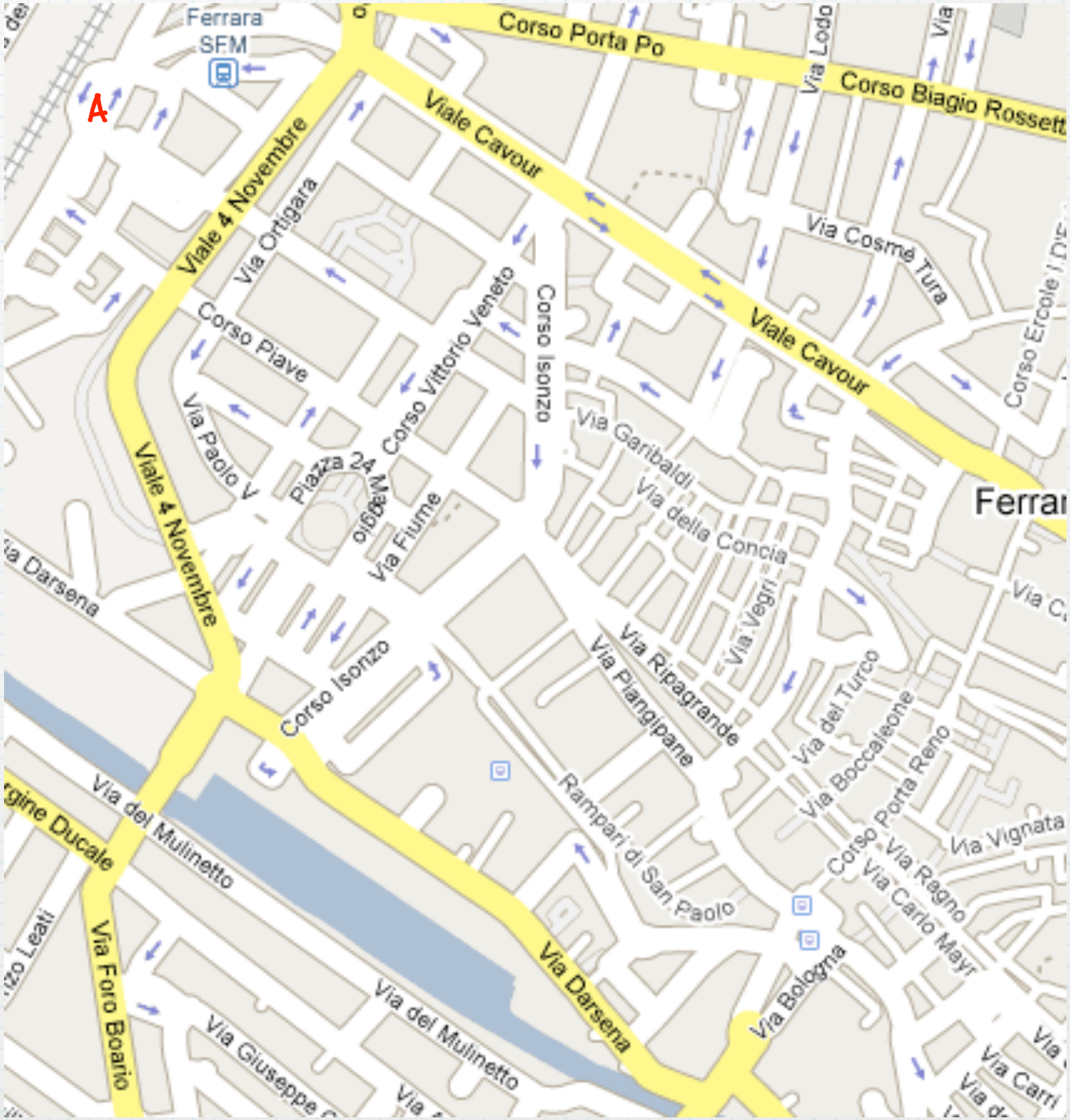
Graphs and network flows

Example

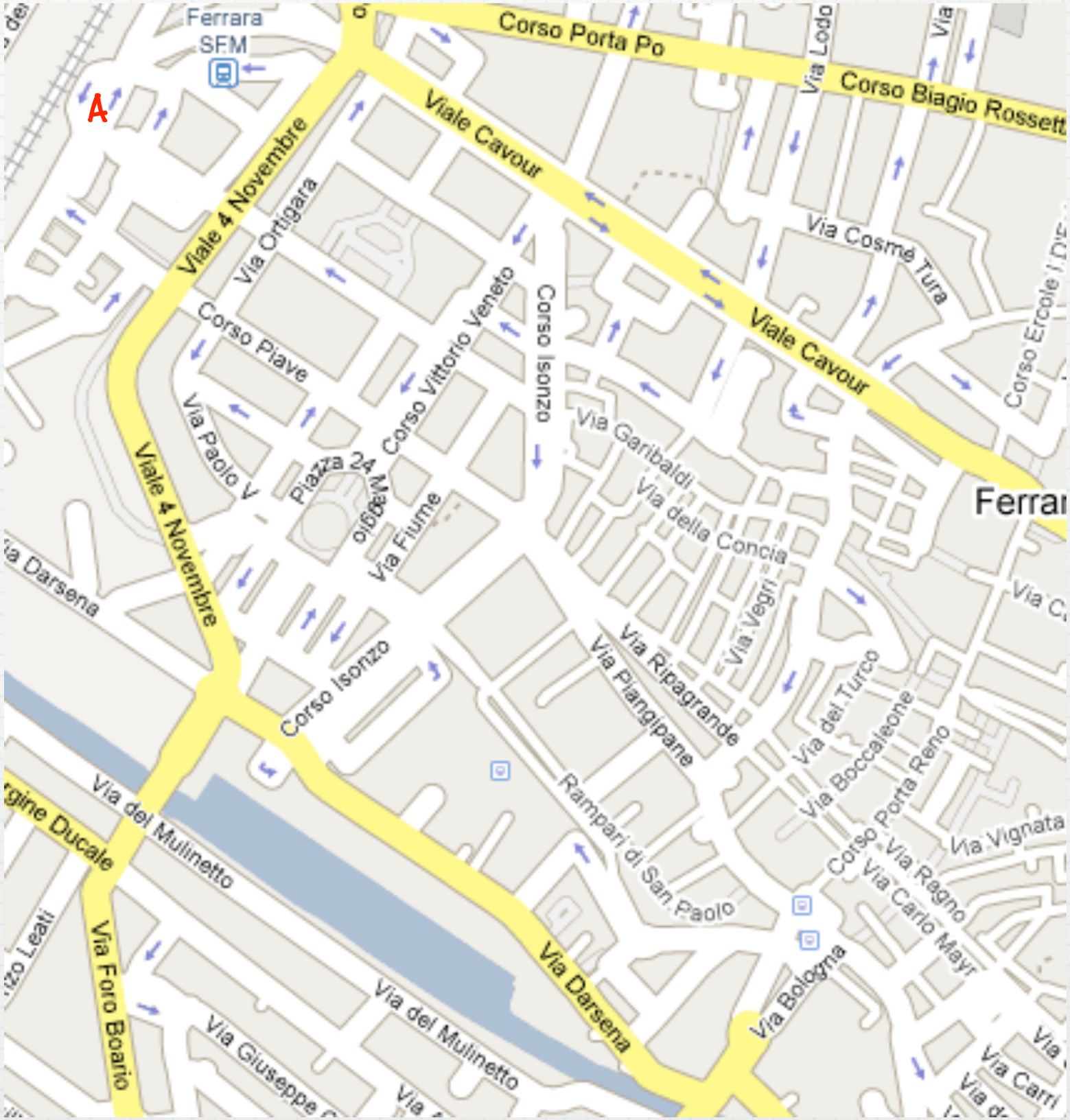

Ferrara
SFM
Corso Porta Po
Corso Biagio Rossetti
Viale Cavour
Viale 4 Novembre
Via Ortigara
Via Lodo
Via Cosmè Tura
Corso Piave
Corso Vittorio Veneto
Corso Isonzo
Via Garibaldi
Via della Concia
Via Fiume
Piazza 24 Maggio
Via Darsena
Via Ripagrande
Via Piangipane
Via del Turco
Via Boccaleone
Corso Porta Reno
Via del Mulinetto
Rampari di San Paolo
Via Vignatagliata
Corso Via Regno
Via Carlo Mayr
Via Bologna
Via Foro Boario
Via Giuseppe

Example


A
Ferrara
SFM
Corso Porta Po
Corso Biagio Rossetti
Viale Cavour
Viale 4 Novembre
Via Ortigara
Via Cosmè Tura
Corso Piave
Corso Vittorio Veneto
Corso Isonzo
Via Garibaldi
Via della Concia
Via Fiume
Via Darsena
Via Ripagrande
Via Piangipane
Rampari di San Paolo
Via del Turco
Via Boccaleone
Corso Porta Reno
Via Vignata
Corso Via Reggio
Via Carlo Mayr
Via Bologna
Via del Mulinetto
Via Foro Boario
Via Giuseppe
Via Carri

