The Travelling Salesperson Problem: A tour of the main points

Continuing Professional Development Standard

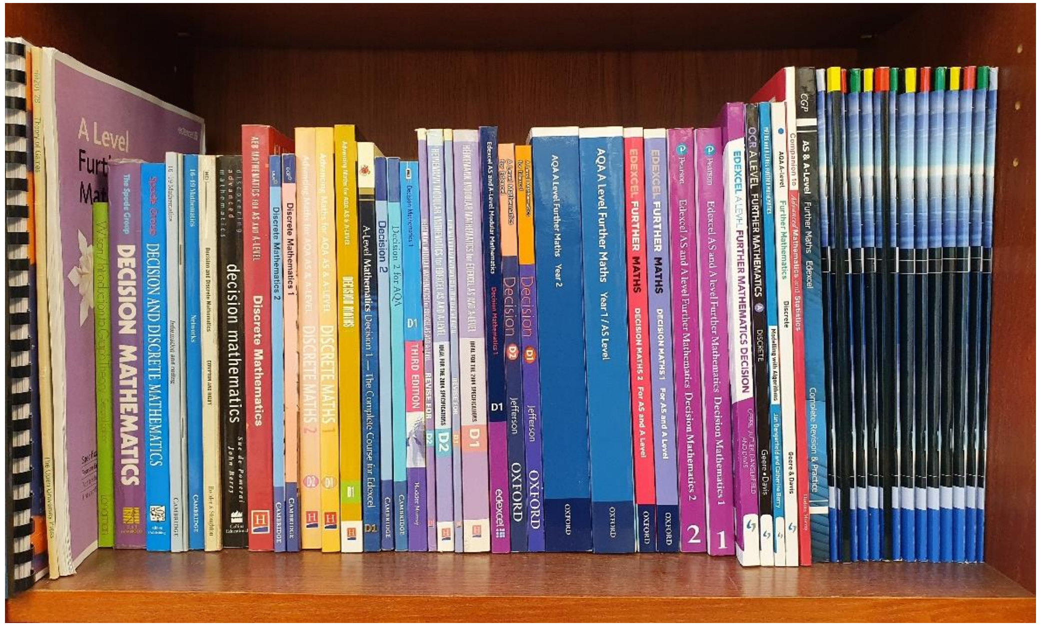
National Centre for Excellence in the Teaching of Mathematics

Jeff Trim

jeff.trim@mei.org.uk

Background and rationale

- It is possible for students to become proficient in the various procedures required for the Travelling Salesperson Problem without ever grasping the bigger picture.
- Aim: to tie several threads together into a holistic understanding



Session outline

- Requirements of the specifications
- The Big Picture
- Required techniques
- Some common questions
- Three examples
- Tying up the loose ends

- Suitable for teachers who are already familiar with the basics of the Travelling Salesperson Problem.

Algorithms on Networks topic mapping

Discrete/Decision Mathematics Topics	AQA	Edexcel	MEI	OCR A
Algorithms on Networks				
Modelling problems using weighted graphs	AS	AS D1	MwA	AS
Adjacency/Incidence matrix	AS	AS D1	MwA	AS
Minimum connector problem, Kruskal and Prim	AS	AS D1	MwA	AS
Shortest path problem, Dijkstra		AS D1	MwA	AS
Complexity of Kruskal's/Prim's/Dijkstra's algorithms			MwA	AS
Floyd's algorithm		A Level D1		
Route inspection problem (Chinese Postman)	AS	AS D1		A Level
Travelling salesperson problem	AS	A Level D1		A Level
Network Flows	AS	AS D2	MwA	
Solving network problems using technology			MwA	

Travelling Salesperson topic content

Discrete/Decision Mathematics Topics	AQA	Edexcel	MEI	OCR A
Travelling salesperson problem				
Modelling problems using weighted graphs	AS	AS D1	MwA	AS
Hamiltonian cycles and Hamiltonian graphs	AS	A Level D1		A Level
Finding a least weight cycle through all vertices	AS	A Level D1		A Level
The Practical problem and the Classical problem	AS	A Level D1		A Level
Use of a minimum spanning tree to find an upper bound	AS	A Level D1		
Shortcuts to improve an upper bound		A Level D1		A Level
Conversion of a problem to a complete network on K_n	AS	A Level D1		A Level
Using a table of shortest distances	AS	A Level D1		A Level
Nearest Neighbour algorithm to find an upper bound	AS	A Level D1		A Level
Lower bound algorithm	AS	A Level D1		A Level
Expressing the result as an inequality	AS	A Level D1		A Level
Solving network problems using technology			MwA	

The Travelling Salesperson Problem (TSP)

The Big Picture

- Mathematical modelling of a real-life situation
- Objective: To find a Hamiltonian cycle of minimum length
- The Millennium Problem: P versus NP
- The Classical problem versus the Practical problem

Required techniques:

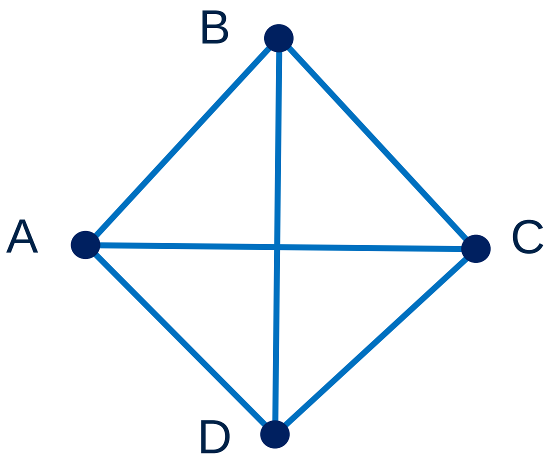
- Three methods for finding an Upper Bound
- One method for finding a Lower Bound
- Expressing the result as an inequality

Some common questions

- How many different tours are there?
- What is “P v NP”?
- How does the Classical problem differ from the Practical problem?
- Why is my Lower Bound bigger than my Upper Bound?
- What is the point of a Lower Bound that is clearly not a tour?
- Should I use ≤ or < ?

How many different tours are there?

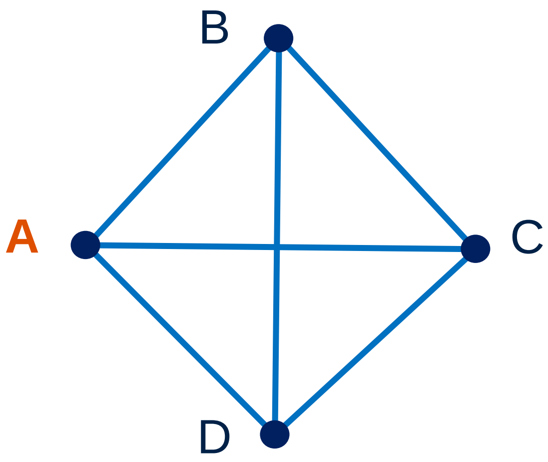
Counting tours



For the Classical problem we are enumerating the tours of length *n* around the *n* vertices in K_n.

Let's take as our example K_4.

Counting tours

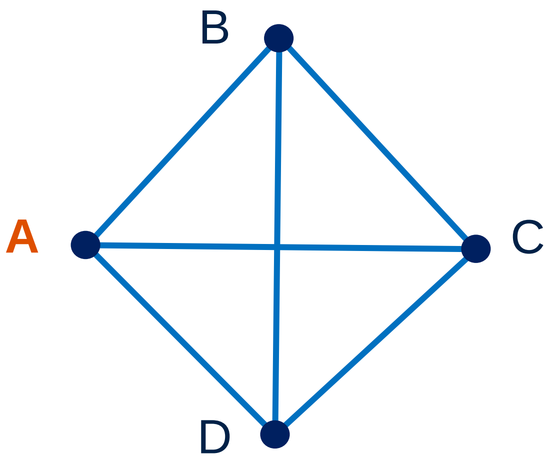


Let's begin by finding all the tours that start at A.
(Four choices)

Tour

A

Counting tours



The second vertex can be any of the remaining three. (Three choices)

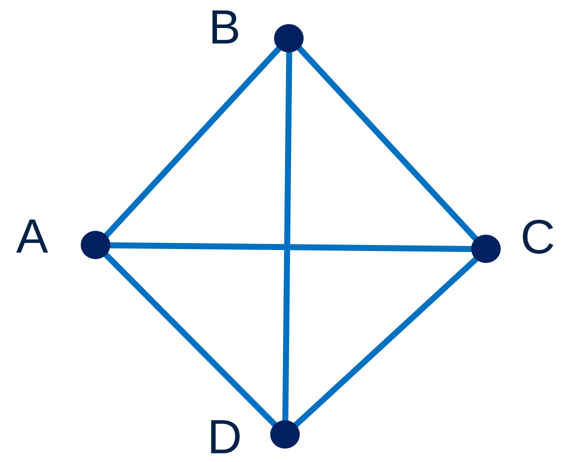
Tour

A B

A C

A D

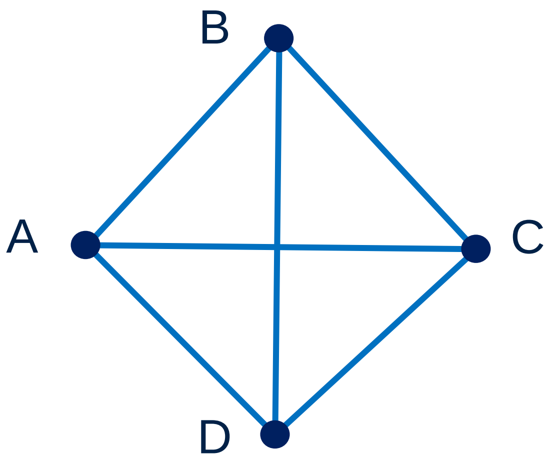
Counting tours



The third vertex is now a choice between the remaining two. (Two choices)

Tour
A B C
A B D
A C B
A C D
A D B
A D C

Counting tours



The fourth vertex has to be the only one not included in the tour so far. (Only one choice!)

Tour

A B C D

A B D C

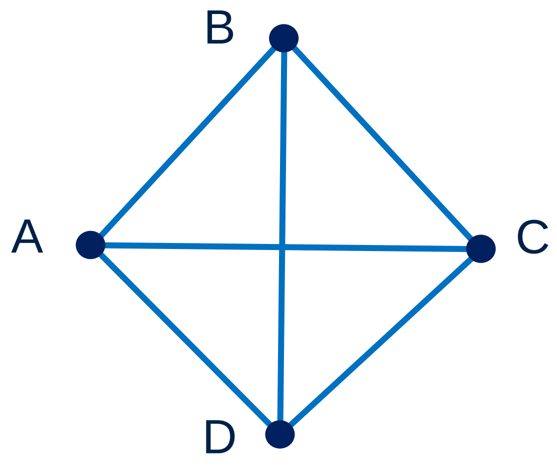
A C B D

A C D B

A D B C

A D C B

Counting tours

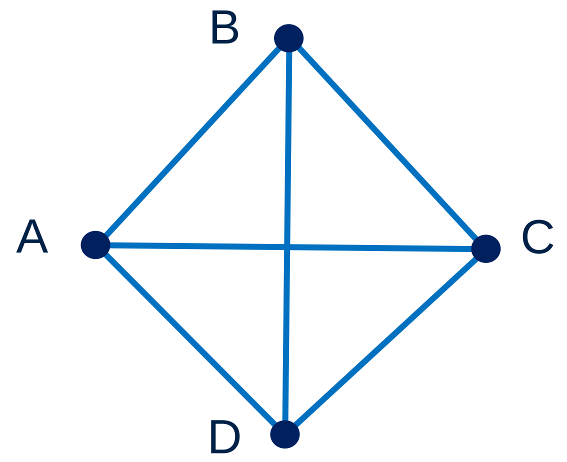


Each Hamiltonian cycle will continue visiting the vertices in the same sequence.

Tour				Continuation of Tour								
A	B	C	D	A	B	C	D	A	B	C	D	…
A	B	D	C	A	B	D	C	A	B	D	C	…
A	C	B	D	A	C	B	D	A	C	B	D	…
A	C	D	B	A	C	D	B	A	C	D	B	…
A	D	B	C	A	D	B	C	A	D	B	C	…
A	D	C	B	A	D	C	B	A	D	C	B	…

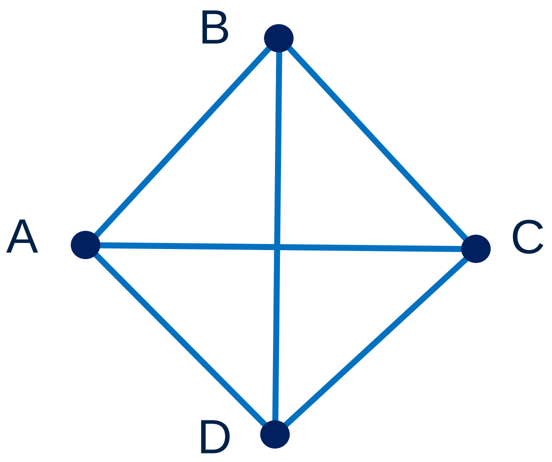
Counting tours

Each cycle could have started at one of the other vertices.



Tour	Continuation of Tour	Related Tours
A B C D	A B C D A B C D …	B C D A
A B D C	A B D C A B D C …	B D C A
A C B D	A C B D A C B D …	C B D A
A C D B	A C D B A C D B …	C D B A
A D B C	A D B C A D B C …	D B C A
A D C B	A D C B A D C B …	D C B A

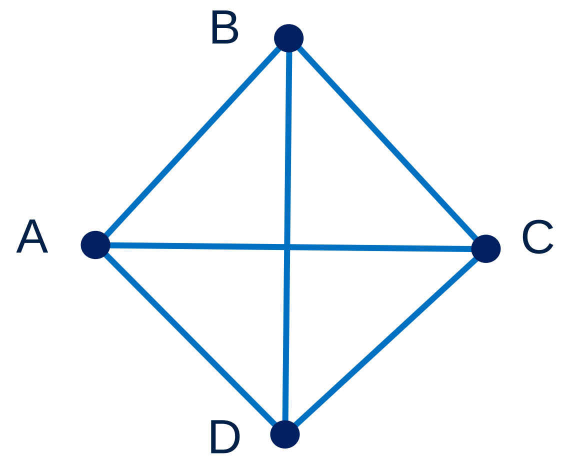
Counting tours



Each cycle could have started at one of the other vertices.

Tour				Continuation of Tour									Related Tours							
A	B	C	D	A	B	C	D	A	B	C	D	…	B	C	D	A	C	D	A	B
A	B	D	C	A	B	D	C	A	B	D	C	…	B	D	C	A	D	C	A	B
A	C	B	D	A	C	B	D	A	C	B	D	…	C	B	D	A	B	D	A	C
A	C	D	B	A	C	D	B	A	C	D	B	…	C	D	B	A	D	B	A	C
A	D	B	C	A	D	B	C	A	D	B	C	…	D	B	C	A	B	C	A	D
A	D	C	B	A	D	C	B	A	D	C	B	…	D	C	B	A	C	B	A	D

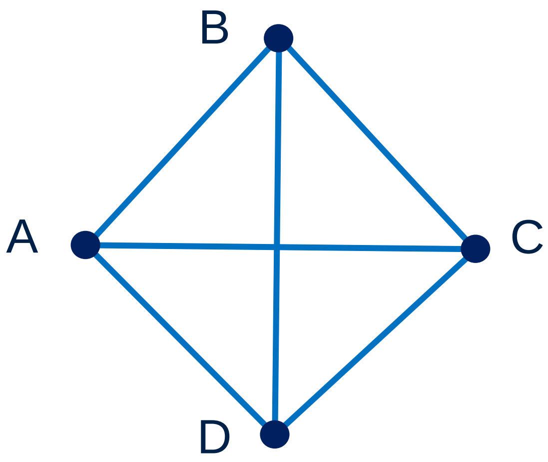
Counting tours



Each cycle could have started at one of the other vertices.

