Algebra and Discrete Mathematics (ADM)

Tutorial 10 Paths and spanning trees

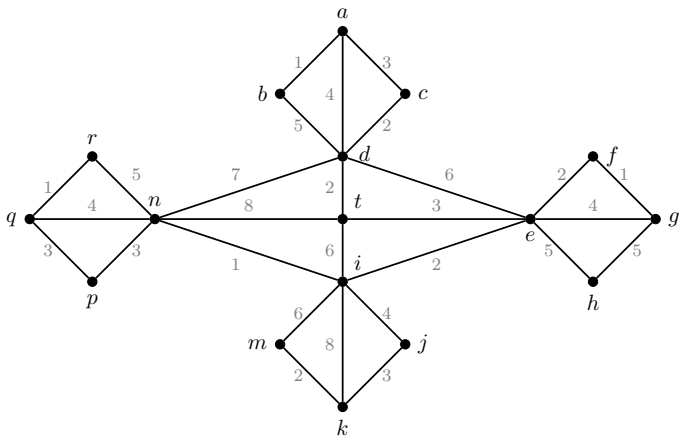
Lecturer: Bc. Xiaolu Hou, PhD.
xiaolu.hou@stuba.sk

Chinese Postman Problem

- Find an Eulerian circuit of minimal total weight
- Eulerization: find vertices with odd degree and pair these in the hopes of minimizing the weight along the path between each pair
- Now we combine the method for Eulerizing a graph and Dijkstra's Algorithm to provide a more complete answer to the Chinese Postman Problem.

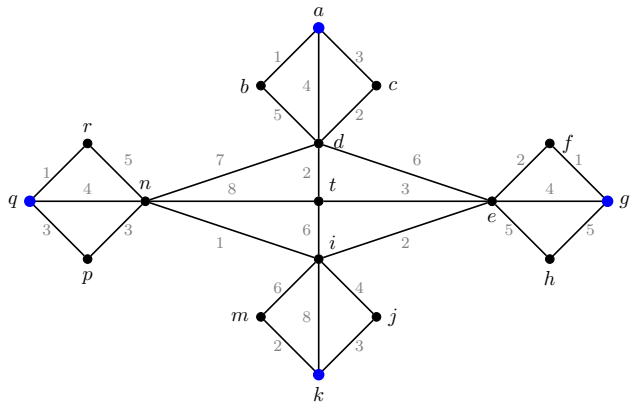
Chinese Postman Problem – example

Use Dijkstra's Algorithm to find the best pairing of odd vertices and the total weight of the edges duplicated in the Eulerization of the graph.



Chinese Postman Problem – example

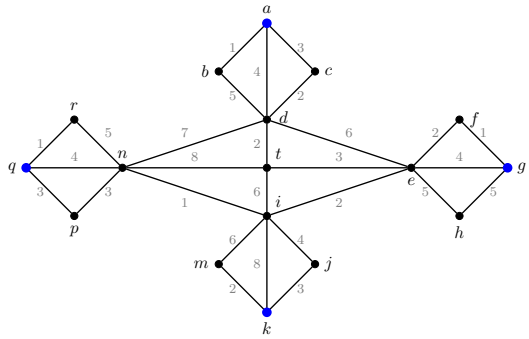
- Four odd vertices: a, g, k, q
- Three possible pairings $a-g, k-q$, or $a-k, g-q$, or $a-q, g-k$



Chinese Postman Problem – example

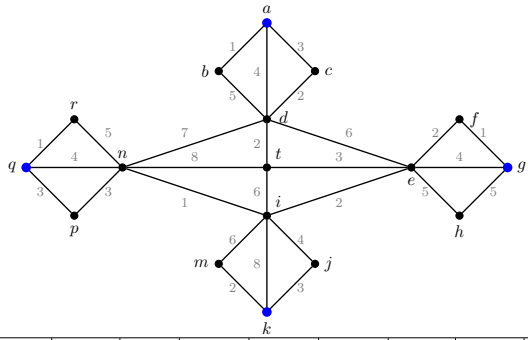
- Three possible pairings $a-g, k-q$, or $a-k, g-q$, or $a-q, g-k$
- Applying Dijkstra's Algorithm, we can find the shortest paths between the paired vertices and the total weight of the two paths needed to Eulerize the graph

Chinese Postman Problem – example



	b	c	d	n	e	t	r	q	p	f	g	h	i	m	j	k
	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$a(0)$	$(a,1)$	$(a,3)$	$(a,4)$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$b(1)$	$(a,1)$	$(a,3)$	$(a,4)$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$c(3)$	$(a,1)$	$(a,3)$	$(a,4)$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞

Chinese Postman Problem – example



	b	c	d	n	e	t	r	q	p	f	g	h	i	m	j	k
	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$a(0)$	$(a,1)$	$(a,3)$	$(a,4)$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$b(1)$	$(a,1)$	$(a,3)$	$(a,4)$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$c(3)$	$(a,1)$	$(a,3)$	$(a,4)$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$d(4)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(d,10)$	$(d,6)$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$t(6)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(t,9)$	$(d,6)$	∞	∞	∞	∞	∞	∞	$(t,12)$	∞	∞	∞
$e(9)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(t,9)$	$(d,6)$	∞	∞	∞	$(e,11)$	$(e,13)$	$(e,14)$	$(e,11)$	∞	∞	∞
$f(11)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(t,9)$	$(d,6)$	∞	∞	∞	$(e,11)$	$(f,12)$	$(e,14)$	$(e,11)$	∞	∞	∞
$i(11)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(t,9)$	$(d,6)$	∞	∞	∞	$(e,11)$	$(f,12)$	$(e,14)$	$(e,11)$	$(i,17)$	$(i,15)$	$(i,19)$
$n(11)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(t,9)$	$(d,6)$	$(n,16)$	$(n,15)$	$(n,14)$	$(e,11)$	$(f,12)$	$(e,14)$	$(e,11)$	$(i,17)$	$(i,15)$	$(i,19)$

Chinese Postman Problem – example

	b	c	d	n	e	t	r	q	p	f	g	h	i	m	j	k
	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$a(0)$	$(a,1)$	$(a,3)$	$(a,4)$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$b(1)$	$(a,1)$	$(a,3)$	$(a,4)$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$c(3)$	$(a,1)$	$(a,3)$	$(a,4)$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$d(4)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(d,10)$	$(d,6)$	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞
$t(6)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(t,9)$	$(d,6)$	∞	∞	∞	∞	∞	∞	$(t,12)$	∞	∞	∞
$e(9)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(t,9)$	$(d,6)$	∞	∞	∞	$(e,11)$	$(e,13)$	$(e,14)$	$(e,11)$	∞	∞	∞
$f(11)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(t,9)$	$(d,6)$	∞	∞	∞	$(e,11)$	$(f,12)$	$(e,14)$	$(e,11)$	∞	∞	∞
$i(11)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(t,9)$	$(d,6)$	∞	∞	∞	$(e,11)$	$(f,12)$	$(e,14)$	$(e,11)$	$(i,17)$	$(i,15)$	$(i,19)$
$n(11)$	$(a,1)$	$(a,3)$	$(a,4)$	$(d,11)$	$(t,9)$	$(d,6)$	$(n,16)$	$(n,15)$	$(n,14)$	$(e,11)$	$(f,12)$	$(e,14)$	$(e,11)$	$(i,17)$	$(i,15)$	$(i,19)$

$$adtefg$$

with weight 12

Chinese Postman Problem – example

- Three possible pairings $a-g, k-q$, or $a-k, g-q$, or $a-q, g-k$
- Applying Dijkstra's Algorithm, we can find the shortest paths between the paired vertices and the total weight of the two paths needed to Eulerize the graph

Path Pairs	Weight	Total Weight
$adtefg$	12	
$kjinq$	12	24
$adteijk$	18	
$gfeinq$	10	28
$adnq$	15	
$gfeijk$	12	27

Exercise: verify!

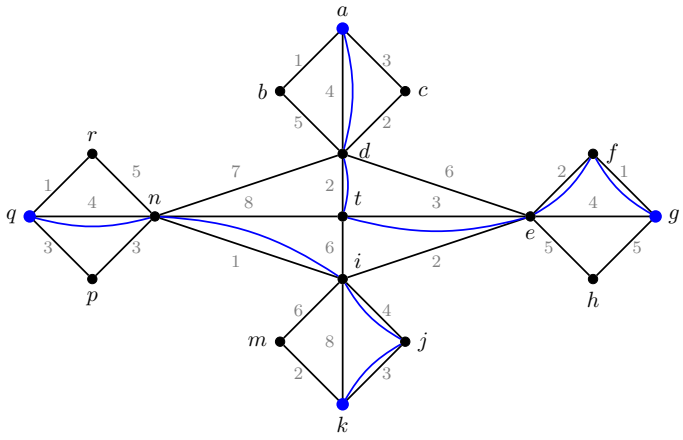
Chinese Postman Problem – example

- Once the paths are found, we duplicate the edges along these paths to obtain the Eulerization
- In doing so, we must be on alert for any edges that appear in both paths of a pairing
- For example, the paths in the second pairing both use the edge ei
- We should never duplicate an edge more than once during an Eulerization
- We modify the paths found by Dijkstra's Algorithm by removing both duplications of ei
- This maintains the degree condition (all vertices have even degree) and reduces the total weight by 4 – same as the first one

