + M)x_{ij} = \sum_{i=1}^m \sum_{j=1}^n x_{ij}c_{ij} + M \sum_{i=1}^m \sum_{j=1}^n x_{ij} = C + M \sum_{1 \leq i \leq m} s_i.$$

## Maximization

## Forbidden routes

## Unbalanced Transportation

# Variations of the Transportation Problem

## Negative costs

Add a big constant  $M$ , such as  $M = -\min_{i,j} -c_{ij}$ , to *all* the edge costs. Now solve the new problem. Notice the new objective function  $C'$  is just a shift of the old one  $C$ , so both problems have the same optimal solutions  $(x_{ij})$ .

$$C' = \sum_{i=1}^m \sum_{j=1}^n (c_{ij} + M)x_{ij} = \sum_{i=1}^m \sum_{j=1}^n x_{ij}c_{ij} + M \sum_{i=1}^m \sum_{j=1}^n x_{ij} = C + M \sum_{1 \leq i \leq m} s_i.$$

## Maximization

Convert to a minimization problem by replacing each cost  $c'_{ij} = -c_{ij}$ .

## Forbidden routes

## Unbalanced Transportation

# Variations of the Transportation Problem

## Negative costs

Add a big constant  $M$ , such as  $M = -\min_{i,j} -c_{ij}$ , to *all* the edge costs. Now solve the new problem. Notice the new objective function  $C'$  is just a shift of the old one  $C$ , so both problems have the same optimal solutions  $(x_{ij})$ .

$$C' = \sum_{i=1}^m \sum_{j=1}^n (c_{ij} + M)x_{ij} = \sum_{i=1}^m \sum_{j=1}^n x_{ij}c_{ij} + M \sum_{i=1}^m \sum_{j=1}^n x_{ij} = C + M \sum_{1 \leq i \leq m} s_i.$$

## Maximization

Convert to a minimization problem by replacing each cost  $c'_{ij} = -c_{ij}$ .

## Forbidden routes

If the route from  $W_i$  to  $M_j$  is forbidden, then put a prohibitively high cost  $c_{ij} = \infty$ .

## Unbalanced Transportation



# Variations of the Transportation Problem

## Negative costs

Add a big constant  $M$ , such as  $M = -\min_{i,j} c_{ij}$ , to all the edge costs. Now solve the new problem. Notice the new objective function  $C'$  is just a shift of the old one  $C$ , so both problems have the same optimal solutions  $(x_{ij})$ .

$$C' = \sum_{i=1}^m \sum_{j=1}^n (c_{ij} + M)x_{ij} = \sum_{i=1}^m \sum_{j=1}^n x_{ij}c_{ij} + M \sum_{i=1}^m \sum_{j=1}^n x_{ij} = C + M \sum_{1 \leq i \leq m} s_i.$$

## Maximization

Convert to a minimization problem by replacing each cost  $c'_{ij} = -c_{ij}$ .

## Forbidden routes

If the route from  $W_i$  to  $M_j$  is forbidden, then put a prohibitively high cost  $c_{ij} = \infty$ .

## Unbalanced Transportation

If total supply exceeds demand

$$\sum_{1 \leq i \leq m} s_i > \sum_{1 \leq j \leq n} d_j,$$

then add a “dummy” market  $M_{n+1}$  with edge costs

$$c_{i,m+1} = 0 \quad \text{for } i = 1, 2, \dots, m$$

and demand

$$d_{m+1} = \sum_{1 \leq i \leq m} s_i = \sum_{1 \leq j \leq n} d_j.$$

# Unbalanced Transportation Problems

	$M_1$	$M_2$	$\dots$	$M_n$	
$W_1$	$c_{11}$	$c_{12}$	$\dots$	$c_{1n}$	$s_1$
$W_2$	$c_{21}$	$c_{22}$	$\dots$	$c_{2n}$	$s_2$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
$W_m$	$c_{m1}$	$c_{m2}$	$\dots$	$c_{mn}$	$s_m$
	$d_1$	$d_2$	$\dots$	$d_n$	$\sum_{i=1}^m s_i \neq \sum_{j=1}^n d_j$

Transform to balanced problem ...

## Case I: demand exceeds supply

$$\sum_{i=1}^m s_i < \sum_{j=1}^n d_j$$

**Solution:** We introduce a fictitious warehouse  $W_{m+1}$  which supplies the excess demand.

- Set  $c_{m+1,j} = 0$  for all  $j = 1, 2, \dots, n$ .
- In reality we may use different costs—loss in sale, alternative supply, ...
- Interpretation—demand of some markets is not fully satisfied.

2	1	2	40
9	4	7	60
1	2	9	10
50	60	30	

## Case II: supply exceeds demand

$$\sum_{i=1}^m s_i > \sum_{j=1}^n d_j$$

**Solution:** We introduce a fictitious market  $M_{n+1}$  which demands the excess supply.

- Set  $c_{i,n+1} = 0$  for all  $i = 1, 2, \dots, m$ .
- In reality we may use different costs—spoilage costs, storage costs, ...
- Interpretation—goods “shipped” to the fictitious market retain in their respective warehouses.

2	1	2	50
9	4	7	70
1	2	9	20
40	50	20	