Tour				Continuation of Tour									Related Tours											
A	B	C	D	A	B	C	D	A	B	C	D	…	B	C	D	A	C	D	A	B	D	A	B	C
A	B	D	C	A	B	D	C	A	B	D	C	…	B	D	C	A	D	C	A	B	C	A	B	D
A	C	B	D	A	C	B	D	A	C	B	D	…	C	B	D	A	B	D	A	C	D	A	C	B
A	C	D	B	A	C	D	B	A	C	D	B	…	C	D	B	A	D	B	A	C	B	A	C	D
A	D	B	C	A	D	B	C	A	D	B	C	…	D	B	C	A	B	C	A	D	C	A	D	B
A	D	C	B	A	D	C	B	A	D	C	B	…	D	C	B	A	C	B	A	D	B	A	D	C

Counting tours



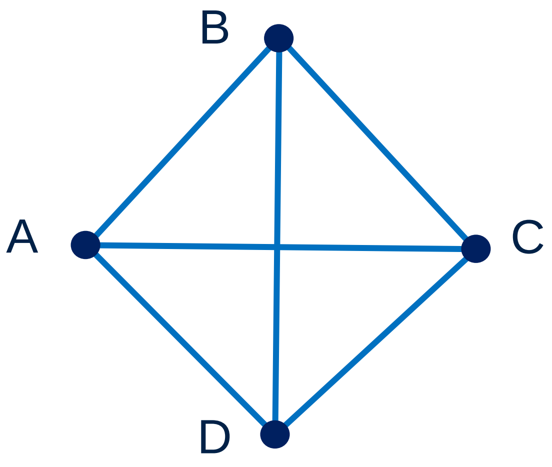
There are 4! permutations of the 4 letters.

In general:

There are n! tours if the starting point matters.

Tour	Related Tours		
A B C D	B C D A	C D A B	D A B C
A B D C	B D C A	D C A B	C A B D
A C B D	C B D A	B D A C	D A C B
A C D B	C D B A	D B A C	B A C D
A D B C	D B C A	B C A D	C A D B
A D C B	D C B A	C B A D	B A D C

Counting tours



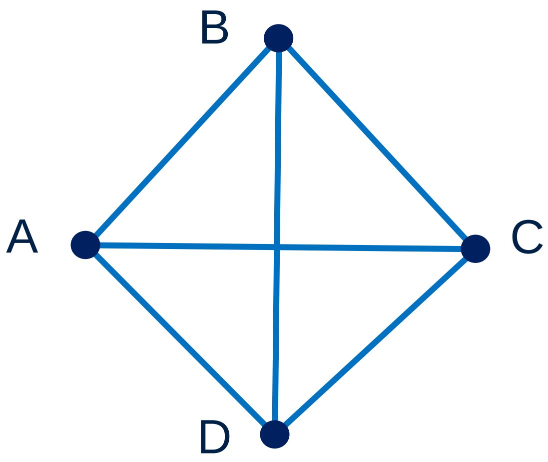
However, these cycles are related in groups of 4.

In general:

There are $n!/n = (n-1)!$ different cycles.

Tour	Related Tours		
A B C D	B C D A	C D A B	D A B C
A B D C	B D C A	D C A B	C A B D
A C B D	C B D A	B D A C	D A C B
A C D B	C D B A	D B A C	B A C D
A D B C	D B C A	B C A D	C A D B
A D C B	D C B A	C B A D	B A D C

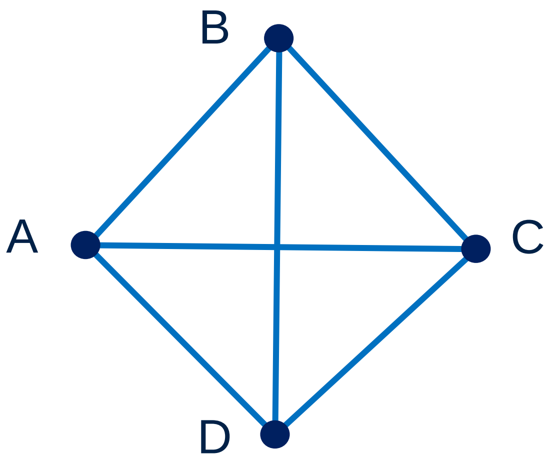
Counting tours



Each cycle could be travelled in either direction.

Tour	Related Tours		
A B C D	B C D A	C D A B	D A B C
A B D C	B D C A	D C A B	C A B D
A C B D	C B D A	B D A C	D A C B
A C D B	C D B A	D B A C	B A C D
A D B C	D B C A	B C A D	C A D B
A D C B	D C B A	C B A D	B A D C

Counting tours



The cycles match up into pairs of equal length.

In general:

There are ½(n-1)! different cycle lengths.

Tour	Related Tours		
A B C D	B C D A	C D A B	D A B C
A B D C	B D C A	D C A B	C A B D
A C B D	C B D A	B D A C	D A C B
A C D B	C D B A	D B A C	B A C D
A D B C	D B C A	B C A D	C A D B
A D C B	D C B A	C B A D	B A D C

Counting tours

Summary:		For K_4
$n!$	Tours, if the starting point matters	$4! = 24$
$(n-1)!$	different cycles	$3! = 6$
$\frac{1}{2}(n-1)!$	different cycle lengths	$\frac{1}{2} \times 3! = 3$

Counting tours

Summary:

$n!$	Tours, if the starting point matters
$(n-1)!$	different cycles
$\frac{1}{2}(n-1)!$	different cycle lengths

Graph	No. of vertices	No. of cycles to check
K_5	5	½ x 4! = 12
K_{10}	10	½ x 9! = 181 440
K_{20}	20	½ x 19! = over 60 000 trillion

What is "P v NP"?

The P versus NP problem

This is one of seven ‘Millennium Prize’ problems selected by the Clay Institute, offering a $1 000 000 reward for the solution.

If the solution to a problem is easy to check for correctness, must the problem be easy to solve?

P problems: solved quickly

Algorithm runs in Polynomial time (P)
[not exponential or factorial]

NP problems: answer verified quickly

Solution *verified* in Polynomial time
NP: Nondeterministic Polynomial time

The P versus NP problem

Is P = NP? or Is P ≠ NP?

John Nash first proposed a case where P ≠ NP in 1955.

The P versus NP problem was defined in 1971 by Stephen Cook.

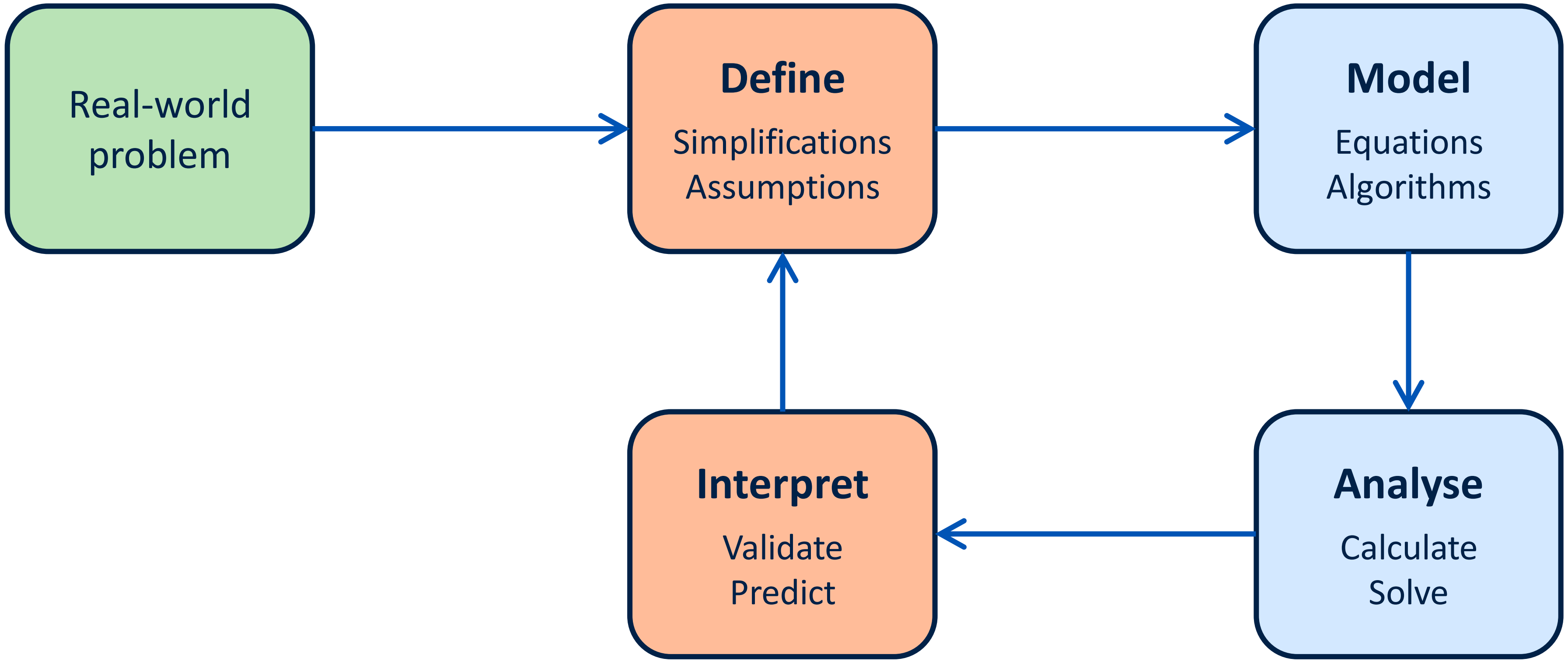
> *“Can problems that can be verified in polynomial time always be solved in polynomial time?”*

The Travelling Salesperson problem has factorial order, O(n!).

How does the Classical problem differ from the Practical problem?

Practical problem vs Classical problem

The modelling cycle:



Practical problem vs Classical problem

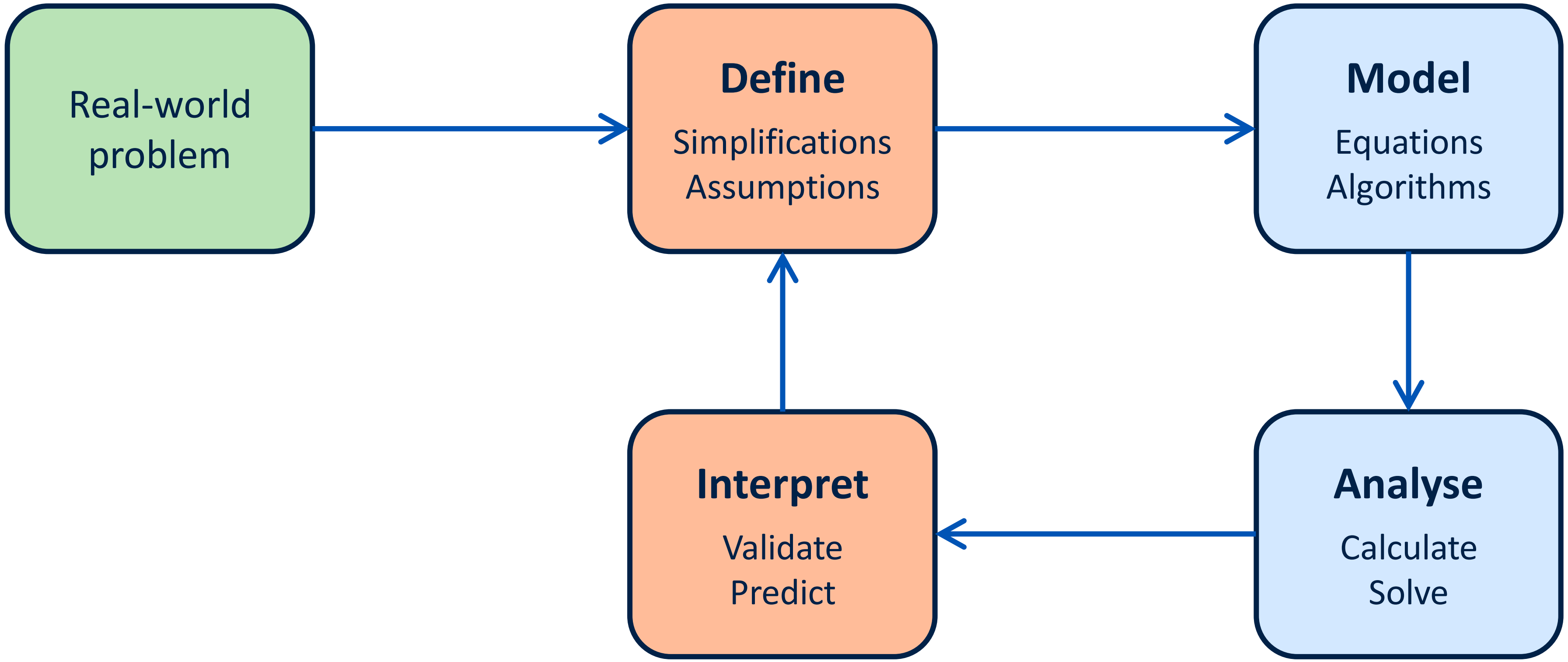
The modelling cycle:

Practical Problem:

Optimal tour on an incomplete graph

Classical Problem:

Optimal H-cycle on a complete graph, K_n



Methods:

Use of MST
Nearest Neighbour
& LB algorithms

Practical problem vs Classical problem

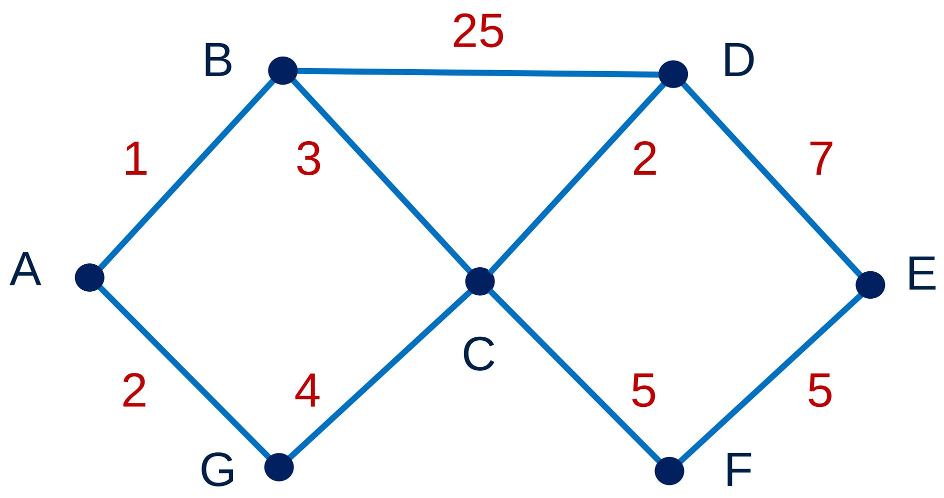
Classical problem	**Practical problem**	**Resolving the difference**
n vertices	*n* vertices	-
complete graph K_n ½ n(n-1) edges	incomplete graph K_n < ½ n(n-1) edges	table of least distances (treat as single edges)
n edges in tour	> *n* edges may be needed	allow ≥ *n* edges
no vertex repeated	may need repeats	allow repeated vertices
c/w tour = a/cw tour	may have directed edges	intuitive approach
Mathematical model	**Real-world problem**	**Interpretation**

Practical problem vs Classical problem

Classical problem	Practical problem	Resolving the difference
n vertices	*n* vertices	-
complete graph K_n ½ n(n-1) edges	incomplete graph K_n < ½ n(n-1) edges	table of least distances (treat as single edges)
n edges in tour	> *n* edges may be needed	allow ≥ *n* edges
no vertex repeated	may need repeats	allow repeated vertices
c/w tour = a/cw tour	may have directed edges	intuitive approach
Mathematical model	**Real-world problem**	**Interpretation**

Why is my Lower Bound bigger than my Upper Bound?