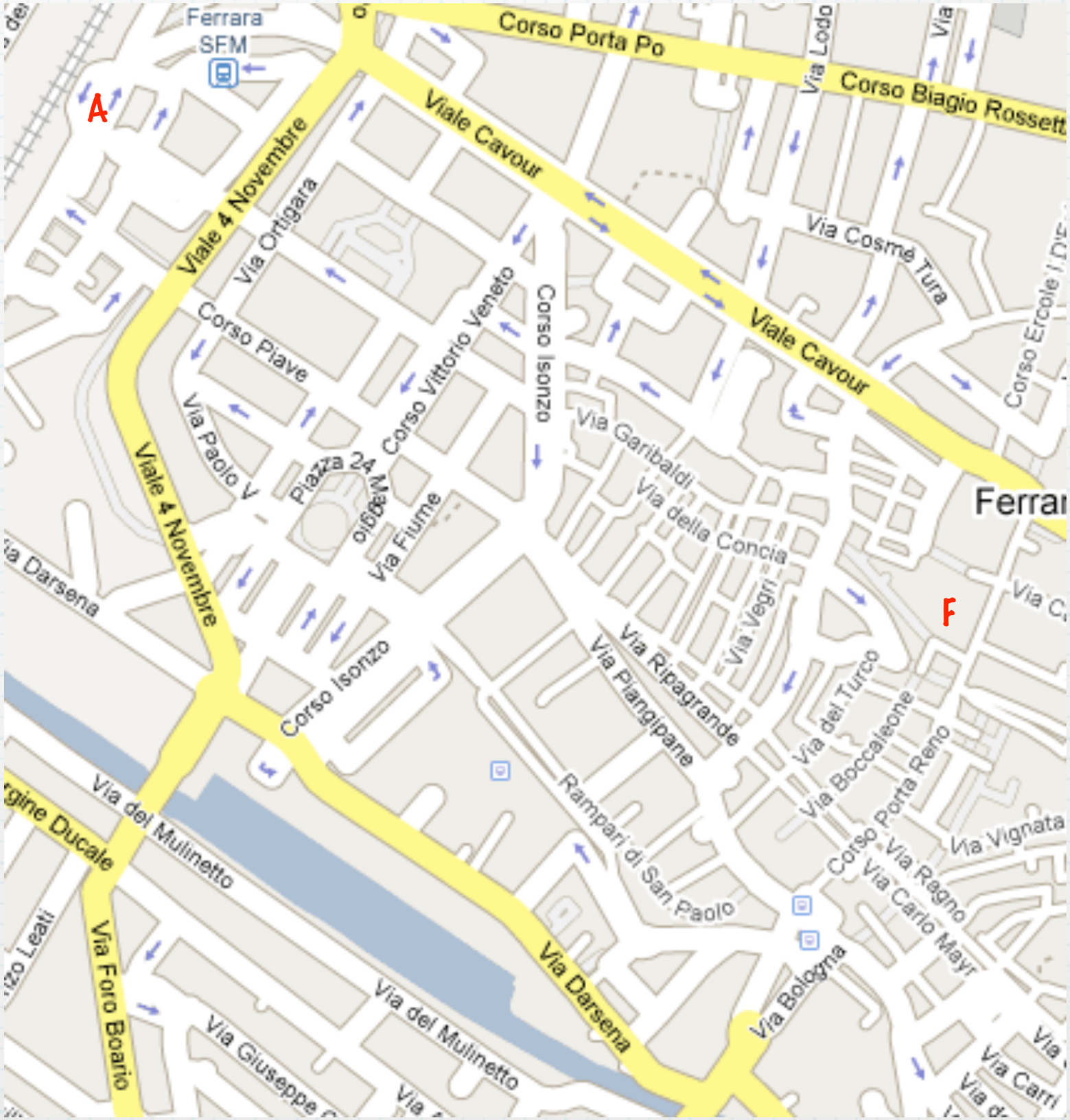
Example

A: starting place



Example

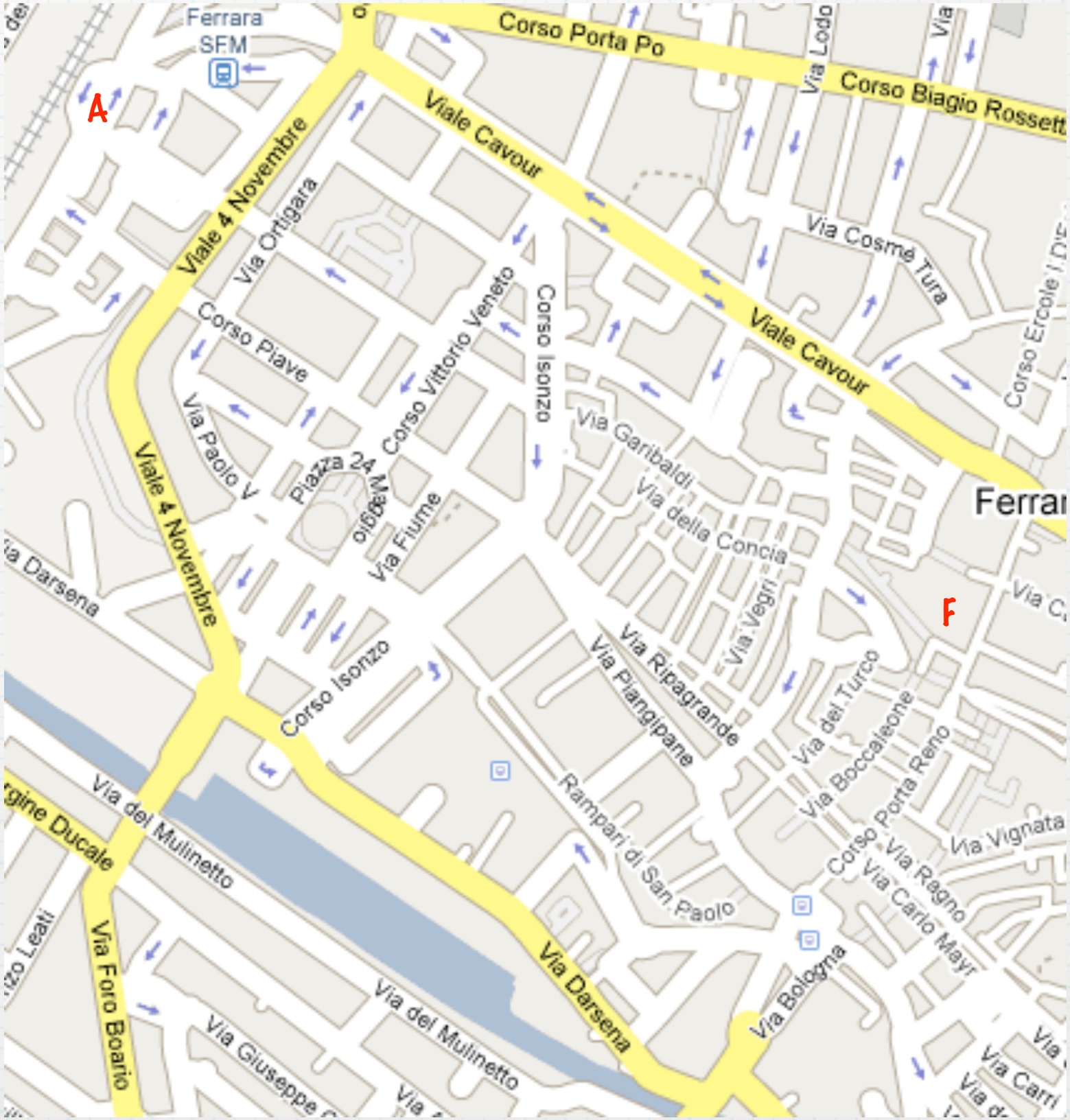
A: starting place



Example

A: starting place