$adteijk$	18	
$gfeinq$	10	28

Chinese Postman Problem – example

- $adtefg$, $kjinq$
- $adteijk$, $gfeinq$



Remarks

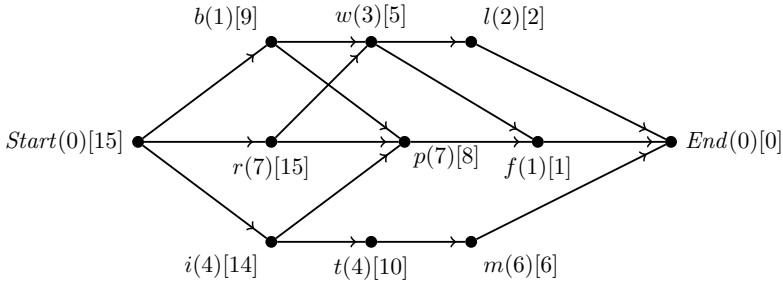
- The use of Dijkstra's Algorithm allows for a methodical approach to the Chinese Postman Problem.
- However, even for small examples the number of paths to find can be quite large.
- More efficient: matchings

Critical path and project scheduling

- This spring you are tackling the jungle that is your backyard. Some good friends have volunteered their time and you have split them into two groups.

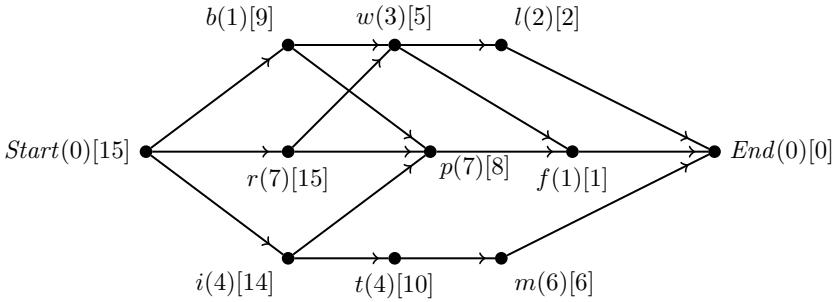
Task	**Vertex Name**	**Processing Time**	**Precedence Relationships**
Buy plants	b	1	
Remove bushes	r	7	
Remove ivy	i	4	
Weed flower beds	w	3	b, r
Plant bushes	p	7	b, r
Plant flowers	f	1	w, p
Trim trees	t	4	i
Mow and rake lawn	m	6	t
Install lighting	l	2	w

Critical path and project scheduling



$\mathrm{ct}[l] = 2,$
$\mathrm{ct}[m] = 6,$
$\mathrm{ct}[t] = \mathrm{pt}(t) + \mathrm{ct}[m] = 4 + 6 = 10,$
$\mathrm{ct}[i] = \mathrm{pt}(i) + \mathrm{ct}[t] = 4 + 10 = 14,$
$\mathrm{ct}[b] = \mathrm{pt}(b) + \mathrm{ct}[p] = 1 + 8 = 9,$
$\mathrm{ct}[f] = 1,$
$\mathrm{ct}[p] = \mathrm{pt}(p) + \mathrm{ct}[f] = 7 + 1 = 8,$
$\mathrm{ct}[w] = \mathrm{pt}(w) + \mathrm{ct}[l] = 3 + 2 = 5,$
$\mathrm{ct}[r] = \mathrm{pt}(r) + \mathrm{ct}[p] = 7 + 8 = 15,$
$\mathrm{ct}[\mathit{Start}] = \mathrm{pt}(\mathit{Start}) + \mathrm{ct}[r] = 0 + 15 = 15$

Critical path and project scheduling



The critical path is

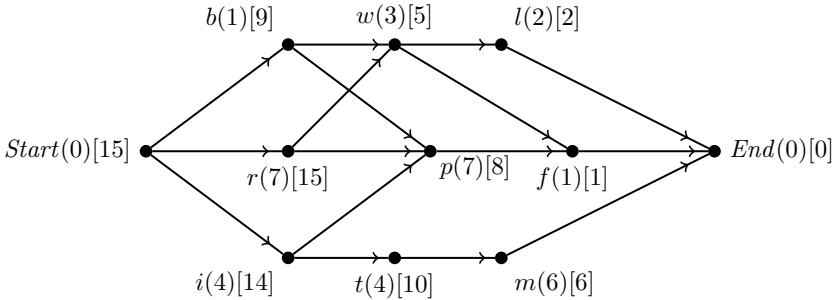
$$Start \rightarrow r \rightarrow p \rightarrow f \rightarrow End$$

The critical path priority list is

$$r - i - t - b - p - m - w - l - f$$

Critical path and project scheduling

$r - i - t - b - p - m - w - l - f$

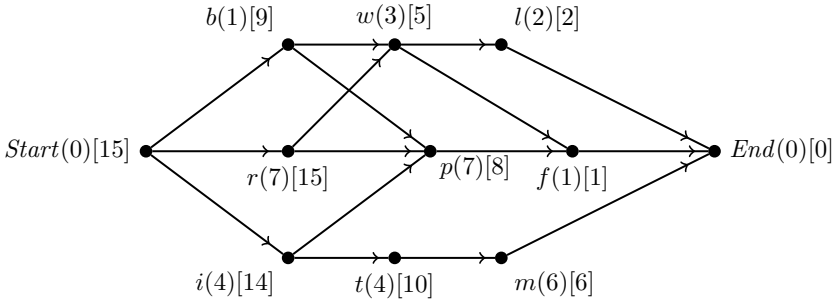


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
P_1	r	r	r	r	r	r	r												
P_2	i	i	i	i															