An interesting example



a) Find the minimum spanning tree for this network.

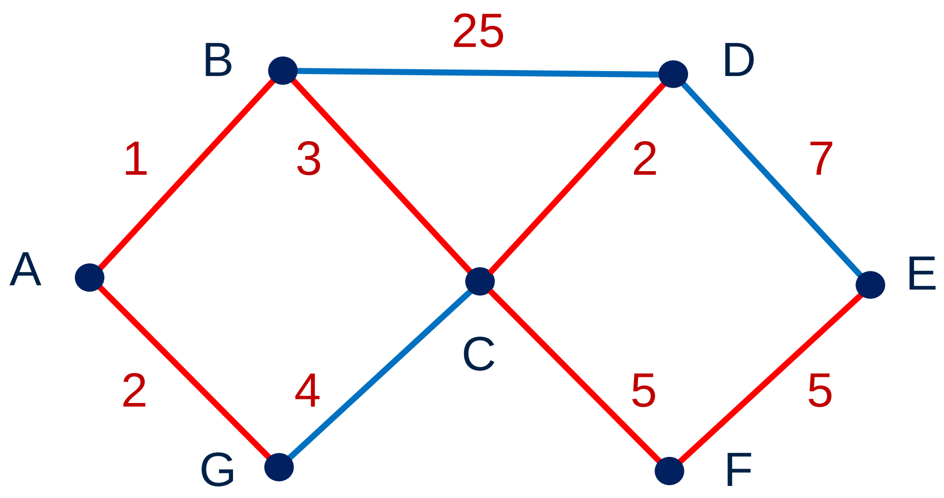
b) Hence state an upper bound for the travelling salesperson problem.

c) Identify a tour which gives an improved upper bound.

d) Use the LB algorithm to find a lower bound.

e) Express your results in the form of an inequality for T, the length of the optimal tour.

An interesting example



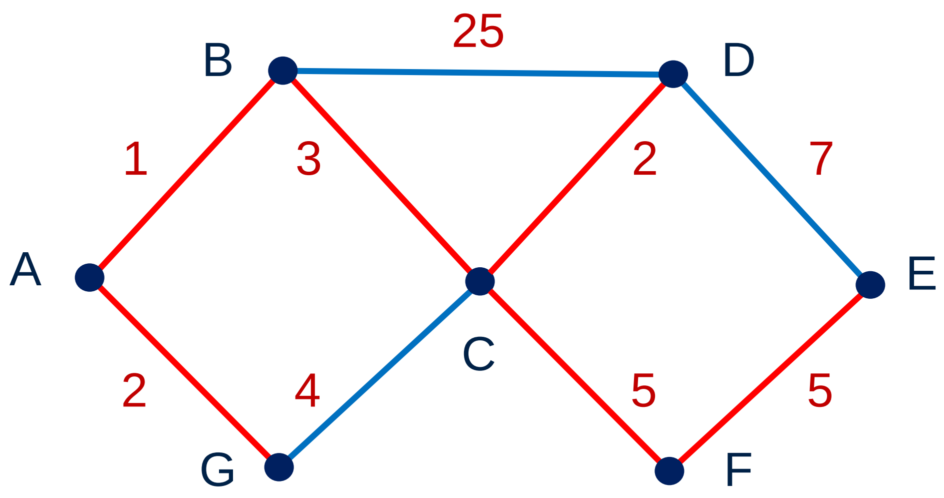
a) Find the minimum spanning tree for this network.

7 vertices, so 6 edges required:

AB	1
AG	2
CD	2
BC	3
~~CG~~	~~4~~ *(forms a cycle)*
CF	5
EF	5

Total length 18

An interesting example



UB = 36

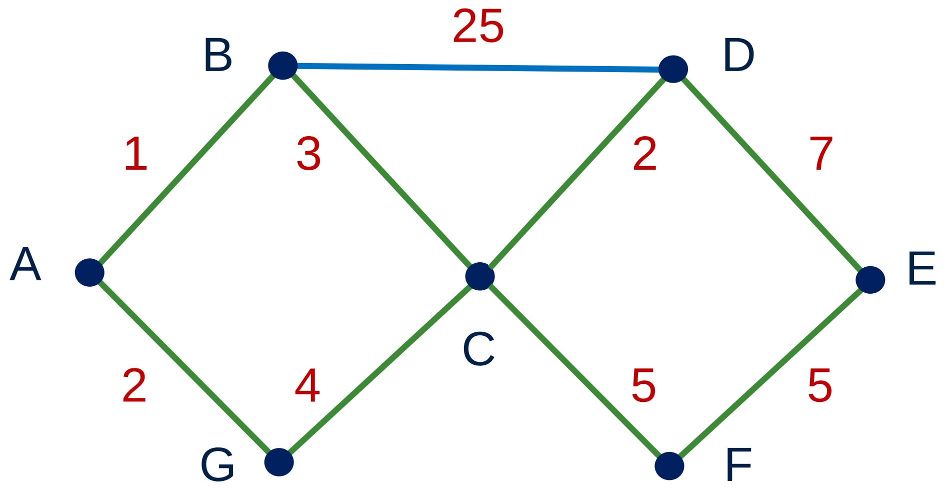
b) Hence state an upper bound for the travelling salesperson problem.

UB = 2 x MST

UB = 2 x 18

UB = 36

An interesting example



~~UB = 36~~

UB = 29

c) Identify a tour which gives an improved upper bound.

A-B-D-E-F-C-G-(A) is a tour without repeated vertices, but includes that 25.

Consider instead:

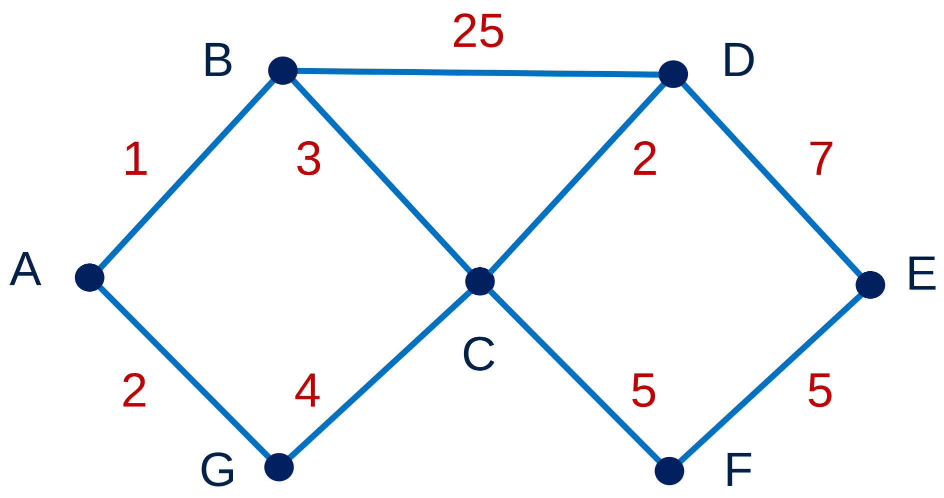
A – B – C – D – E – F – C – G – (A)

1 + 3 + 2 + 7 + 5 + 5 + 4 + 2 = 29

Any tour is an UB because it is a feasible solution and may be improved upon:

UB = 29

An interesting example



~~UB = 36~~

UB = 29

d) Use the LB algorithm to find a lower bound.

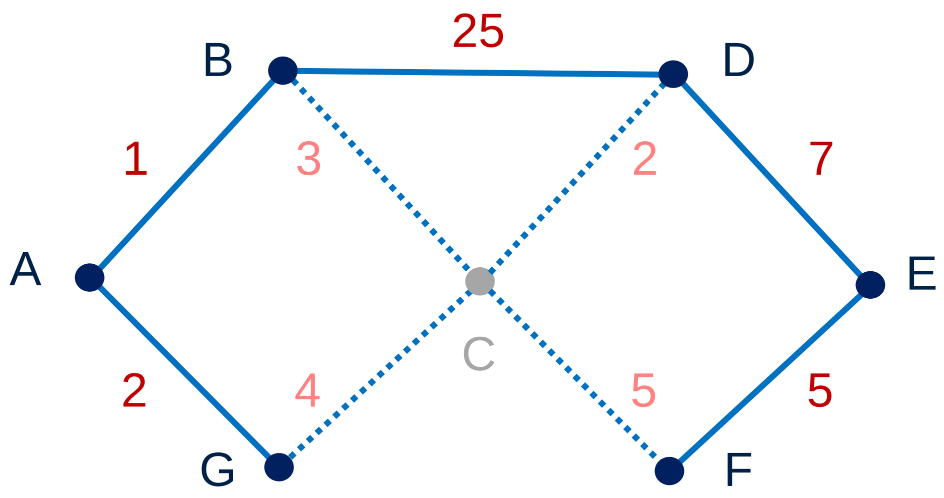
This method has three stages:

(i) Delete a vertex and its incident edges

(ii) Find the MST on the residual network

(iii) LB = $MST_{(Res)}$ + two shortest edges to the deleted vertex

An interesting example



~~UB = 36~~

UB = 29

d) Use the LB algorithm to find a lower bound.

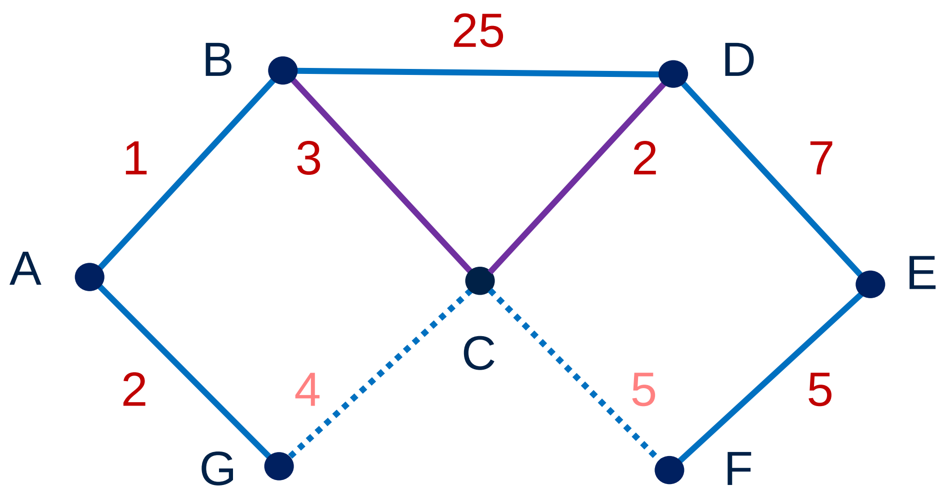
Let us delete vertex C.

The remaining edges form a MST.

AB	1
AG	2
EF	5
DE	7
BD	25

Total length 40

An interesting example



~~UB = 36~~

UB = 29

LB = 45 ???

d) Use the LB algorithm to find a lower bound.

Let us delete vertex C.

The remaining edges form a MST.

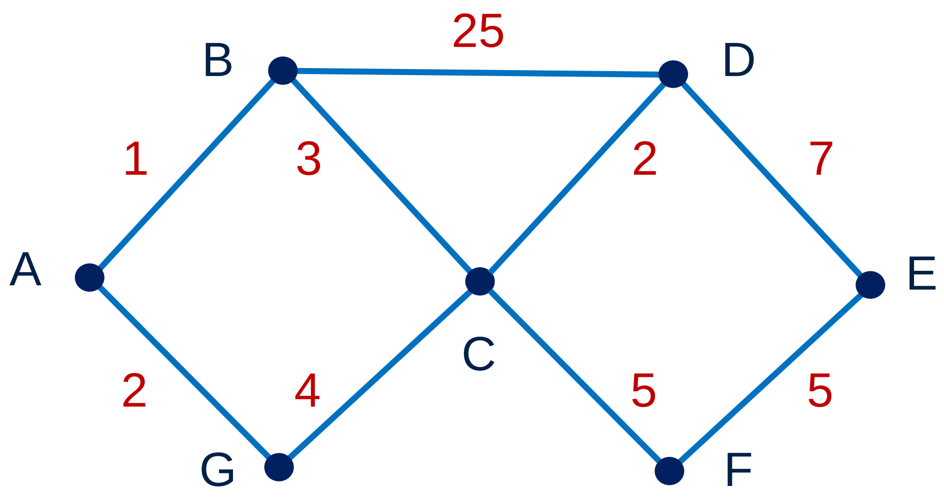
Hence:

$$LB = MST_{(res)} + CD + BC$$

$$LB = 40 + 2 + 3$$

$$LB = 45 \quad ???$$

An interesting example



~~UB = 36~~

UB = 29

LB = 45 ???

e) Express your results in the form of an inequality for T, the length of the optimal tour.

Typically we write an expression of the form:

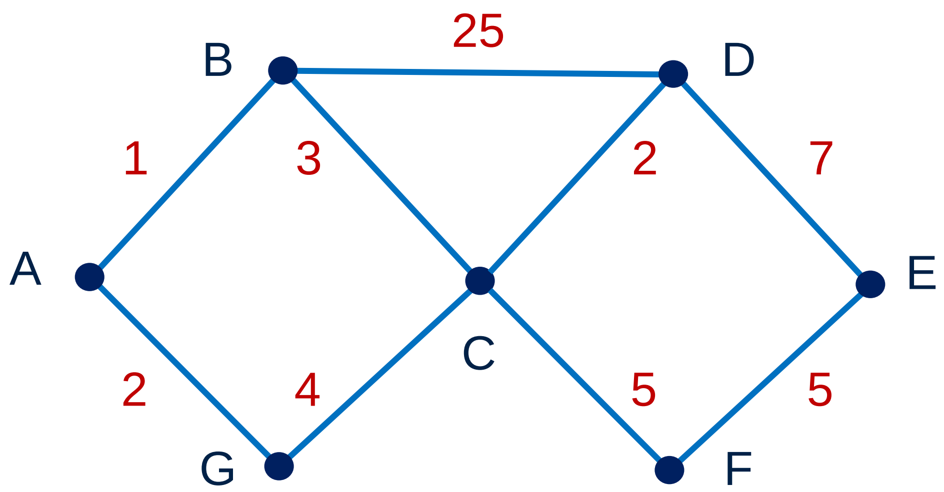
$$LB \leq T \leq UB$$

But it will be nonsense to write:

$$45 \leq T \leq 29$$

Where is the mistake?