F: destination

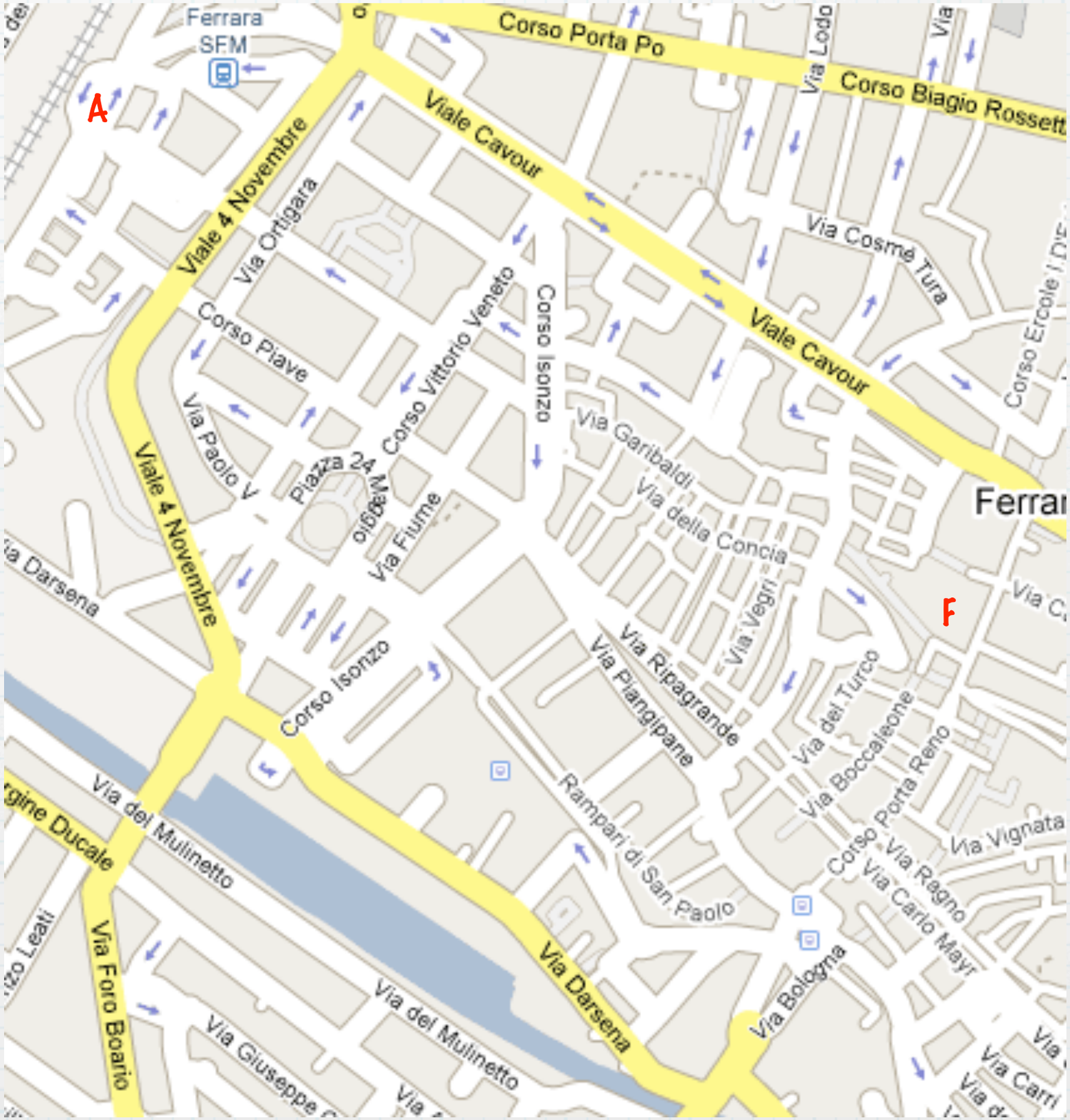


Example

site or intersection

A: starting place

F: destination

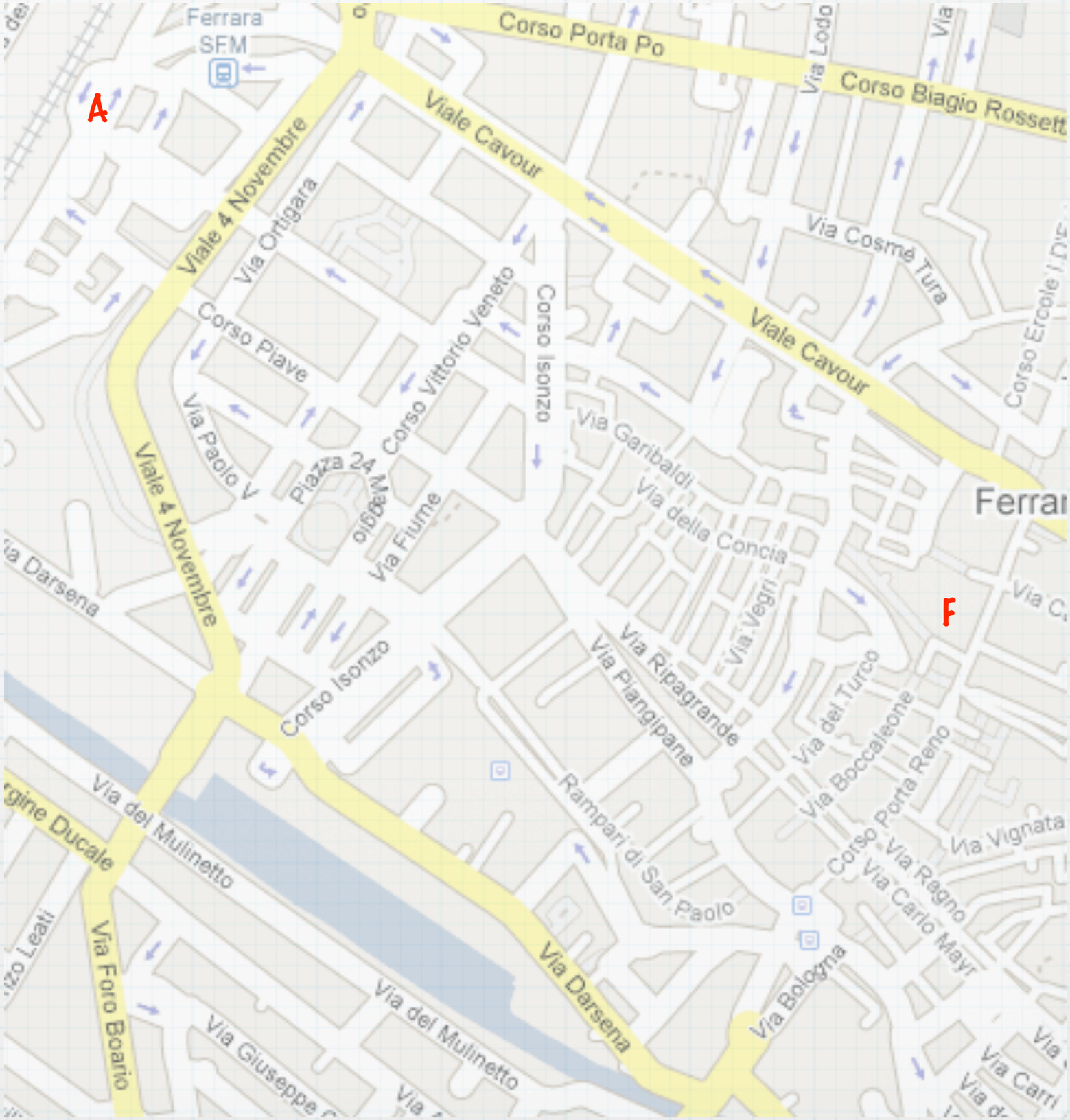