- Step 1. $(T = 0)$ Assign r to P_1, i to P_2

Critical path and project scheduling

$r - i - t - b - p - m - w - l - f$

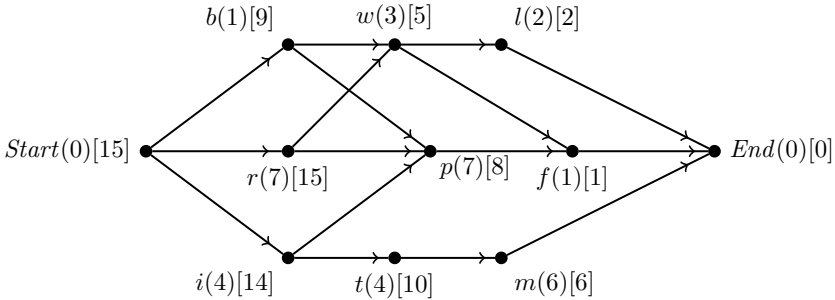


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
P_1	r	r	r	r	r	r	r												
P_2	i	i	i	i	t	t	t	t											

- Step 2. ($T = 4$) Assign t to P_2

Critical path and project scheduling

$r - i - t - b - p - m - w - l - f$

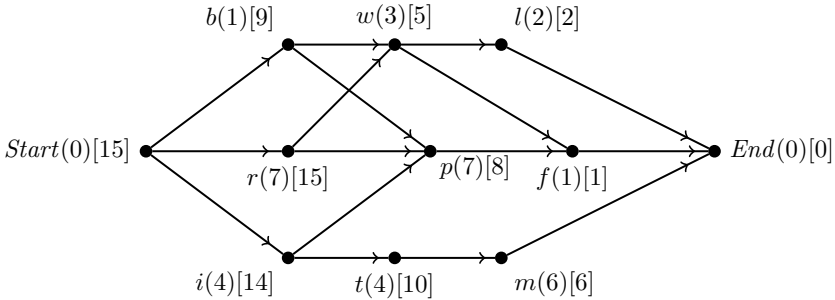


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
P_1	r	r	r	r	r	r	r	b											
P_2	i	i	i	i	t	t	t	t											

- Step 3. $(T = 7)$ Assign b to P_1

Critical path and project scheduling

$r - i - t - b - p - m - w - l - f$

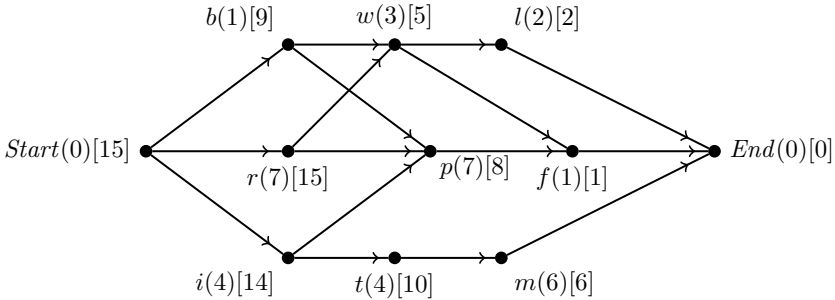


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
P_1	r	r	r	r	r	r	r	b	p	p	p	p	p	p	p				
P_2	i	i	i	i	t	t	t	t	m	m	m	m	m	m					

- Step 4. ($T = 8$) Assign p to P_1, m to P_2

Critical path and project scheduling

$r - i - t - b - p - m - w - l - f$

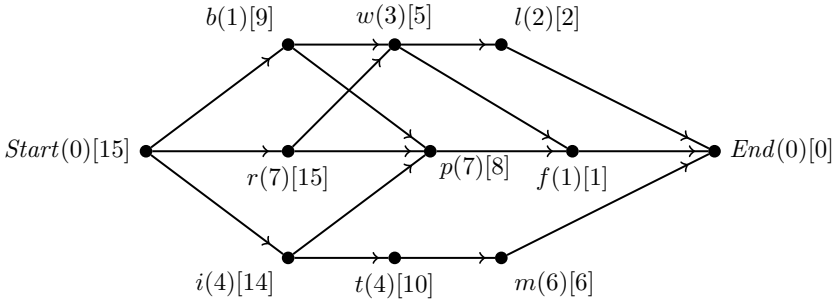


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
P_1	r	r	r	r	r	r	r	b	p	p	p	p	p	p	p				
P_2	i	i	i	i	t	t	t	t	m	m	m	m	m	m	w	w	w		