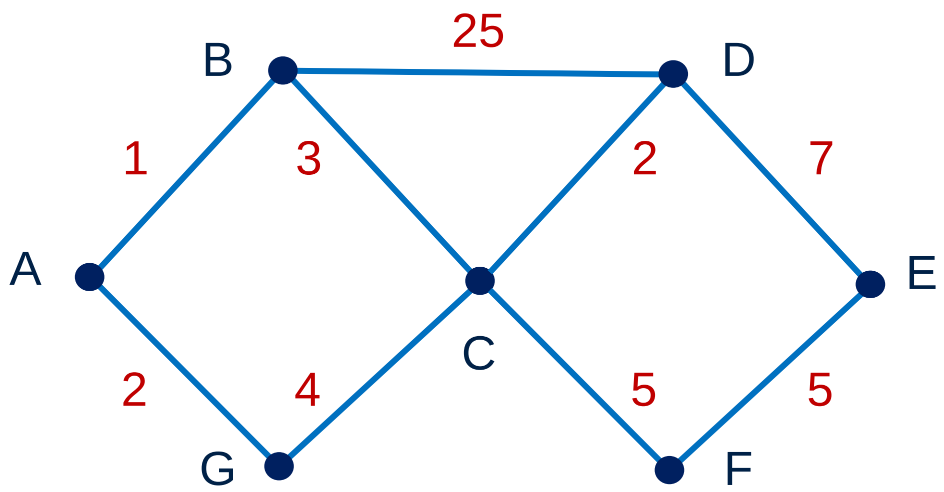
An interesting example



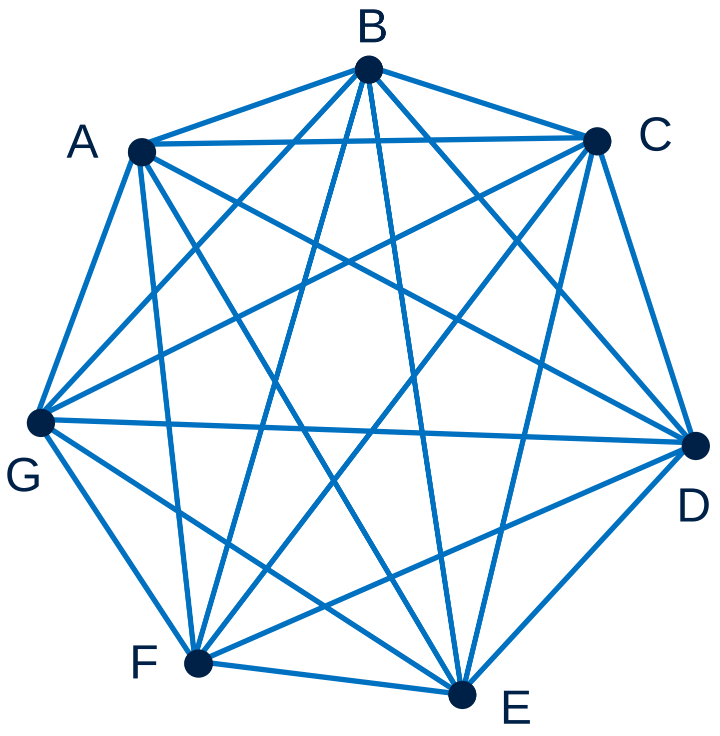
The problem is that this network is not on a compete graph and some of the methods used only work for the Classical problem and not the Practical problem.

This problem should be converted into a Classical problem by creating a table of least distances and operating on that instead.

An interesting example

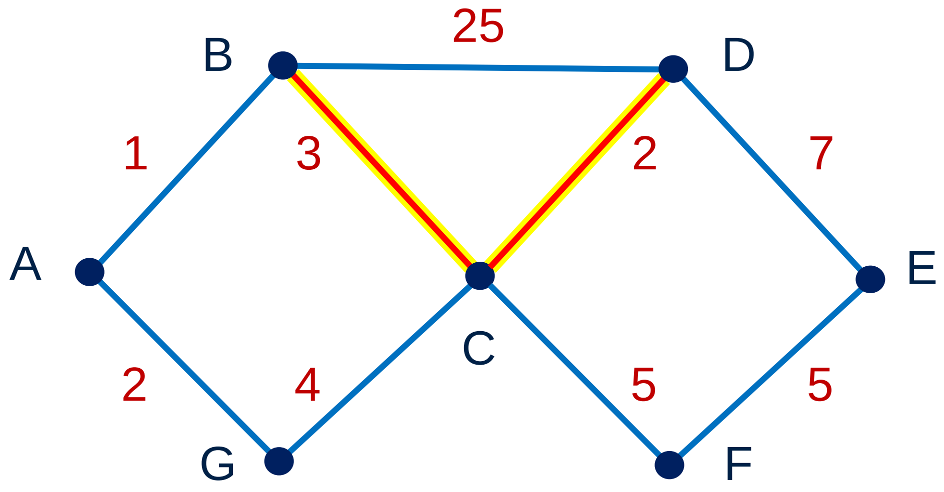


	A	B	C	D	E	F	G
A	-	1					2
B	1	-	3	(25)			
C		3	-	2		5	4
D		(25)	2	-	7		
E				7	-	5	
F			5		5	-	
G	2		4				-

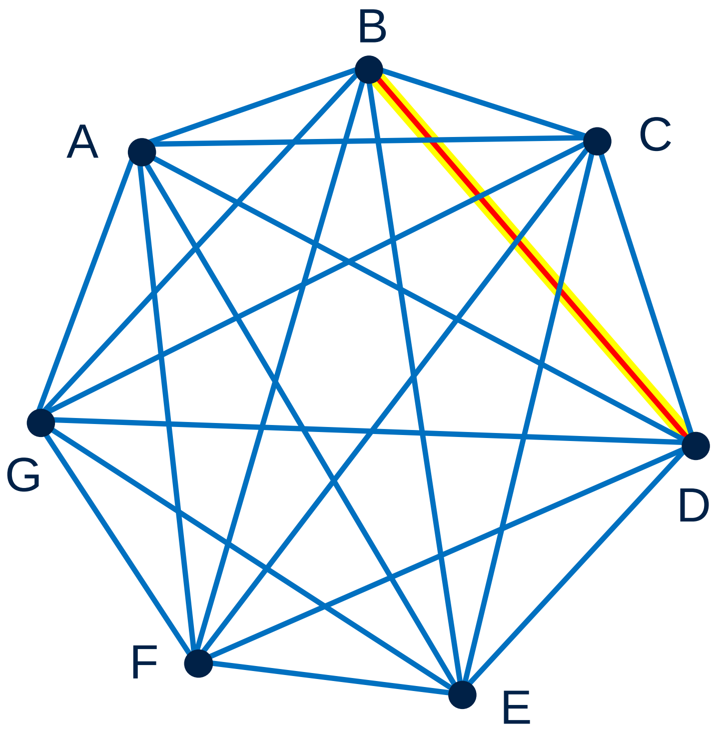


The network is converted into K_n, with each edge representing the shortest distance between that pair of vertices.

An interesting example

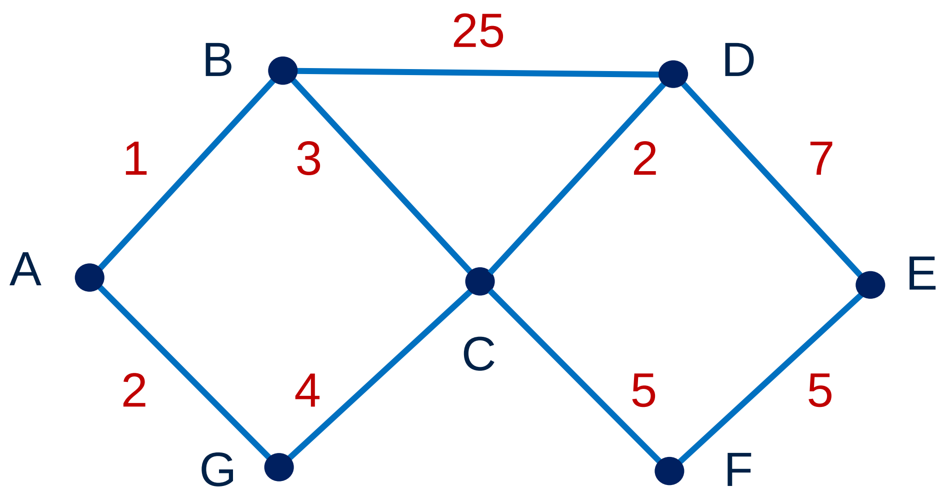


	A	B	C	D	E	F	G
A	-	1					2
B	1	-	3	5			
C		3	-	2		5	4
D		5	2	-	7		
E				7	-	5	
F			5		5	-	
G	2		4				-

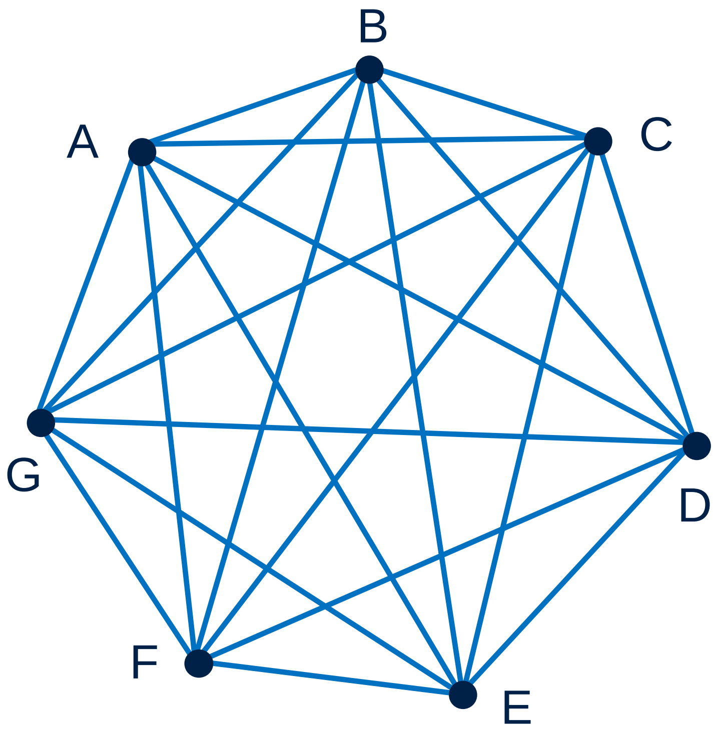


In the process, the troublesome edge BD of length 25 will be replaced by the actual shortest distance between B and D (via C).

An interesting example

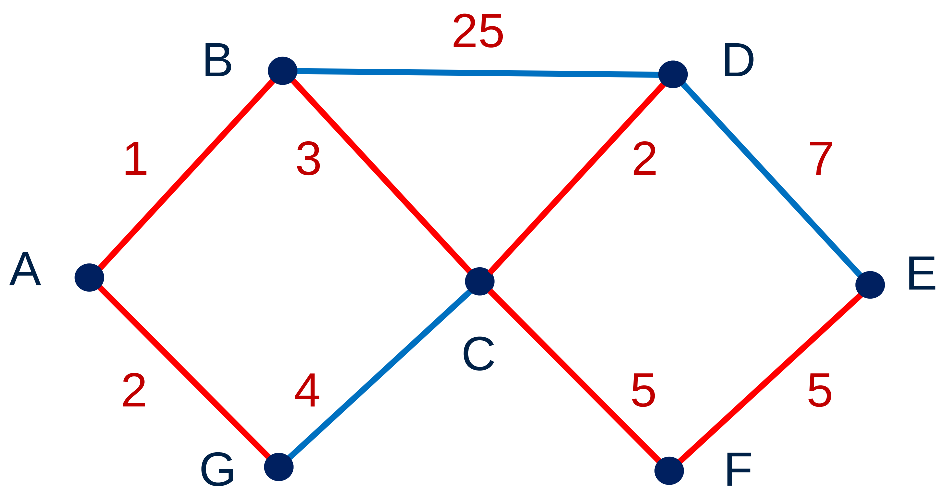


	A	B	C	D	E	F	G
A	-	1	4	6	13	9	2
B	1	-	3	5	12	8	3
C	4	3	-	2	9	5	4
D	6	5	2	-	7	7	6
E	13	12	9	7	-	5	13
F	9	8	5	7	5	-	9
G	2	3	4	6	13	9	-

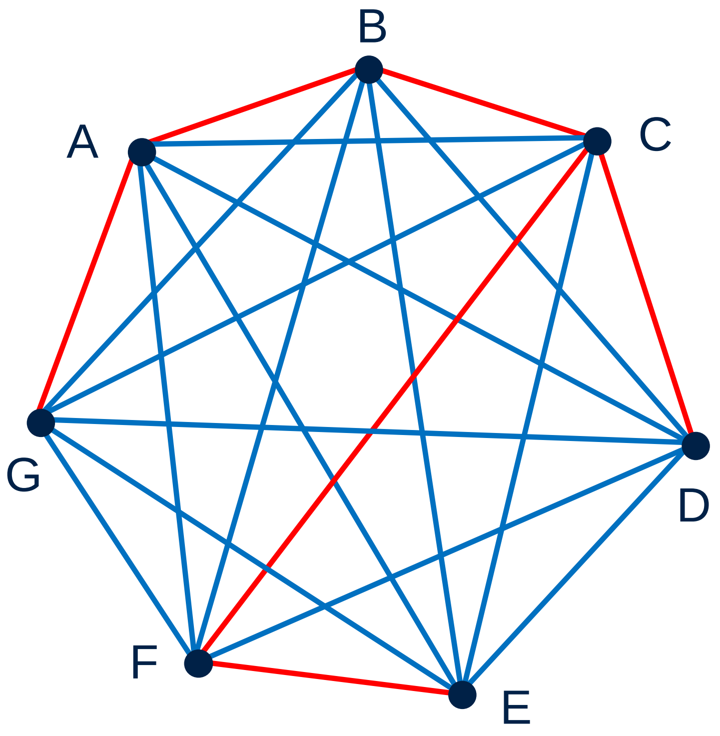


a) Find the minimum spanning tree for this network.