Example

site or intersection

A: starting place

F: destination

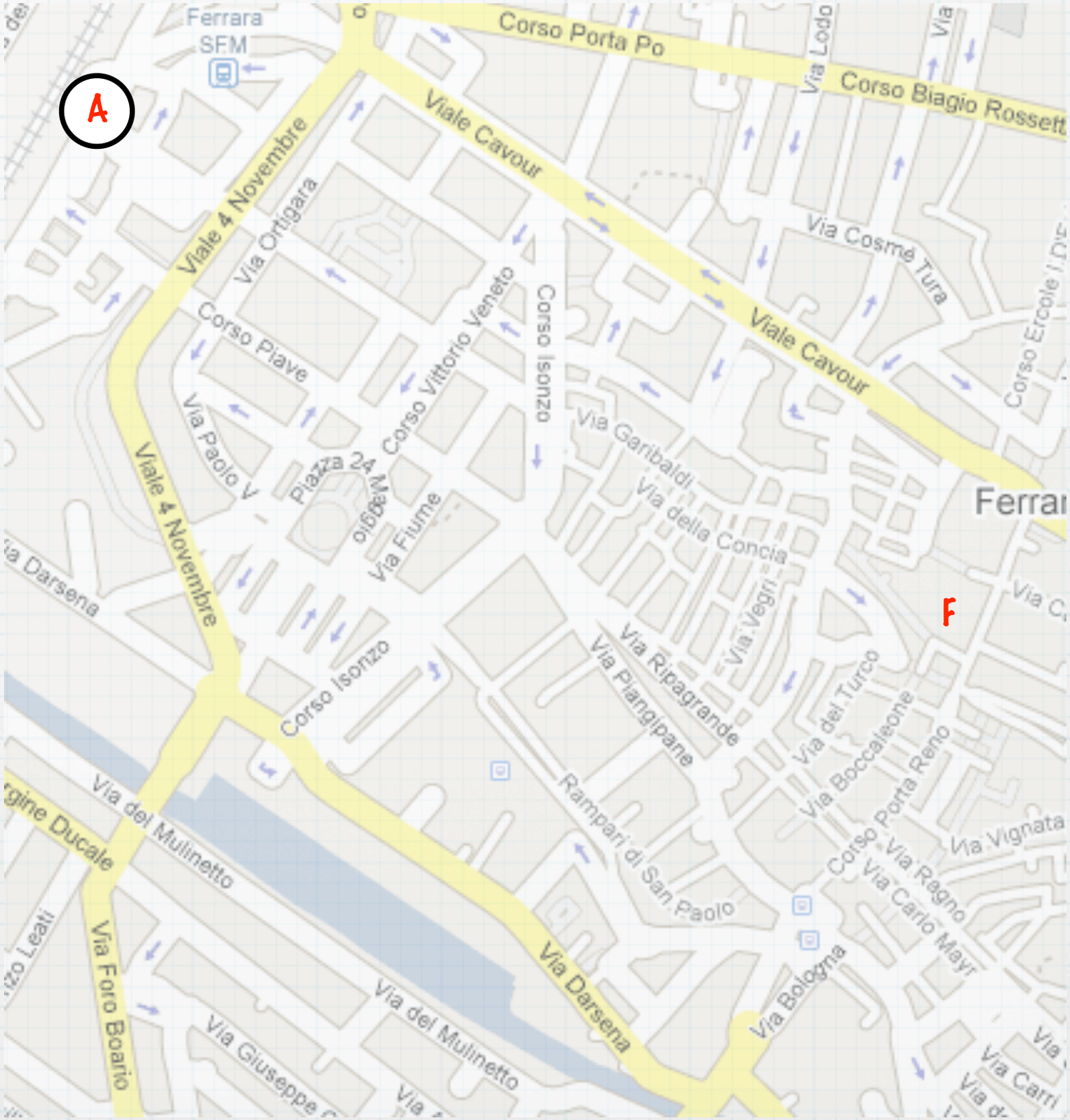


Example

site or intersection

A: starting place