- Step 5. ($T = 14$) Assign w to P_2

Critical path and project scheduling

$r - i - t - b - p - m - w - l - f$

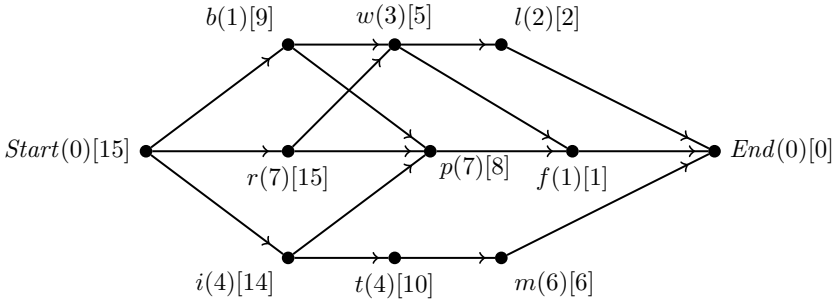


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
P_1	r	r	r	r	r	r	r	b	p	p	p	p	p	p	p	$*$	$*$		
P_2	i	i	i	i	t	t	t	t	m	m	m	m	m	m	w	w	w		

- Step 6. ($T = 15$) No eligible task. P_1 will remain idle.

Critical path and project scheduling

$r - i - t - b - p - m - w - l - f$



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
P_1	r	r	r	r	r	r	r	b	p	p	p	p	p	p	p	$*$	$*$	l	l
P_2	i	i	i	i	t	t	t	t	m	m	m	m	m	m	w	w	w	f	$*$

- Step 7. ($T = 17$) Assign l to P_1 and f to P_2

Critical path and project scheduling

- By using the two metrics described in the lecture, we know

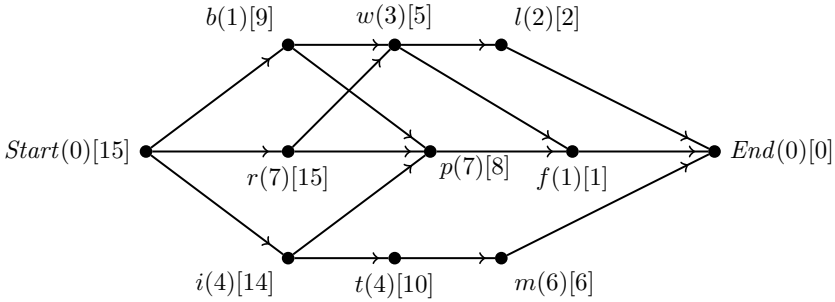
$$OPT \geq 15, \quad OPT \geq \frac{35}{2} = 17.5 \text{ for two processors}$$

- Since there is no half-unit of time, we know the optimal time in fact must be at least 18 hours.
- The schedule we have found has 3 total hours of idle time
- w did not place high on the priority list, but needed to be completed earlier since both l and f relied upon its completion
- The optimal schedule is as follows

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
P_1	r	r	r	r	r	r	r	p	p	p	p	p	p	p	l	l	f	$*$
P_2	i	i	i	i	b	t	t	t	t	w	w	w	m	m	m	m	m	m

What if we use 3 processors?

$r - i - t - b - p - m - w - l - f$

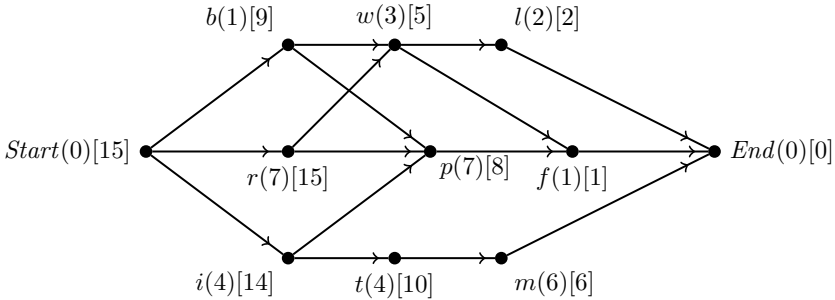


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
P_1	r	r	r	r	r	r	r											
P_2	i	i	i	i														
P_3	b	$*$	$*$	$*$														

At $T = 1$, there are no eligible tasks

What if we use 3 processors?