An interesting example



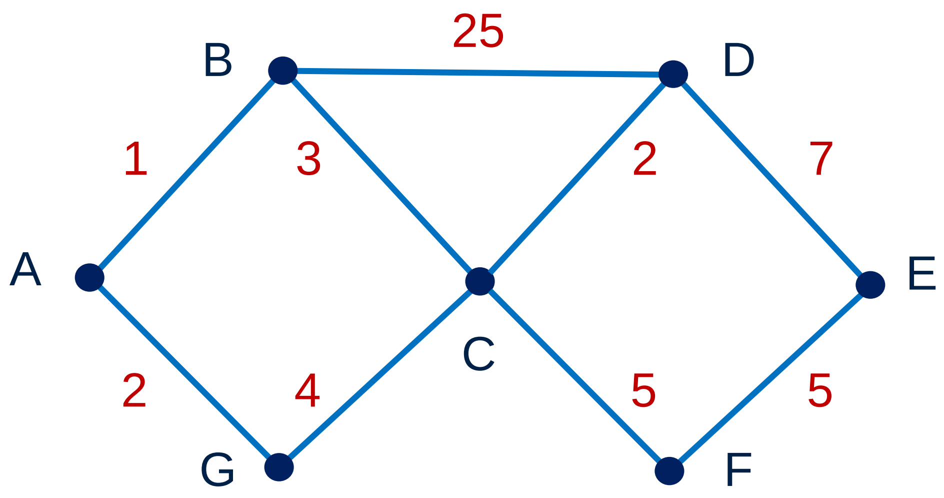
	A	B	C	D	E	F	G
A	-	1	4	6	13	9	2
B	1	-	3	5	12	8	3
C	4	3	-	2	9	5	4
D	6	5	2	-	7	7	6
E	13	12	9	7	-	5	13
F	9	8	5	7	5	-	9
G	2	3	4	6	13	9	-



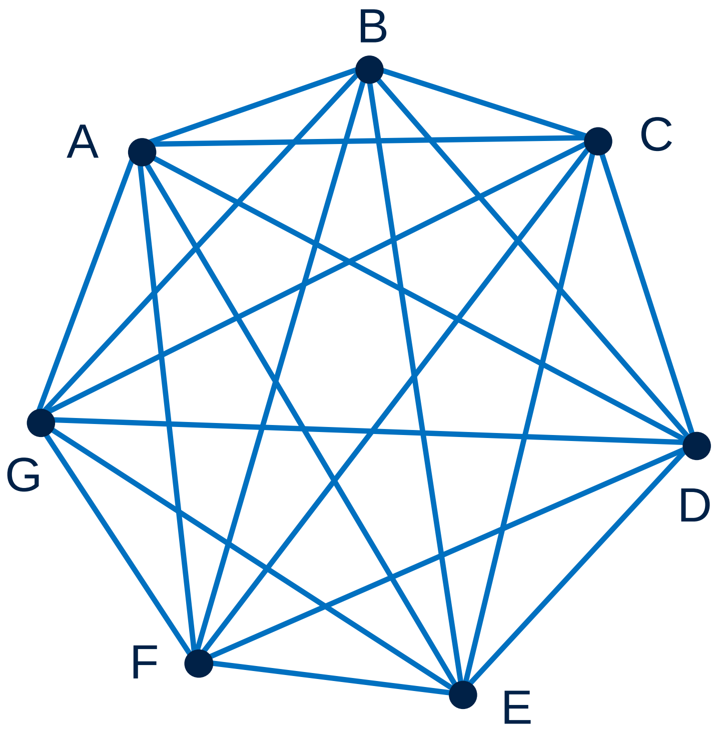
a) Find the minimum spanning tree for this network.

Edges chosen: AB, AG, CD, BC, CF, EF Total length 18 (as before)

An interesting example



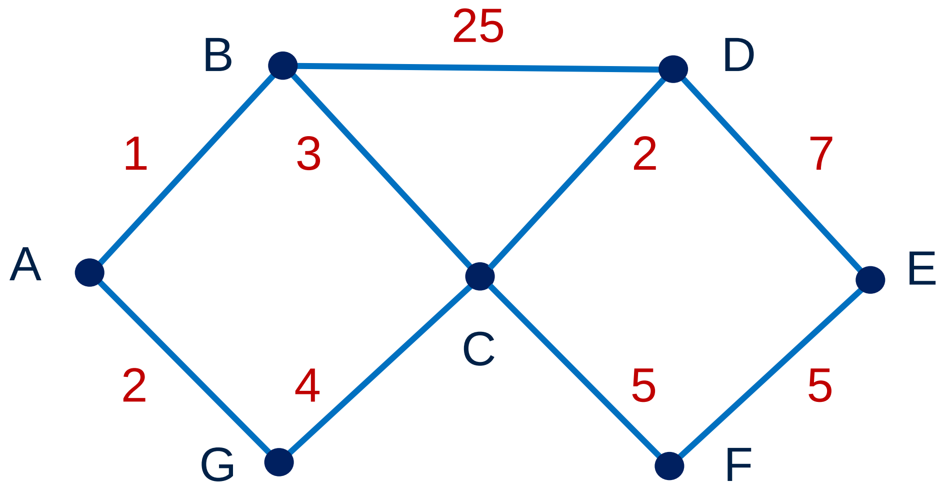
	A	B	C	D	E	F	G
A	-	1	4	6	13	9	2
B	1	-	3	5	12	8	3
C	4	3	-	2	9	5	4
D	6	5	2	-	7	7	6
E	13	12	9	7	-	5	13
F	9	8	5	7	5	-	9
G	2	3	4	6	13	9	-



b) Hence state an upper bound for the travelling salesperson problem.

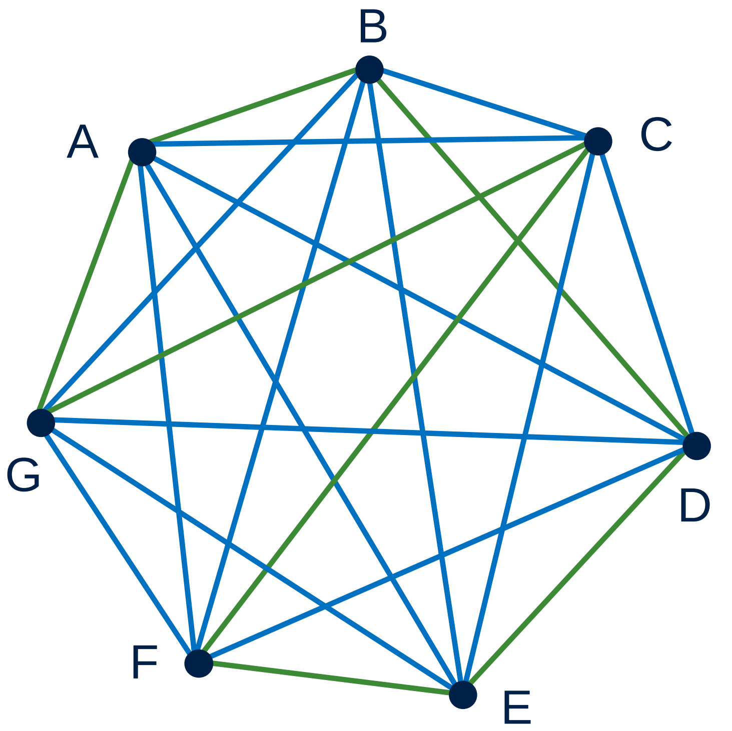
UB = 2 x MST = 2 x 18 = 36 (as before)

An interesting example



UB = ~~36~~ 29

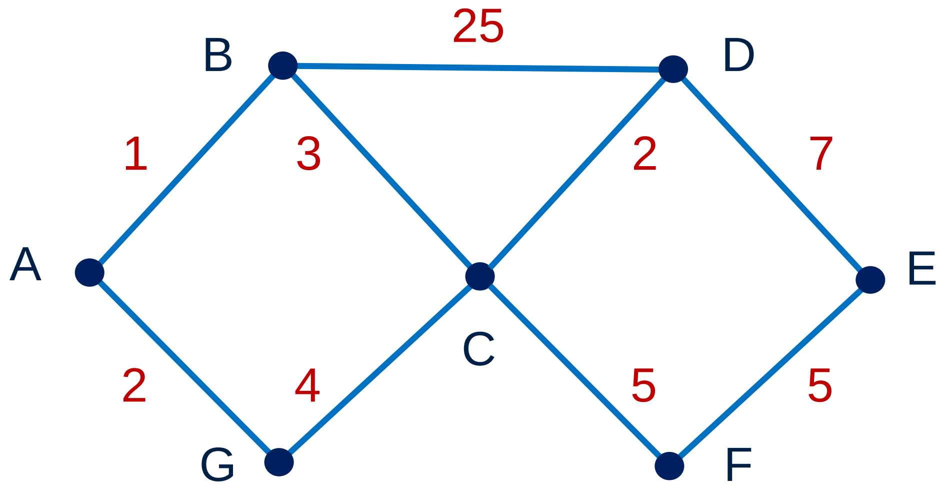
	A	B	C	D	E	F	G
A	-	1	4	6	13	9	2
B	1	-	3	5	12	8	3
C	4	3	-	2	9	5	4
D	6	5	2	-	7	7	6
E	13	12	9	7	-	5	13
F	9	8	5	7	5	-	9
G	2	3	4	6	13	9	-



c) Identify a tour which gives an improved upper bound.

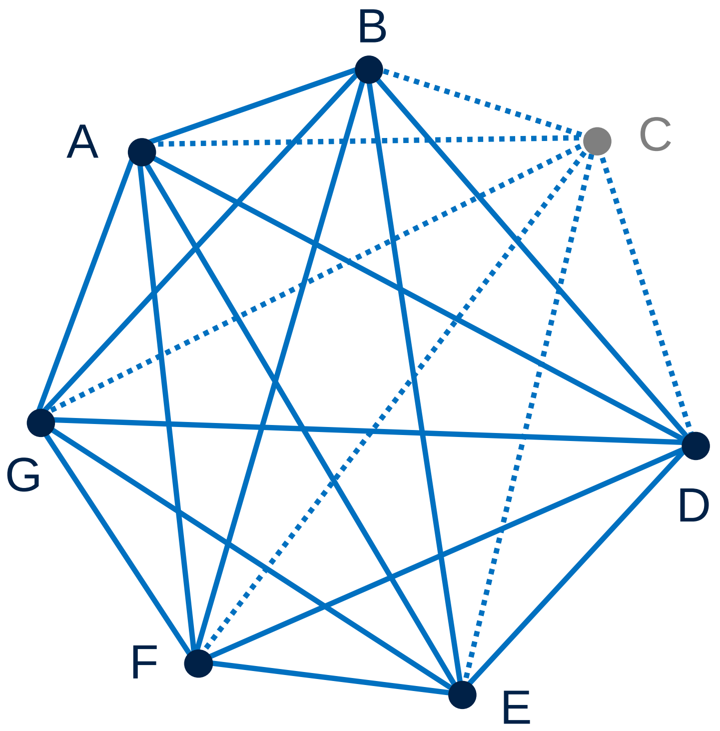
A-B-D-E-F-C-G-(A) now has total length 29. UB = 29 (as before)

An interesting example



UB = ~~36~~ 29

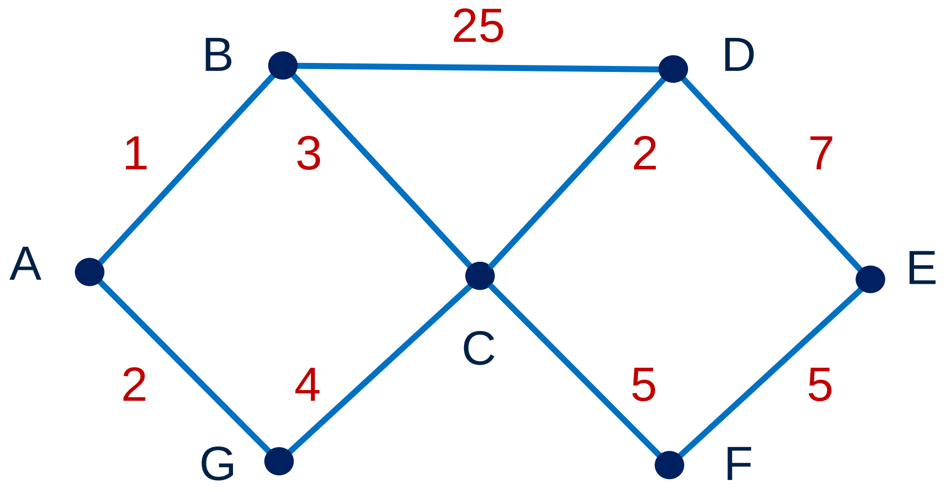
	A	B	C	D	E	F	G
A	-	1	4	6	13	9	2
B	1	-	3	5	12	8	3
C	4	3	-	2	9	5	4
D	6	5	2	-	7	7	6
E	13	12	9	7	-	5	13
F	9	8	5	7	5	-	9
G	2	3	4	6	13	9	-



d) Use the Lower Bound algorithm to find a lower bound.

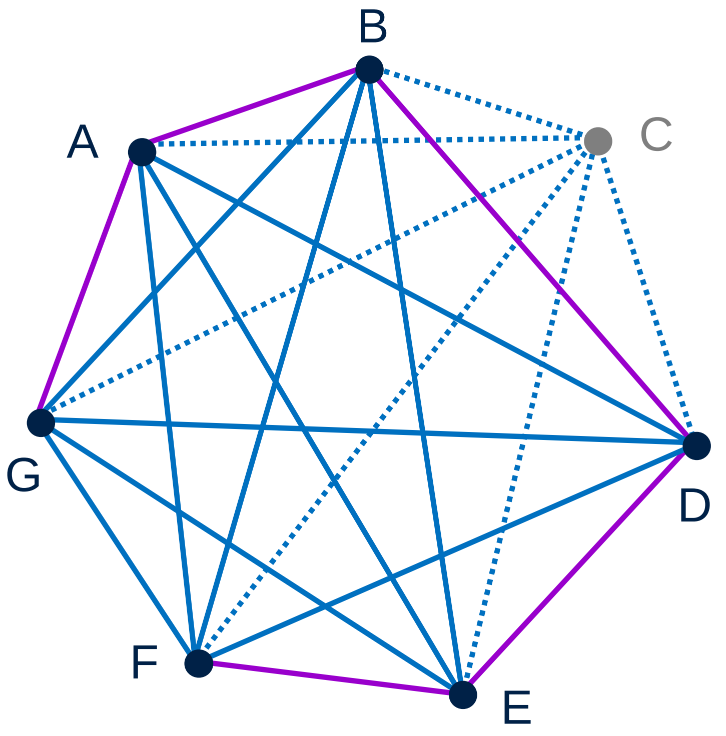
Let us delete vertex C.

An interesting example



UB = ~~36~~ 29

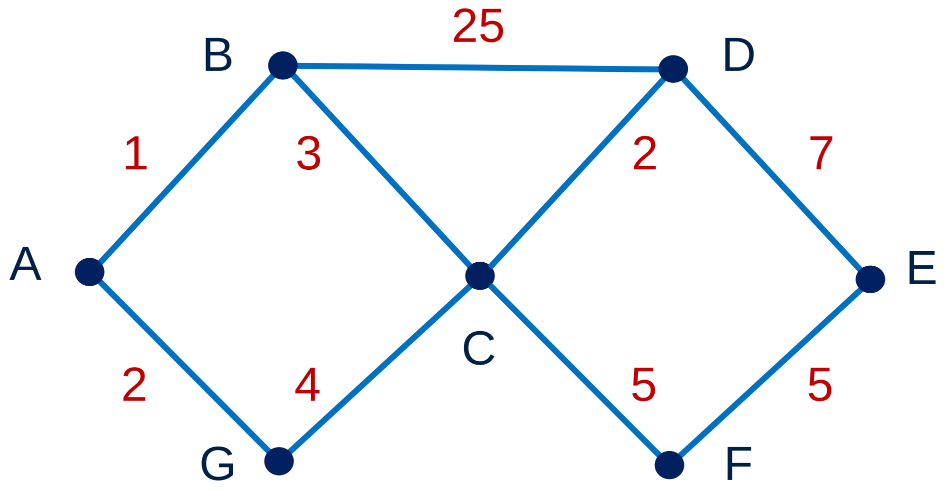
	A	B	C	D	E	F	G
A	-	1	4	6	13	9	2
B	1	-	3	5	12	8	3
C	4	3	-	2	9	5	4
D	6	5	2	-	7	7	6
E	13	12	9	7	-	5	13
F	9	8	5	7	5	-	9
G	2	3	4	6	13	9	-



d) Use the Lower Bound algorithm to find a lower bound.