F: destination

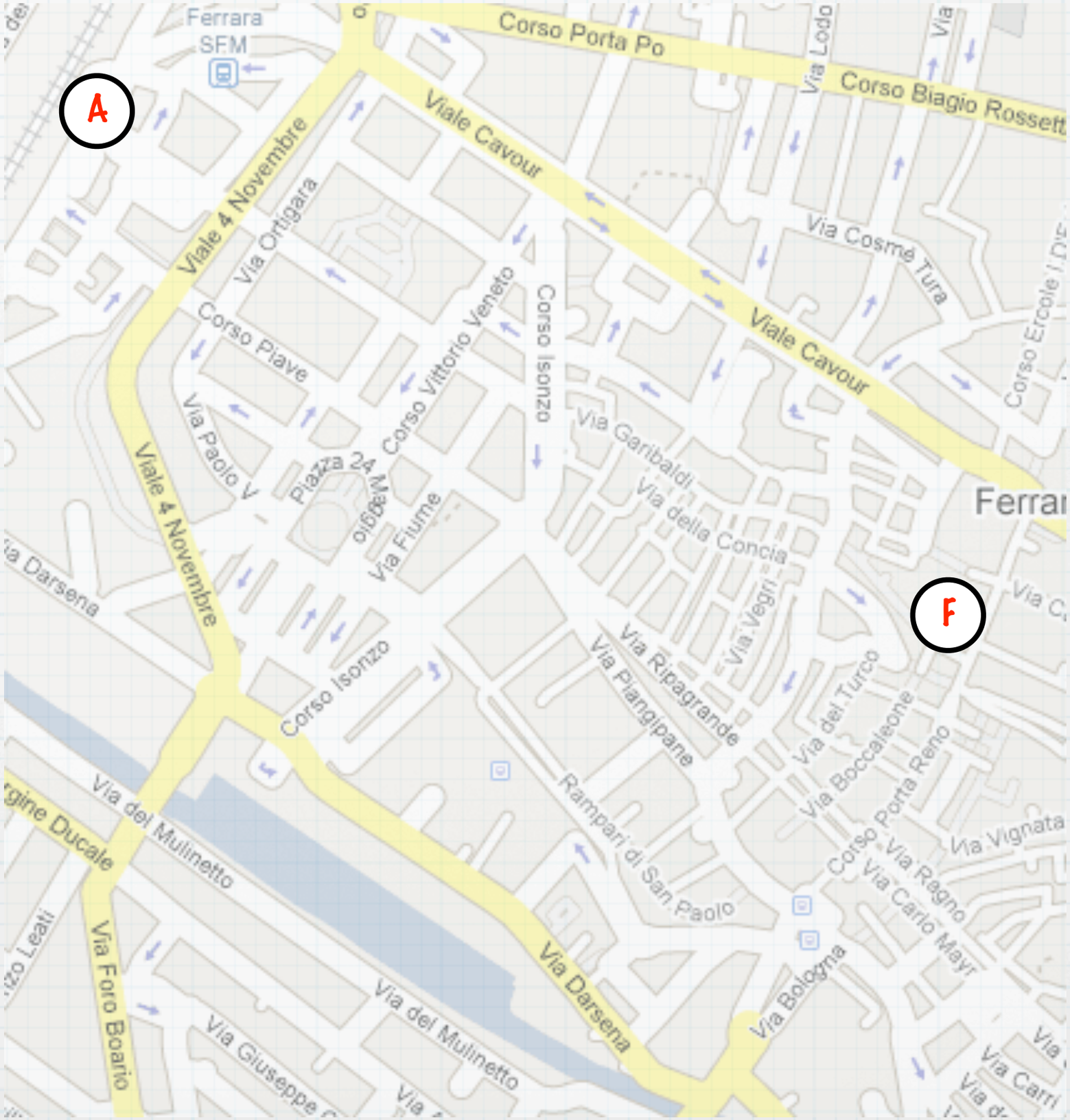


Example

site or intersection

A: starting place

F: destination

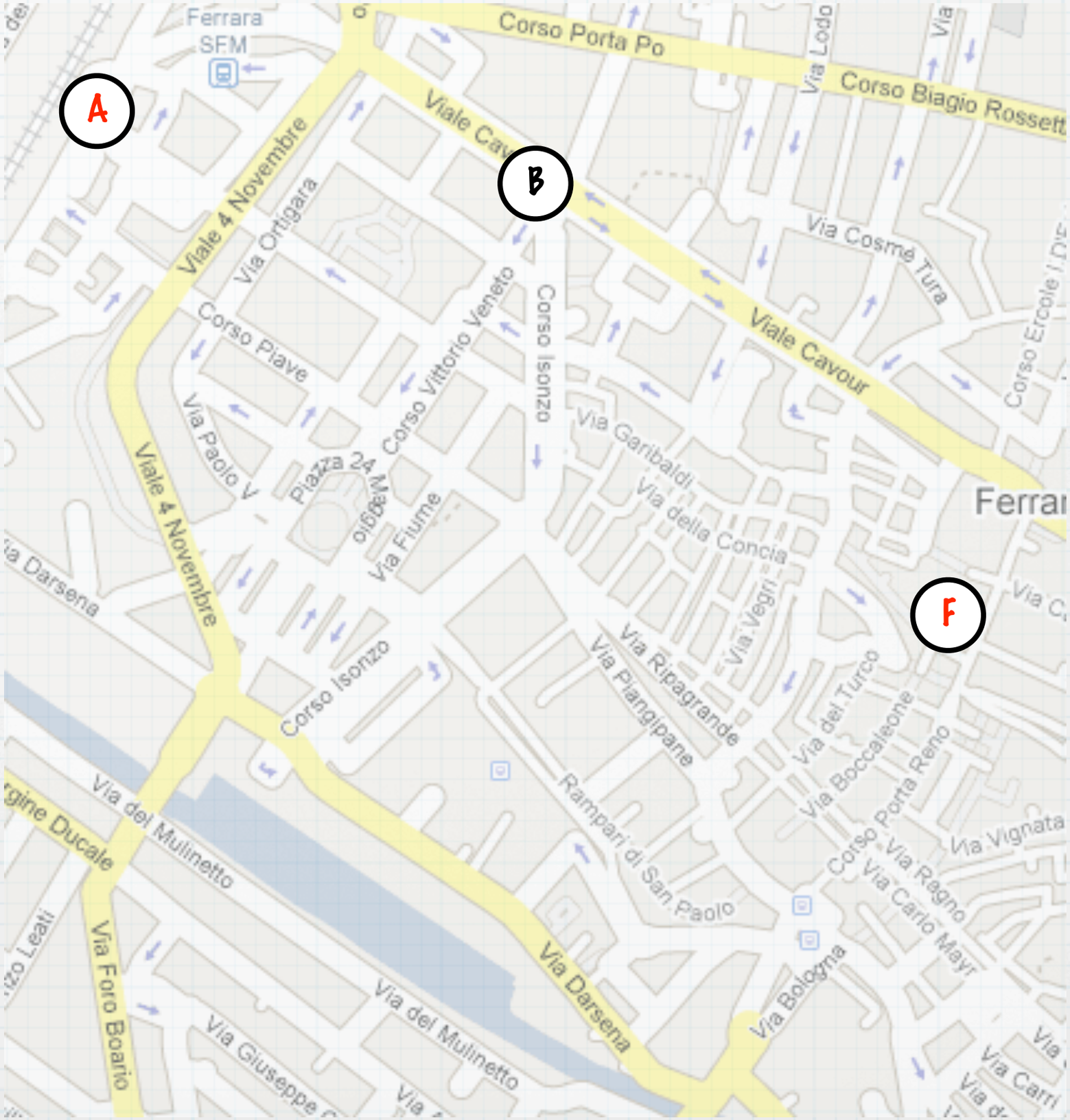


Example