$r - i - t - b - p - m - w - l - f$

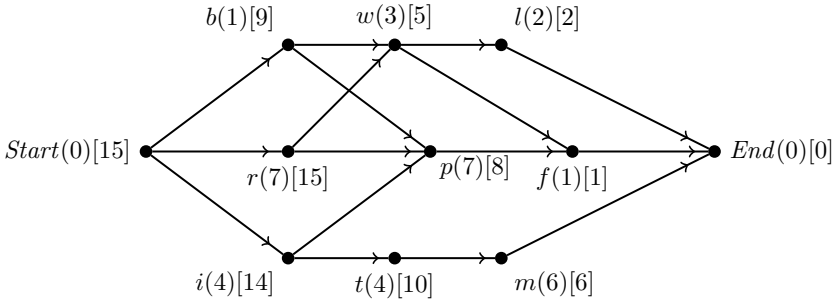


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
P_1	r	r	r	r	r	r	r	p	p	p	p	p	p	p				
P_2	i	i	i	i	t	t	t	t										
P_3	b	$*$	$*$	$*$	$*$	$*$	$*$	w	w	w								

At $T = 4$, the only eligible task is t. At $T = 7$, the eligible tasks are p and w

What if we use 3 processors?

$r - i - t - b - p - m - w - l - f$

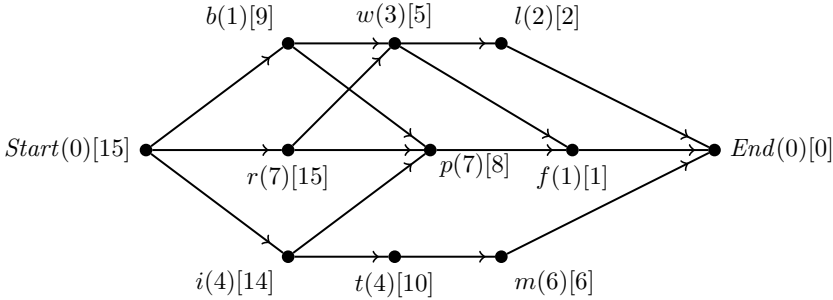


	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
P_1	r	r	r	r	r	r	r	p	p	p	p	p	p	p				
P_2	i	i	i	i	t	t	t	t	m	m	m	m	m	m				
P_3	b	$*$	$*$	$*$	$*$	$*$	$*$	w	w	w	l	l						

At $T = 8$, the next eligible task is m. At $T = 10$, the next eligible task is l.

What if we use 3 processors?

$r - i - t - b - p - m - w - l - f$



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
P_1	r	r	r	r	r	r	r	p	p	p	p	p	p	p	f			
P_2	i	i	i	i	t	t	t	t	m	m	m	m	m	m	$*$			
P_3	b	$*$	$*$	$*$	$*$	$*$	$*$	w	w	w	l	l	$*$	$*$	$*$			

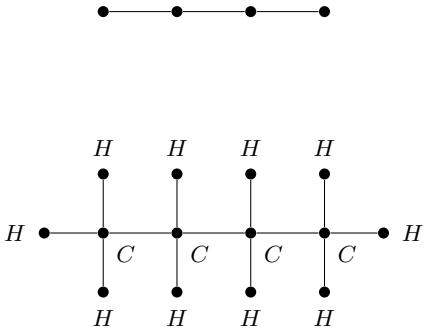
The last task f is only eligible after p is completed. Thus, we get an optimal schedule having a finishing time of 15 hours.

Tree – example

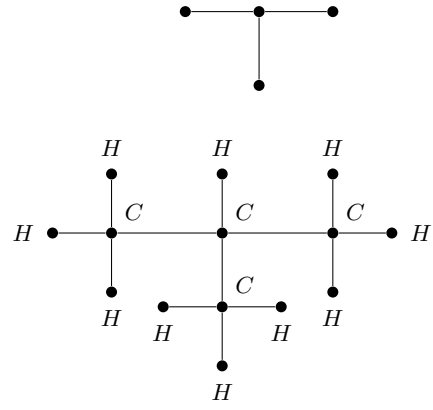
- Chemical Graph Theory uses concepts from graph theory to obtain results about chemical compounds
- Individual atoms in a molecule are represented by vertices and an edge denotes a bond between the atoms
- One way to determine the number of isomers (compounds with the same formula but a different arrangement of atoms) for a molecule is to determine the number of distinct graphs that contain the correct type of each atom
- For hydrocarbons (molecules only containing carbon and hydrogen atoms) the hydrogen-depleted graph is used since the bonds between the carbon atoms will uniquely determine the locations of the hydrogen atoms.

Tree – example

- Trees on four vertices: only two choices
- $\rightarrow$ the only possible isomers of butane C_4H_{10}
- By using graph theory, it can be proven that no other isomers of butane are possible since no other trees on four vertices exist



n-butane



isobutane

Kruskal's Algorithm – example

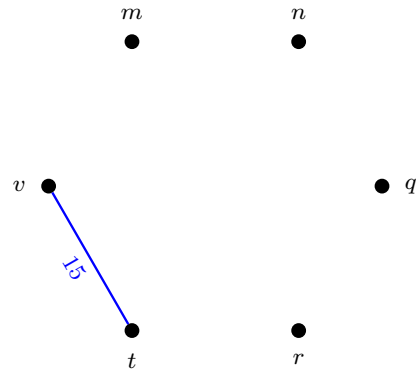
- The Optos Cable Company is expanding its fiber optic network over the next few years
- The company will need to lay new cable, but wishes to do so with minimal cost
- Cost of the cable is fixed per meter
- The distances of required cables between any two towns is given in the table