$\text{MST}_{(\text{Res})}$ uses the following edges: AB, AG, BD, EF, DE. Total length 20

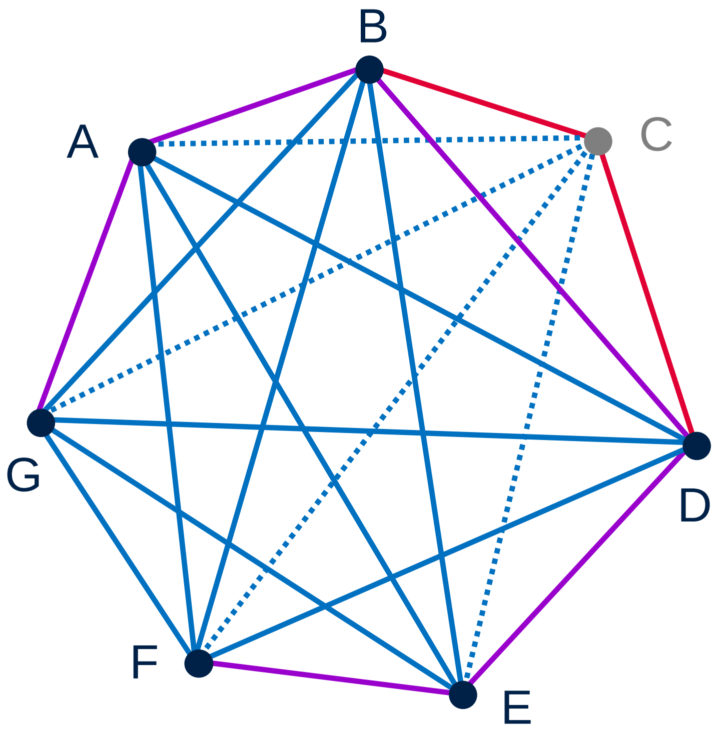
An interesting example



UB = ~~36~~ 29

LB = 25

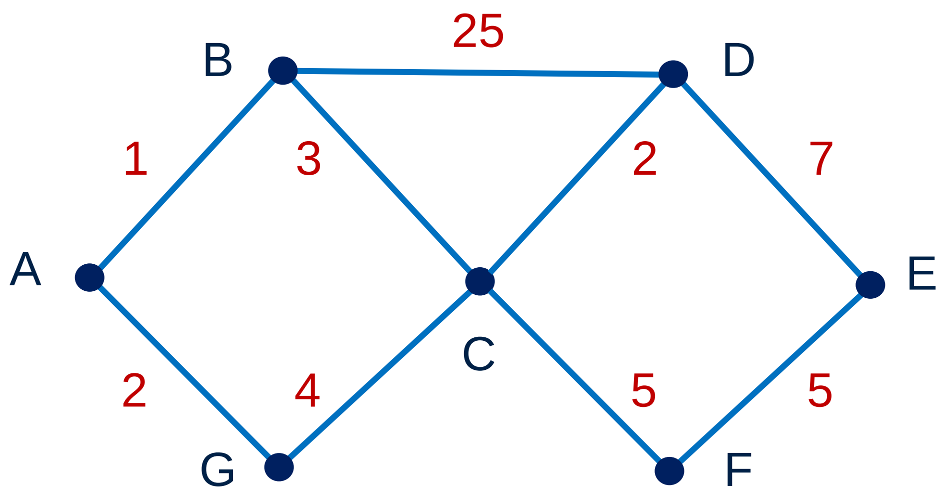
	A	B	C	D	E	F	G
A	-	1	4	6	13	9	2
B	1	-	3	5	12	8	3
C	4	3	-	2	9	5	4
D	6	5	2	-	7	7	6
E	13	12	9	7	-	5	13
F	9	8	5	7	5	-	9
G	2	3	4	6	13	9	-



d) Use the Lower Bound algorithm to find a lower bound.

$LB = MST_{(Res)} + BC + CD = 20 + 2 + 3 = 25$ LB = 25

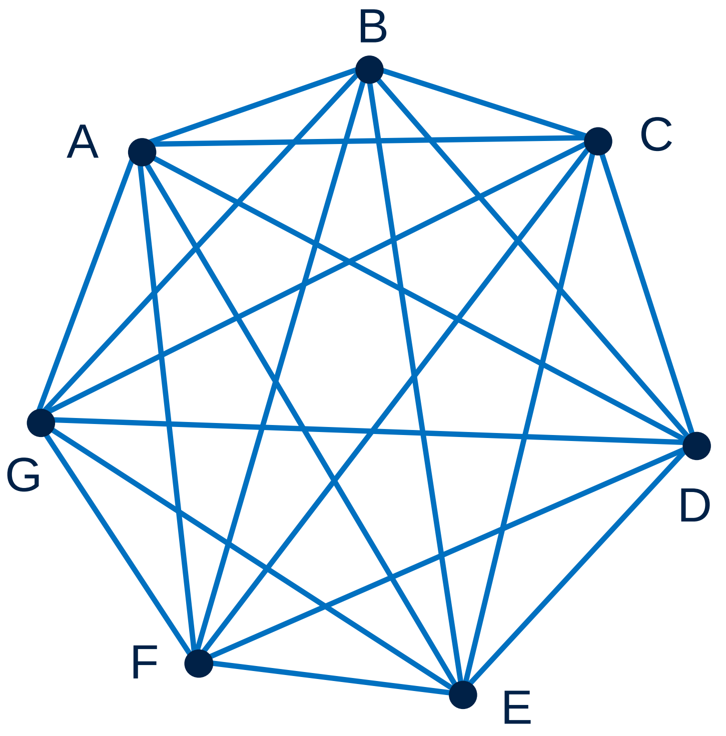
An interesting example



UB = ~~36~~ 29

LB = 25

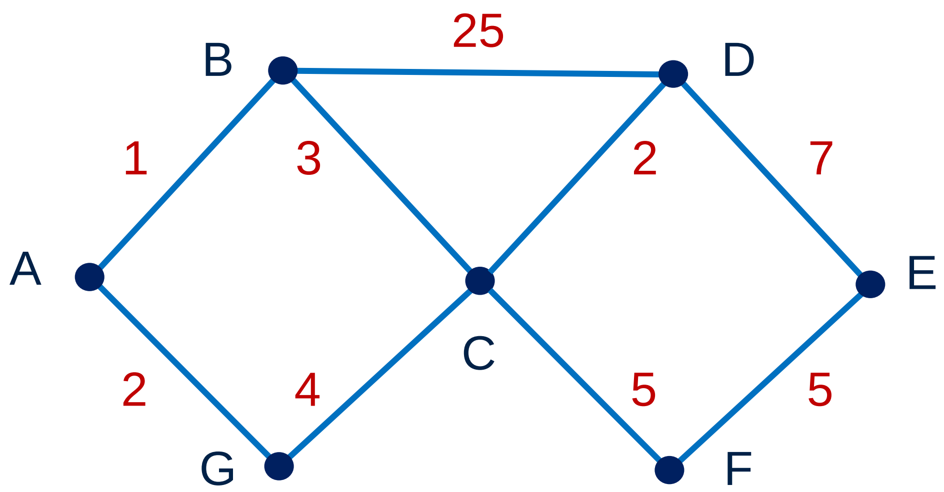
	A	B	C	D	E	F	G
A	-	1	4	6	13	9	2
B	1	-	3	5	12	8	3
C	4	3	-	2	9	5	4
D	6	5	2	-	7	7	6
E	13	12	9	7	-	5	13
F	9	8	5	7	5	-	9
G	2	3	4	6	13	9	-



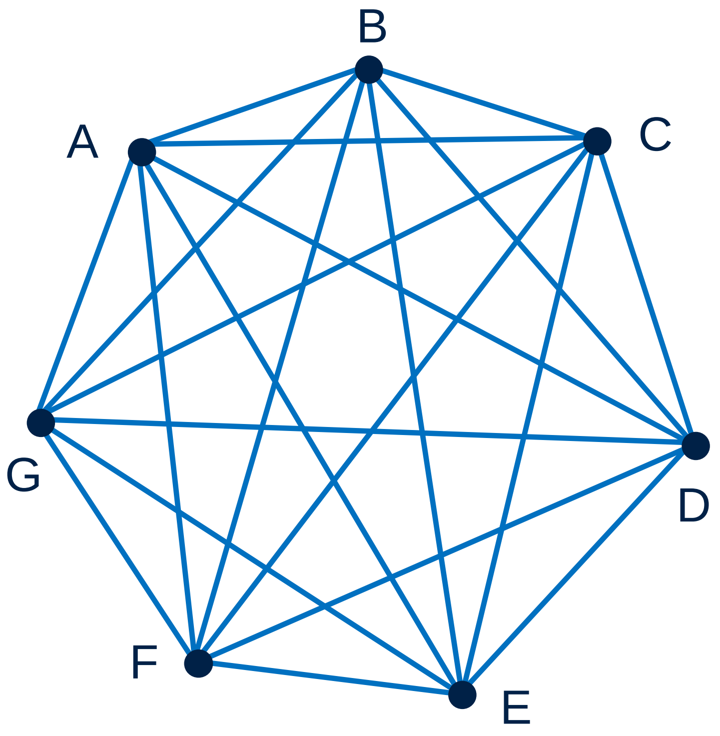
e) Express your results in the form of an inequality for T, the length of the optimal tour.

Finally we can write: $25 \le T \le 29$

An interesting example



	A	B	C	D	E	F	G
A	-	1	4	6	13	9	2
B	1	-	3	5	12	8	3
C	4	3	-	2	9	5	4
D	6	5	2	-	7	7	6
E	13	12	9	7	-	5	13
F	9	8	5	7	5	-	9
G	2	3	4	6	13	9	-



Practical problem **Table of least distances** **Classical problem**

The Practical problem **must** be converted into the Classical problem in order to apply the **lower bound algorithm**. The same applies to the **Nearest Neighbour algorithm**.

What is the point of a Lower Bound that is clearly not a tour?

Using Minimum Spanning Trees

For a network on *n* vertices:

- A cycle has n edges
- A minimum spanning tree has n-1 edges

Upper Bound methods

- Any tour — n edges
- 2 x MST — $2(n-1)$ edges

For a network with $n > 2$: $2(n-1) > n$

$\Rightarrow$ 2 x MST is usually an overly large Upper Bound.

Using Minimum Spanning Trees

For a network on *n* vertices:

- A cycle has *n* edges
- A minimum spanning tree has *n*-1 edges

Upper Bound methods

• Any tour	*n* edges
• 2 x MST	2(*n*-1) edges
• Nearest Neighbour	*n* edges

Using Minimum Spanning Trees

The Lower Bound Algorithm:

(i) Delete a vertex (V) and its incident edges

(ii) Find the MST on the residual network

(iii) LB = $MST_{(res)}$ + two shortest edges to V

Using Minimum Spanning Trees

The Lower Bound Algorithm:

(i) Delete a vertex (V) and its incident edges — n-1 vertices

(ii) Find the MST on the residual network — n-2 edges

(iii) LB = $\text{MST}_{(res)}$ + two shortest edges to V — n-2 + 2 edges

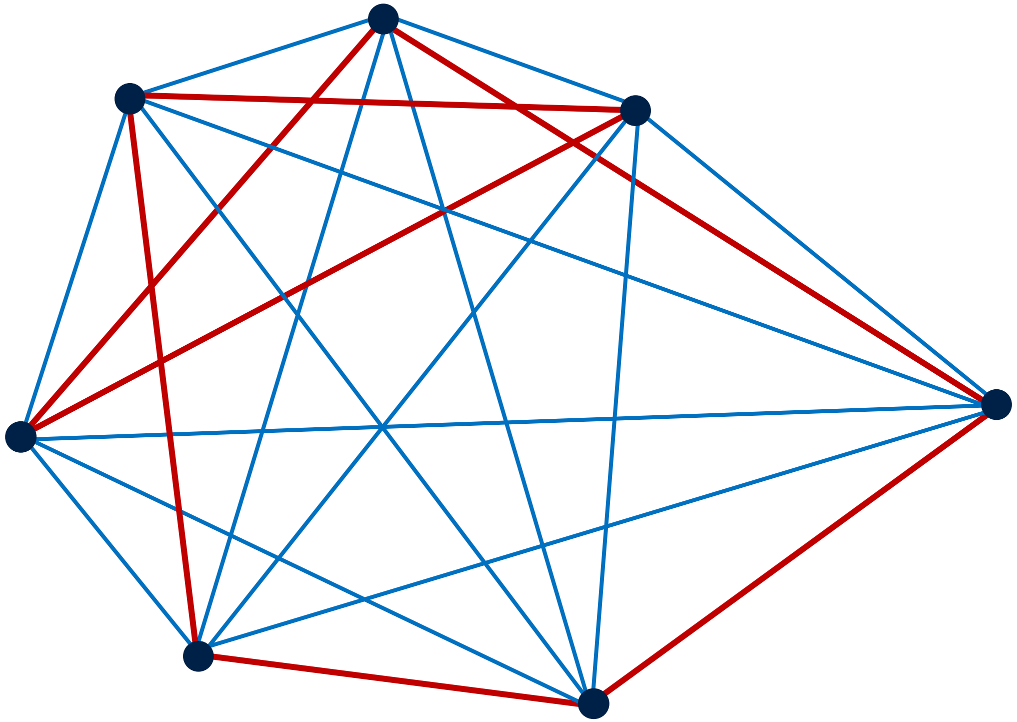
The algorithm delivers the right number of edges.

But how do we know it is a Lower Bound when it's not a tour?

Justifying the Lower Bound algorithm

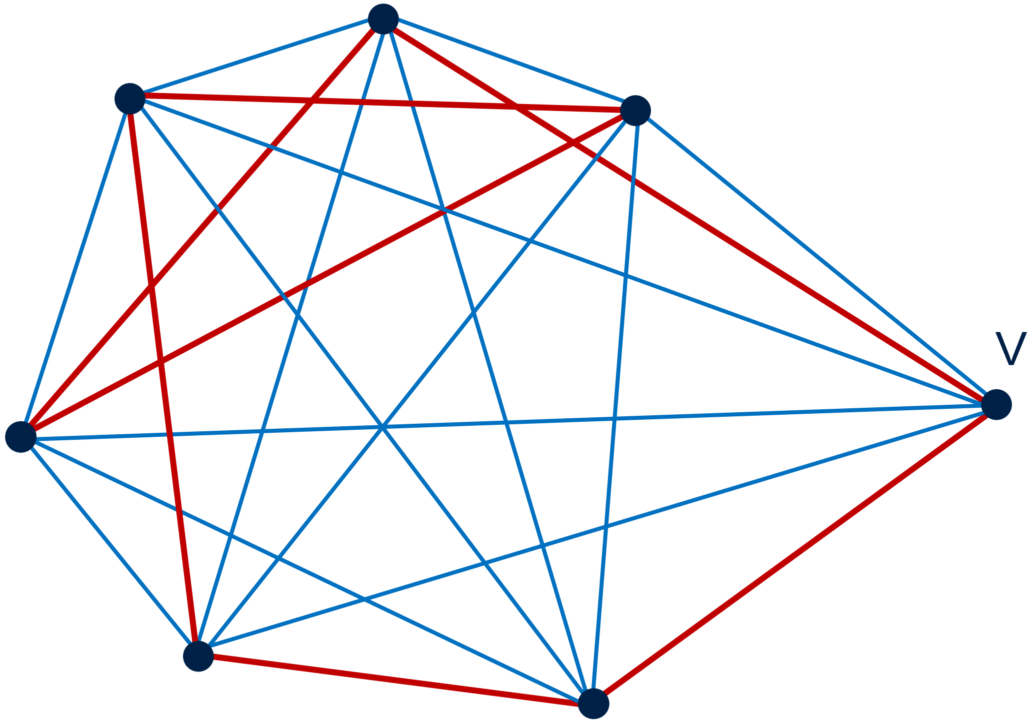
Suppose that for a particular network there is an optimal tour, T.