site or intersection

A: starting place

F: destination

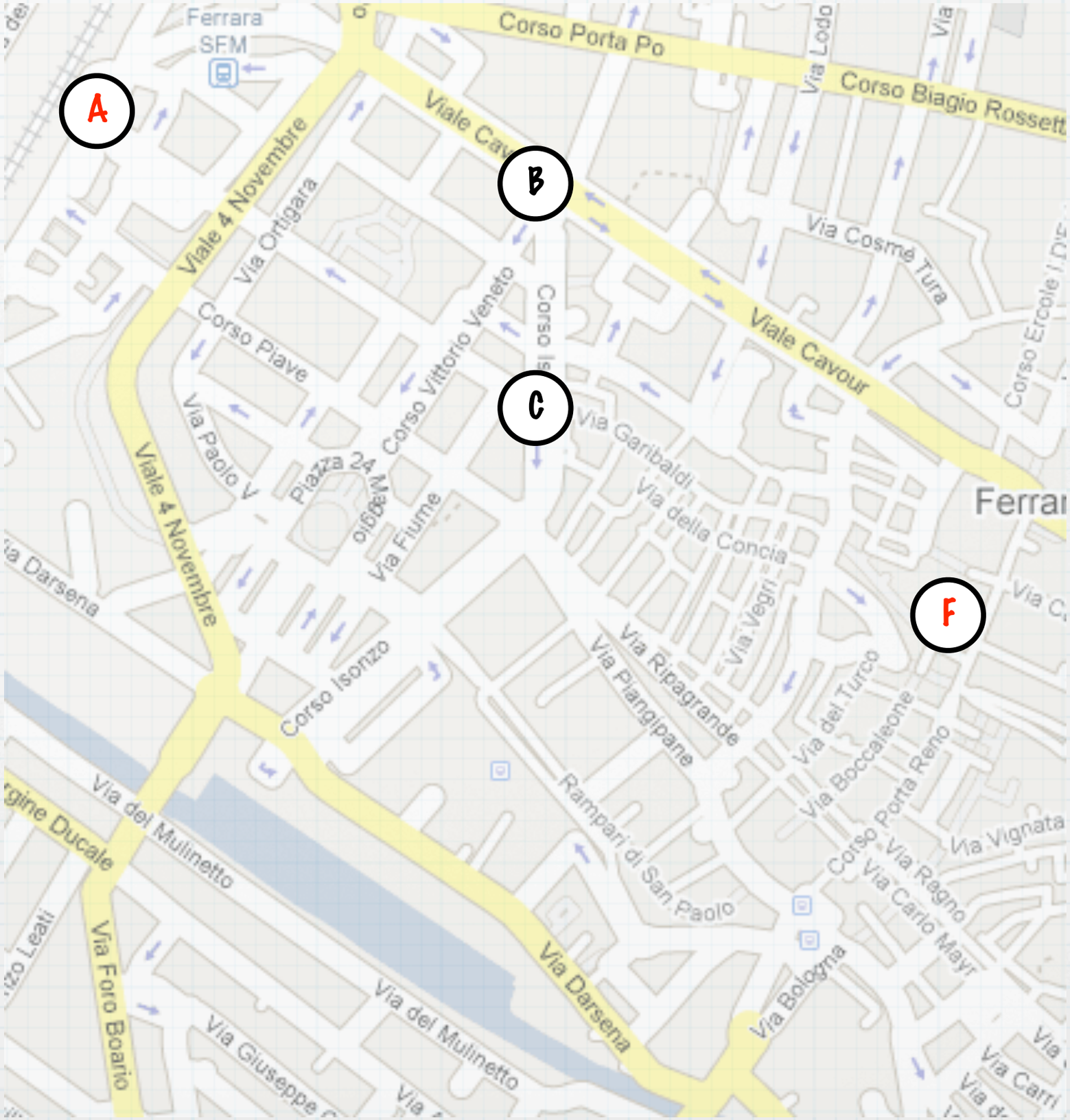


Example

site or intersection

A: starting place

F: destination