	Mesa	Natick	Quechee	Rutland	Tempe	Vinton
Mesa	·	18	35	36	20	45
Natick	18	·	50	42	40	45
Quechee	35	50	·	41	25	19
Rutland	36	42	41	·	37	38
Tempe	20	40	25	37	·	15
Vinton	45	45	19	38	15	·

Kruskal's Algorithm – example

- Apply Kruskal's Algorithm, the edges will be chosen directly from the table
- Smallest weight is t-v

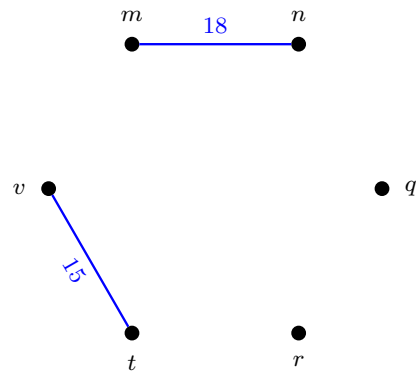
	Mesa	Natick	Quechee	Rutland	Tempe	Vinton
Mesa	·	18	35	36	20	45
Natick	18	·	50	42	40	45
Quechee	35	50	·	41	25	19
Rutland	36	42	41	·	37	38
Tempe	20	40	25	37	·	15
Vinton	45	45	19	38	15	·



Kruskal's Algorithm – example

- The next smallest weight is 18 for m-n

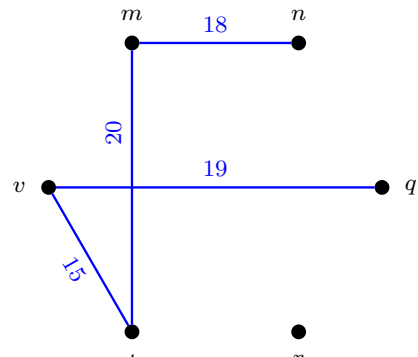
	Mesa	Natick	Quechee	Rutland	Tempe	Vinton
Mesa	·	18	35	36	20	45
Natick	18	·	50	42	40	45
Quechee	35	50	·	41	25	19
Rutland	36	42	41	·	37	38
Tempe	20	40	25	37	·	15
Vinton	45	45	19	38	15	·



Kruskal's Algorithm – example

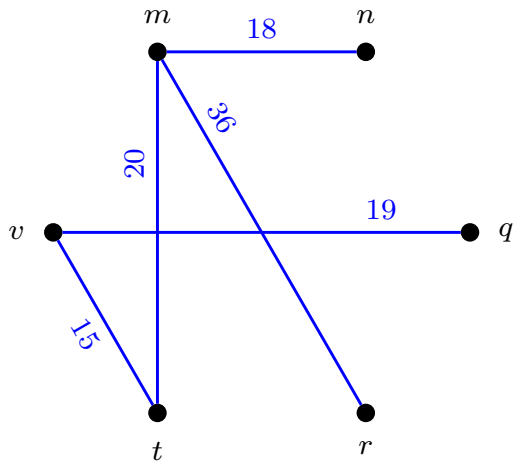
- The next smallest weight is 19 for q-v
- Next is 20 for m-t
- The next smallest weight is 25 for q-t. We need to skip it
- The next is m-q with 35, skip
- Next is m-r with 36, highlight mr

	Mesa	Natick	Quechee	Rutland	Tempe	Vinton
Mesa	·	18	35	36	20	45
Natick	18	·	50	42	40	45
Quechee	35	50	·	41	25	19
Rutland	36	42	41	·	37	38
Tempe	20	40	25	37	·	15
Vinton	45	45	19	38	15	·



Kruskal's Algorithm – example

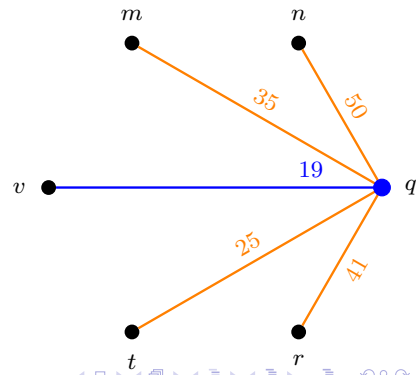
- We have a spanning tree of total weight 180



Prim's Algorithm – example

- Let's solve the same problem with Prim's Algorithm
- The drawing of the underlying complete graph is omitted
- Let us start from q
- Edge with smallest weight: qv