Let the red edges represent the optimal tour, T.



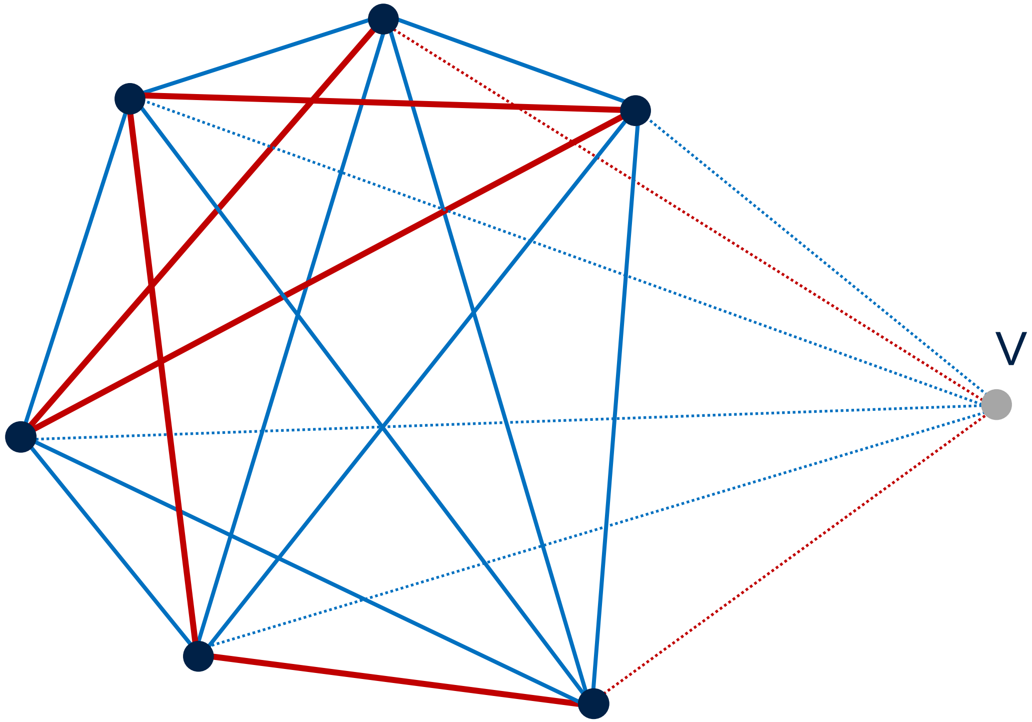
Justifying the Lower Bound algorithm

Consider in turn each stage of the Lower Bound algorithm, where vertex V is to be deleted.



Justifying the Lower Bound algorithm

Delete vertex V and its incident edges.

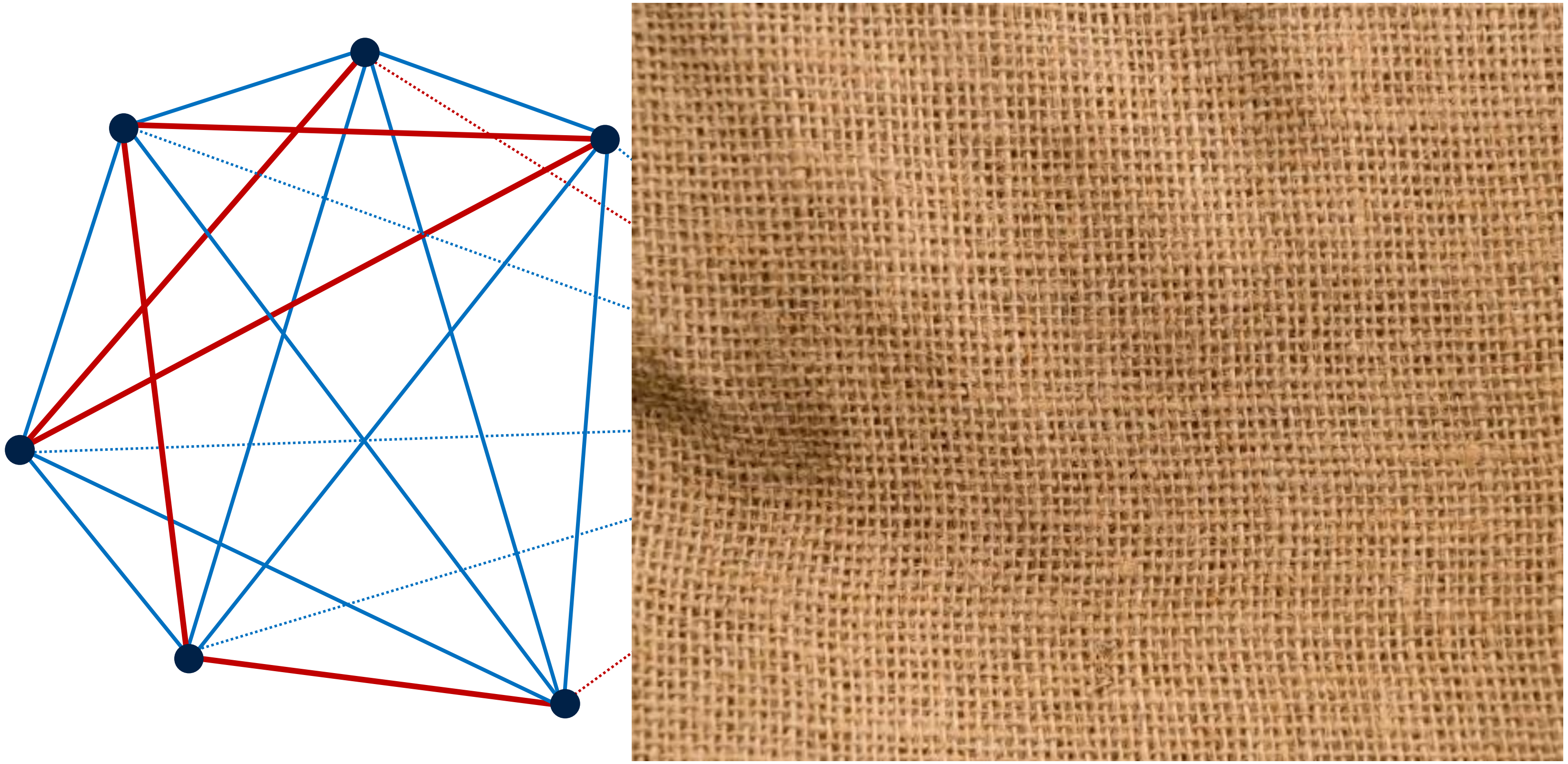


Justifying the Lower Bound algorithm

Find the MST on the residual network.

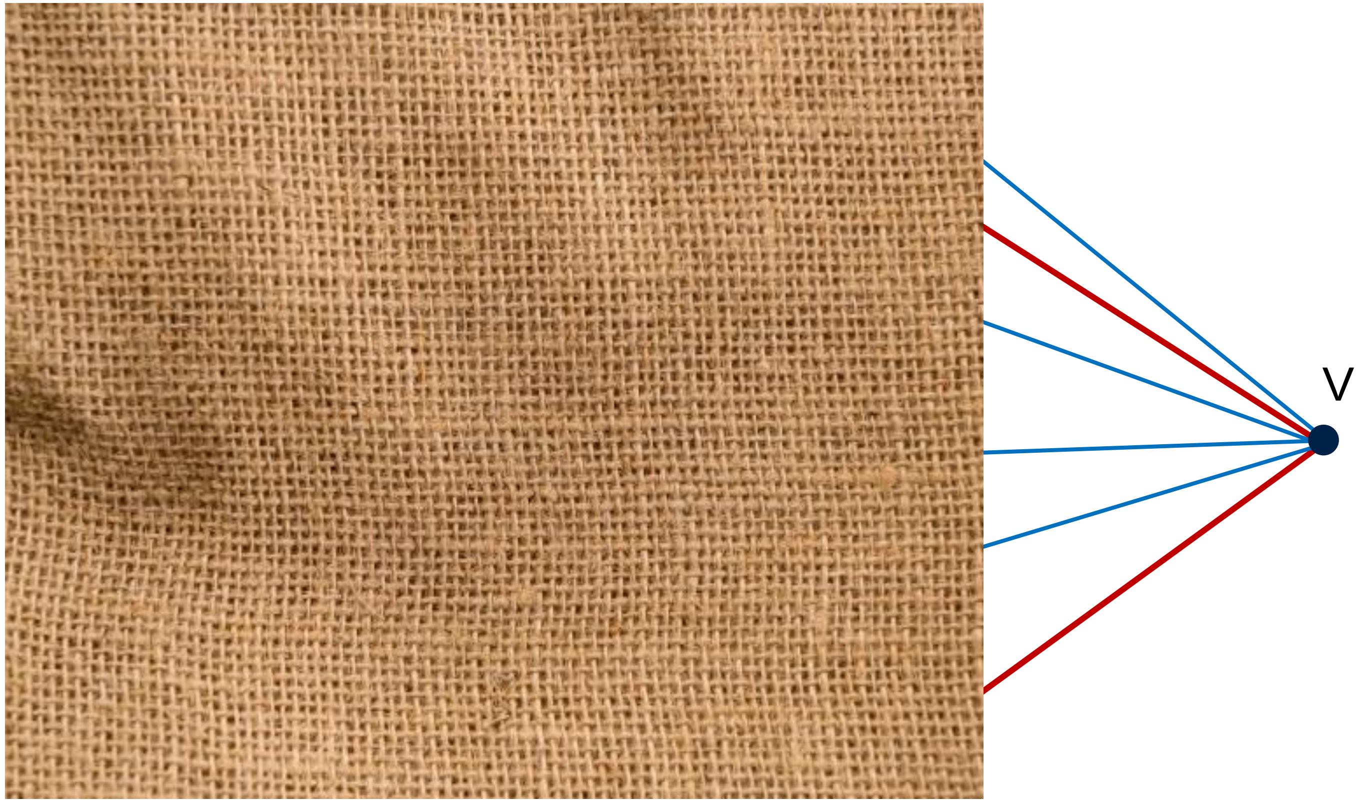
The *n*-2 edges of T that remain must be a spanning tree on the *n*-1 vertices.

The MST on these *n*-1 vertices will be less than or equal in length to this spanning tree.



Justifying the Lower Bound algorithm

LB = $MST_{(Res)}$ + two shortest edges to V



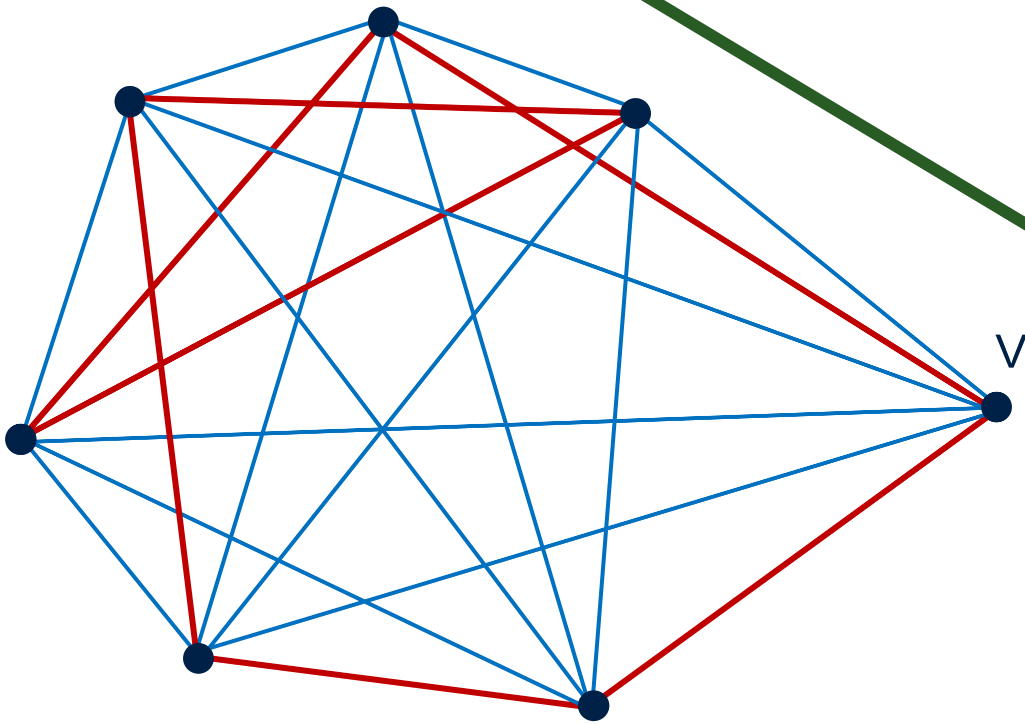
The edges used in T may not be the two shortest edges from V.

The sum of the two shortest edges from V will be less than or equal to the sum of those two in T.

Justifying the Lower Bound algorithm

$LB = MST_{(Res)}$ + two shortest edges to V $\leq T$

The MST on these *n*-1 vertices will be less than or equal in length to the sum of the *n*-2 edges of T.



The sum of the two shortest edges from V will be less than or equal to the sum of those two in T.

Should I use ≤ or < ?

Expressing the result using inequalities

Once we have established our best Upper and Lower Bounds, we know the value of T lies between them.

But …

LB ≤ T ≤ UB ?

< <

? ?

Expressing the result using inequalities

Once we have established our best Upper and Lower Bounds, we know the value of T lies between them.

But … $LB \le T \le UB$?

$<$ $<$

? ?

All three UB methods generate tours.
The smallest of these *might* actually be the optimal tour, T.

Expressing the result using inequalities

Once we have established our best Upper and Lower Bounds, we know the value of T lies between them.

But … $LB \leq T \leq UB$?

✓

$<$

?

Expressing the result using inequalities

Once we have established our best Upper and Lower Bounds, we know the value of T lies between them.

But … $LB \leq T \leq UB$?

✓

$<$

?

Some past paper mark schemes have insisted that $<$ be used, possibly because the LB found was not a tour.

An informative example

Consider the following network problem:

a) Find an Upper Bound using the Nearest Neighbour algorithm starting from D.

b) Find a Lower Bound by deleting D.

	A	B	C	D
A	-	9	9	6
B	9	-	8	7
C	9	8	-	10
D	6	7	10	-

An informative example

Solution:

a) Find an Upper Bound using the Nearest Neighbour algorithm starting from D.