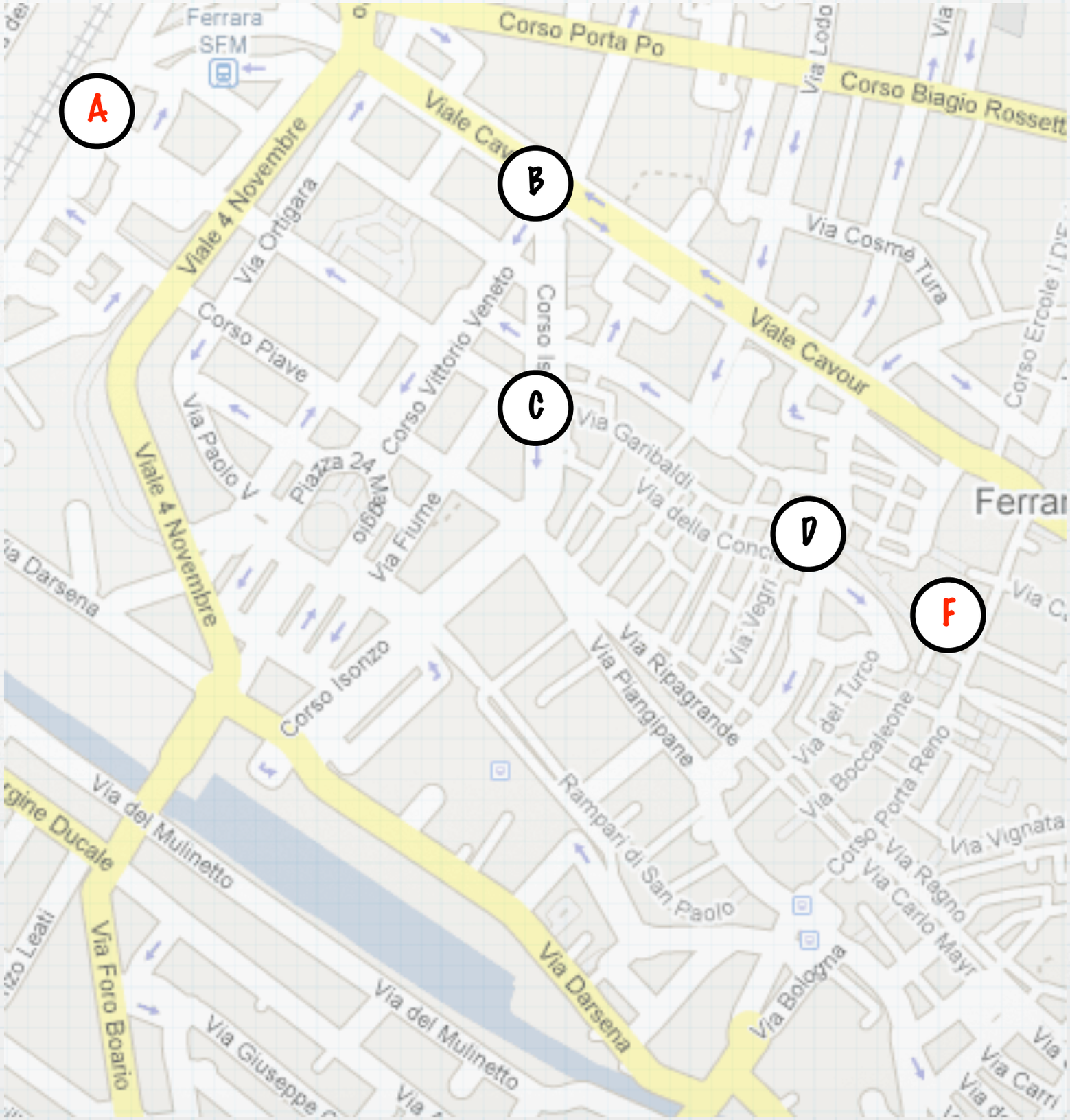


Example

site or intersection

A: starting place

F: destination

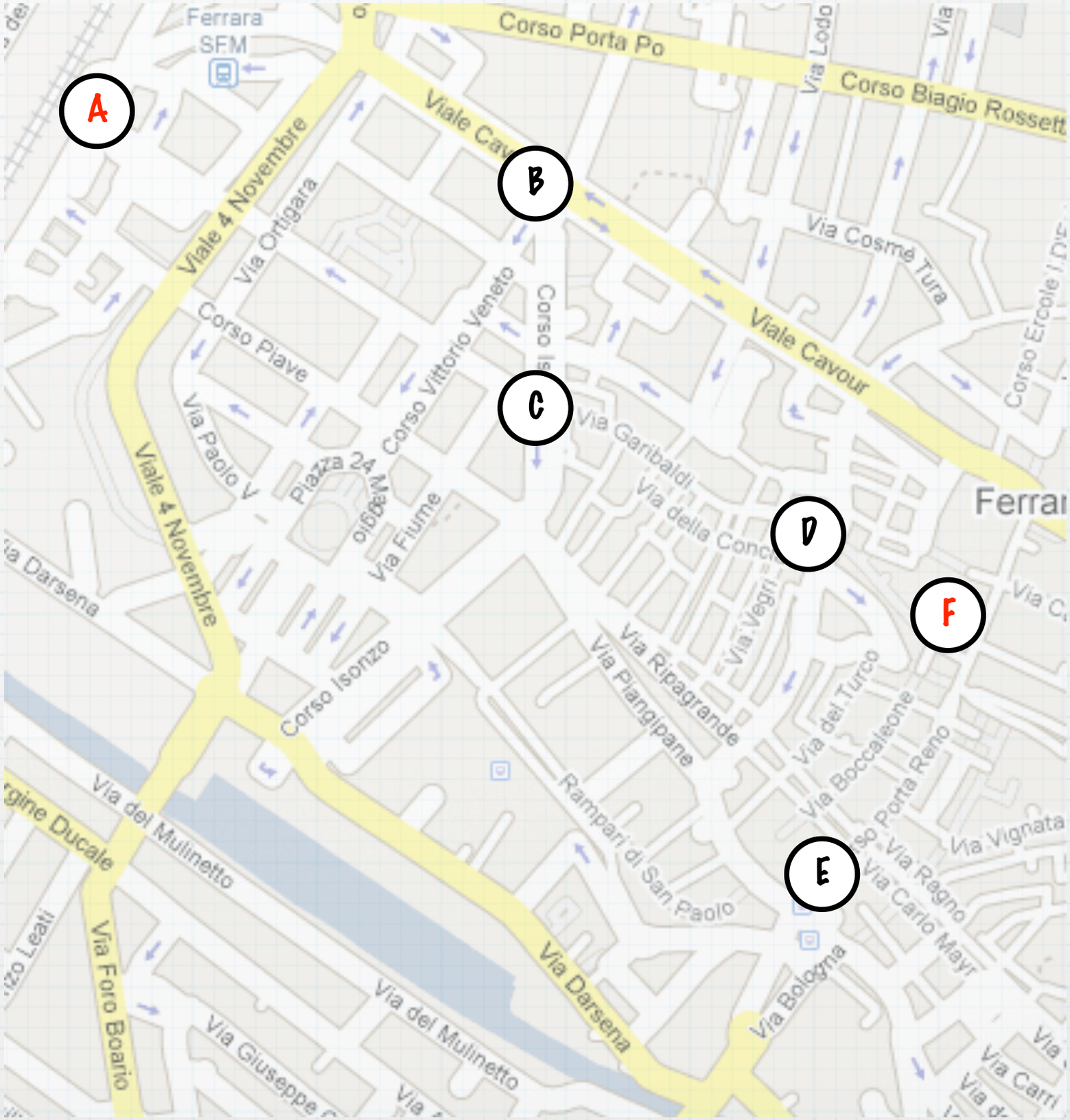


Example