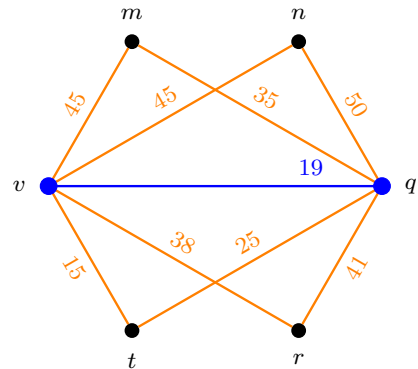
	Mesa	Natick	Quechee	Rutland	Tempe	Vinton
Mesa	·	18	35	36	20	45
Natick	18	·	50	42	40	45
Quechee	35	50	·	41	25	19
Rutland	36	42	41	·	37	38
Tempe	20	40	25	37	·	15
Vinton	45	45	19	38	15	·



Prim's Algorithm – example

- Next search for the edge of smallest weight with exactly one endpoint q or v
- Edge with smallest weight: vt

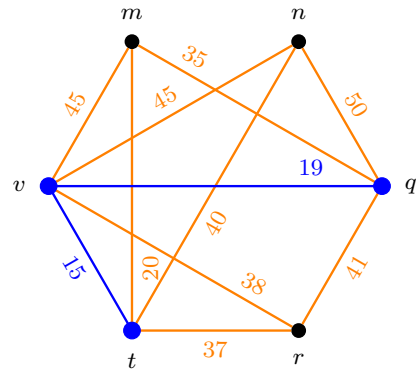
	Mesa	Natick	Quechee	Rutland	Tempe	Vinton
Mesa	·	18	35	36	20	45
Natick	18	·	50	42	40	45
Quechee	35	50	·	41	25	19
Rutland	36	42	41	·	37	38
Tempe	20	40	25	37	·	15
Vinton	45	45	19	38	15	·



Prim's Algorithm – example

- Next search for the edge of smallest weight with exactly one endpoint q, t, or v
- Edge with smallest weight: tm

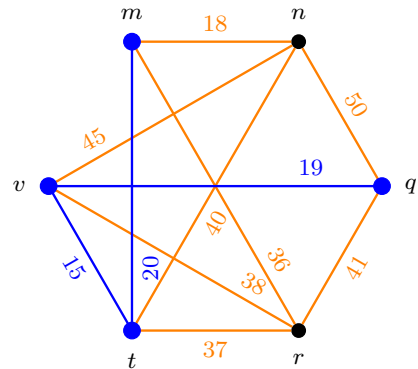
	Mesa	Natick	Quechee	Rutland	Tempe	Vinton
Mesa	·	18	35	36	20	45
Natick	18	·	50	42	40	45
Quechee	35	50	·	41	25	19
Rutland	36	42	41	·	37	38
Tempe	20	40	25	37	·	15
Vinton	45	45	19	38	15	·



Prim's Algorithm – example

- Next search for the edge of smallest weight with exactly one endpoint q, t, v, or m
- Edge with smallest weight: mn

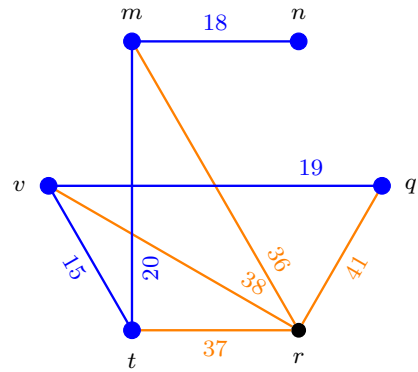
	Mesa	Natick	Quechee	Rutland	Tempe	Vinton
Mesa	·	18	35	36	20	45
Natick	18	·	50	42	40	45
Quechee	35	50	·	41	25	19
Rutland	36	42	41	·	37	38
Tempe	20	40	25	37	·	15
Vinton	45	45	19	38	15	·



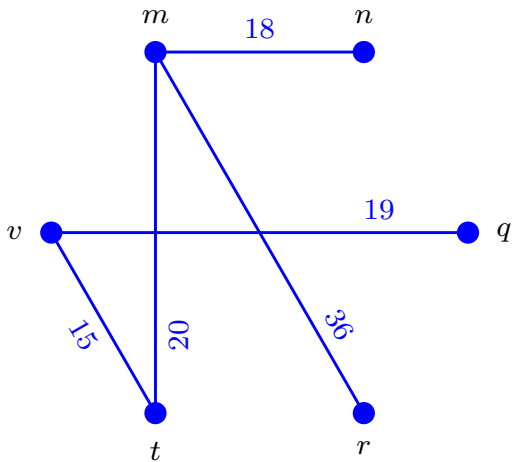
Prim's Algorithm – example

- There is only one vertex left to add to the spanning tree, and so we must find the smallest edge to r
- Edge with smallest weight: mr

	Mesa	Natick	Quechee	Rutland	Tempe	Vinton
Mesa	·	18	35	36	20	45
Natick	18	·	50	42	40	45
Quechee	35	50	·	41	25	19
Rutland	36	42	41	·	37	38
Tempe	20	40	25	37	·	15
Vinton	45	45	19	38	15	·



Prim's Algorithm – example



- MST has total weight 108