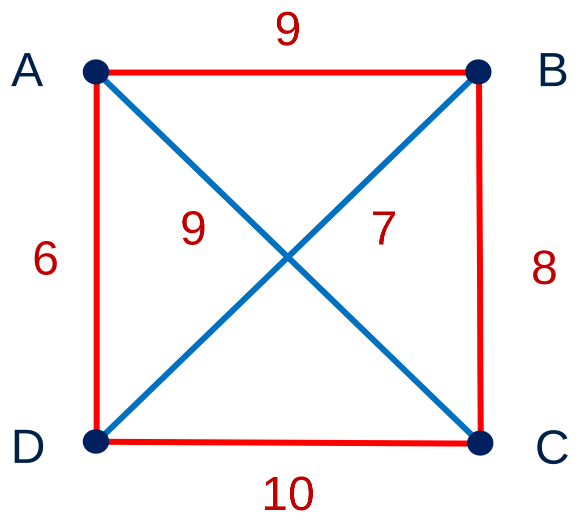
From D:

D – A – B – C – (A)

6 + 9 + 8 + 10 = 33

UB = 33

	A	B	C	D
A	-	9	9	6
B	9	-	8	7
C	9	8	-	10
D	6	7	10	-

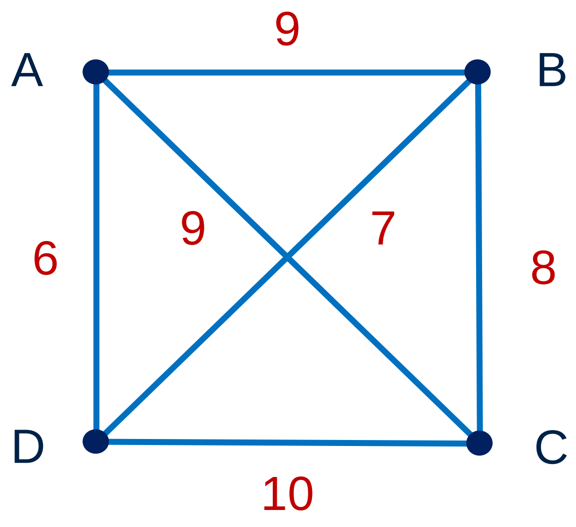


An informative example

Solution:

b) Find a Lower Bound by deleting D.

	A	B	C	D
A	-	9	9	6
B	9	-	8	7
C	9	8	-	10
D	6	7	10	-



An informative example

Solution:

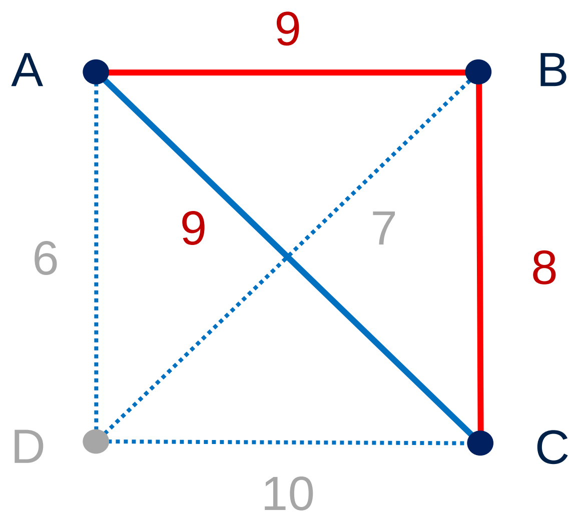
b) Find a Lower Bound by deleting D.

Delete D.

$MST_{(Res)}$ = BC + AB

8 + 9 = 17

	A	B	C	D
A	-	9	9	6
B	9	-	8	7
C	9	8	-	10
D	6	7	10	-



An informative example

Solution:

b) Find a Lower Bound by deleting D.

Delete D.

$MST_{(Res)} = BC + AB$

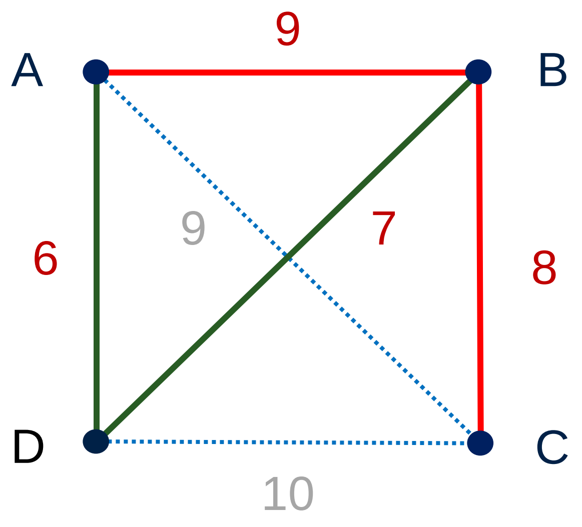
$8 + 9 = 17$

$LB = MST_{(Res)} + AD + BD$

$17 + 6 + 7 = 30$

LB = 30

	A	B	C	D
A	-	9	9	6
B	9	-	8	7
C	9	8	-	10
D	6	7	10	-



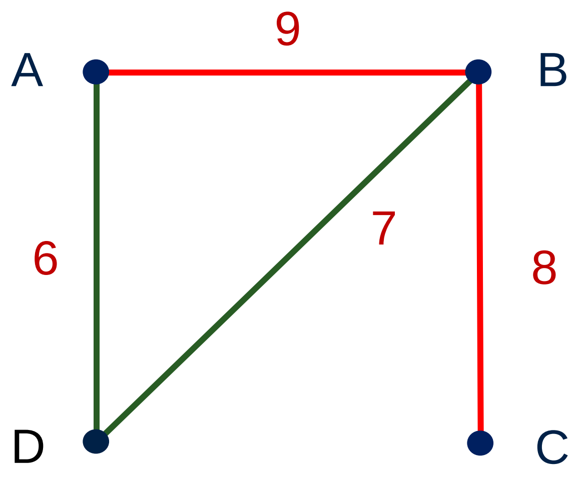
An informative example

This Lower Bound is clearly not a tour.

Should we therefore write:

$$30 < T \leq 33 \quad ?$$

	A	B	C	D
A	-	9	9	6
B	9	-	8	7
C	9	8	-	10
D	6	7	10	-



An informative example

This Lower Bound is clearly not a tour.

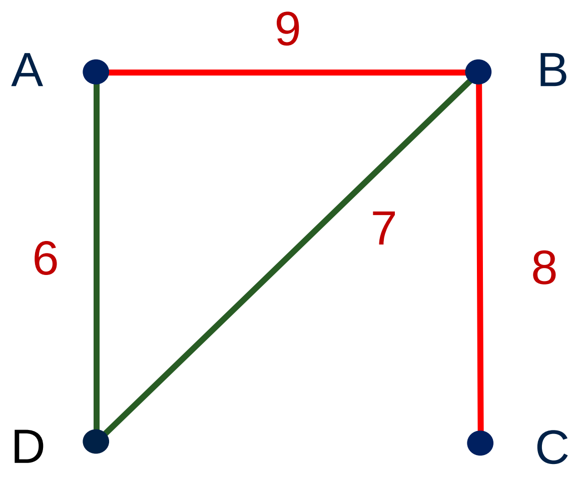
Should we therefore write:

$$30 < T \leq 33 \quad ?$$

This rules out the possibility that T = 30.

NO! We have not proved that there is <u>no</u> LB of 30 which *is* a tour!

	A	B	C	D
A	-	9	9	6
B	9	-	8	7
C	9	8	-	10
D	6	7	10	-

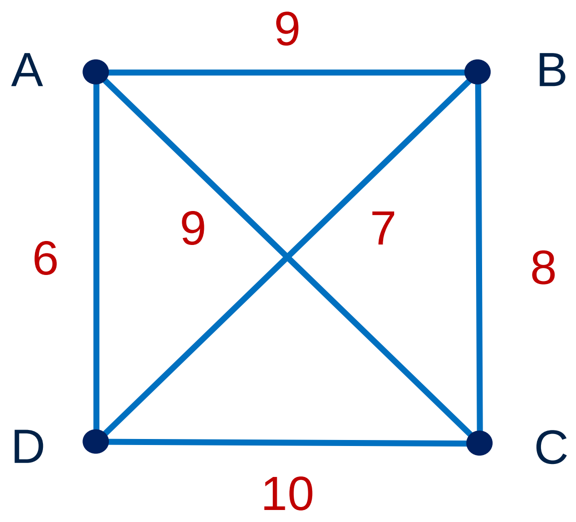


An informative example

Suppose the question had said:

b) Find a Lower Bound by deleting B.

	A	B	C	D
A	-	9	9	6
B	9	-	8	7
C	9	8	-	10
D	6	7	10	-



An informative example

Suppose the question had said:

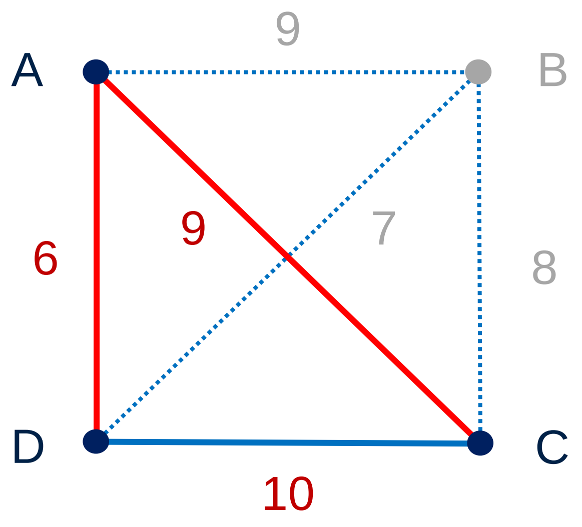
b) Find a Lower Bound by deleting B.

Delete B.

$MST_{(Res)} = AD + AC$

$6 + 9 = 15$

	A	B	C	D
A	-	9	9	6
B	9	-	8	7
C	9	8	-	10
D	6	7	10	-



An informative example

Suppose the question had said:

b) Find a Lower Bound by deleting B.

Delete B.

$MST_{(Res)} = AD + AC$

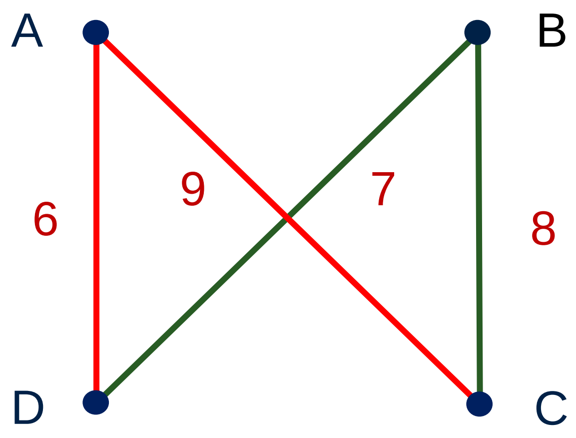
$6 + 9 = 15$

$LB = MST_{(Res)} + BD + BC$

$15 + 7 + 8 = 30$

LB = 30 *and* it's a cycle!

	A	B	C	D
A	-	9	9	6
B	9	-	8	7
C	9	8	-	10
D	6	7	10	-



An informative example

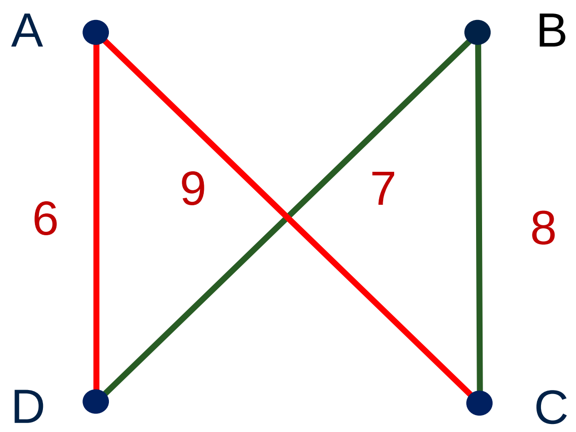
So we have:

LB = 30 *and* a cycle ⇒ UB = 30

⇒ $30 \leq T \leq 30$

⇒ $T = 30$

	A	B	C	D
A	-	9	9	6
B	9	-	8	7
C	9	8	-	10
D	6	7	10	-



Summary - Modelling

Travelling Salesperson:

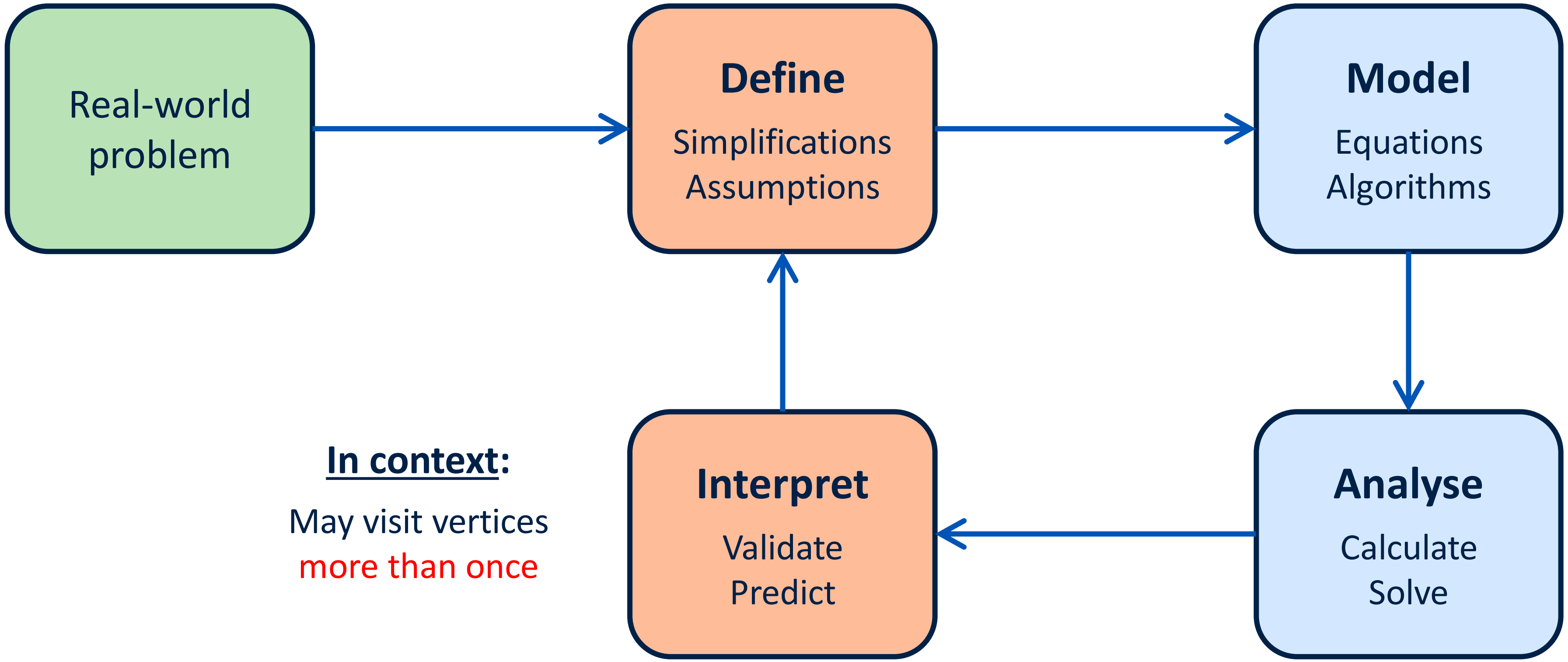
Has factorial order, O(n!)
P versus NP

Practical Problem:

Optimal tour on an
incomplete graph

Classical Problem:

Optimal H-cycle on a
complete graph, K_n



Methods:

Use of MST
Nearest Neighbour
& LB algorithms

In context:

May visit vertices
more than once

Solution:

Write in the form
LB ≤ T ≤ UB
(in general …)