site or intersection

A: starting place

F: destination



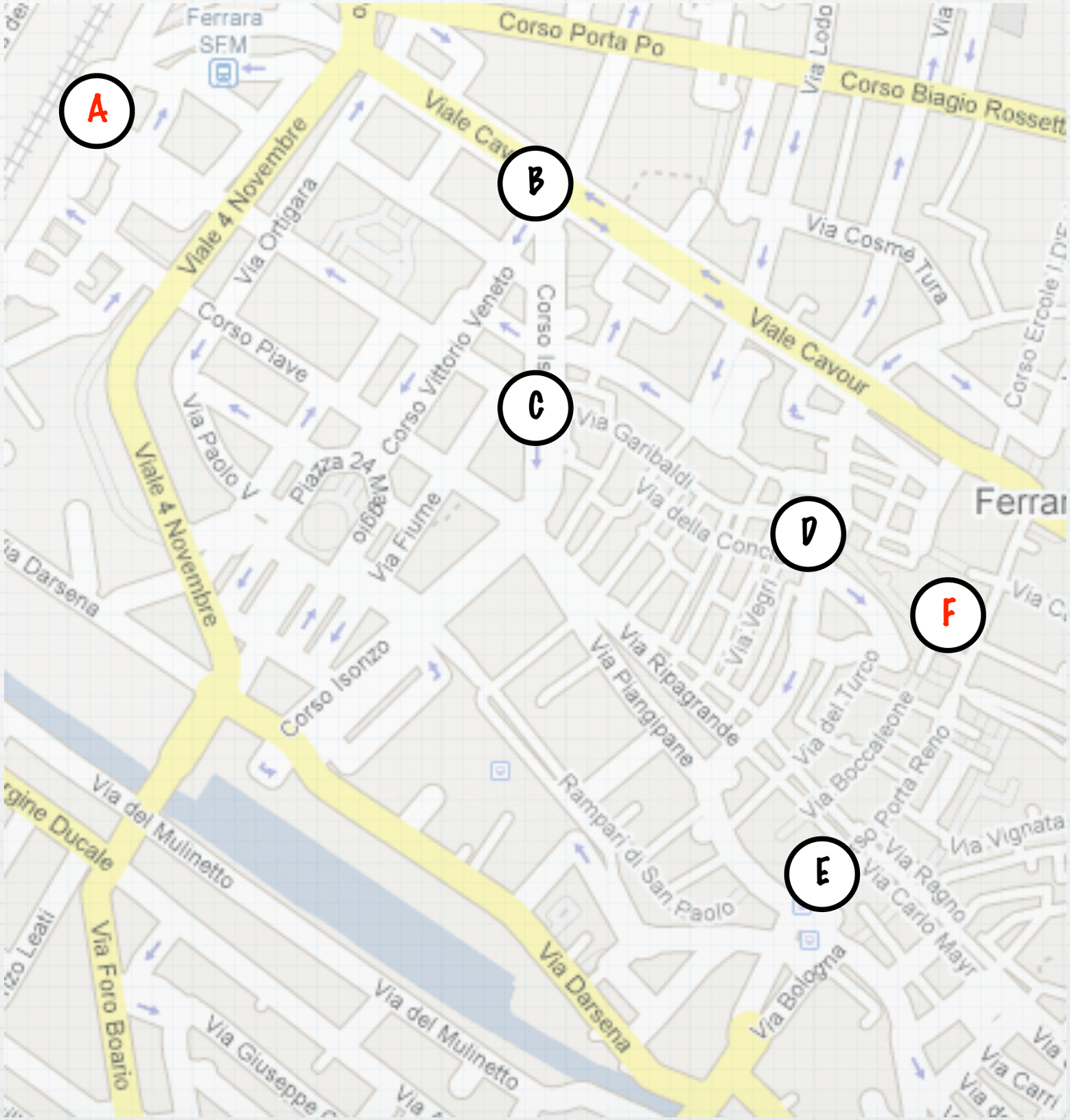
Example

site or intersection

street

A: starting place

F: destination



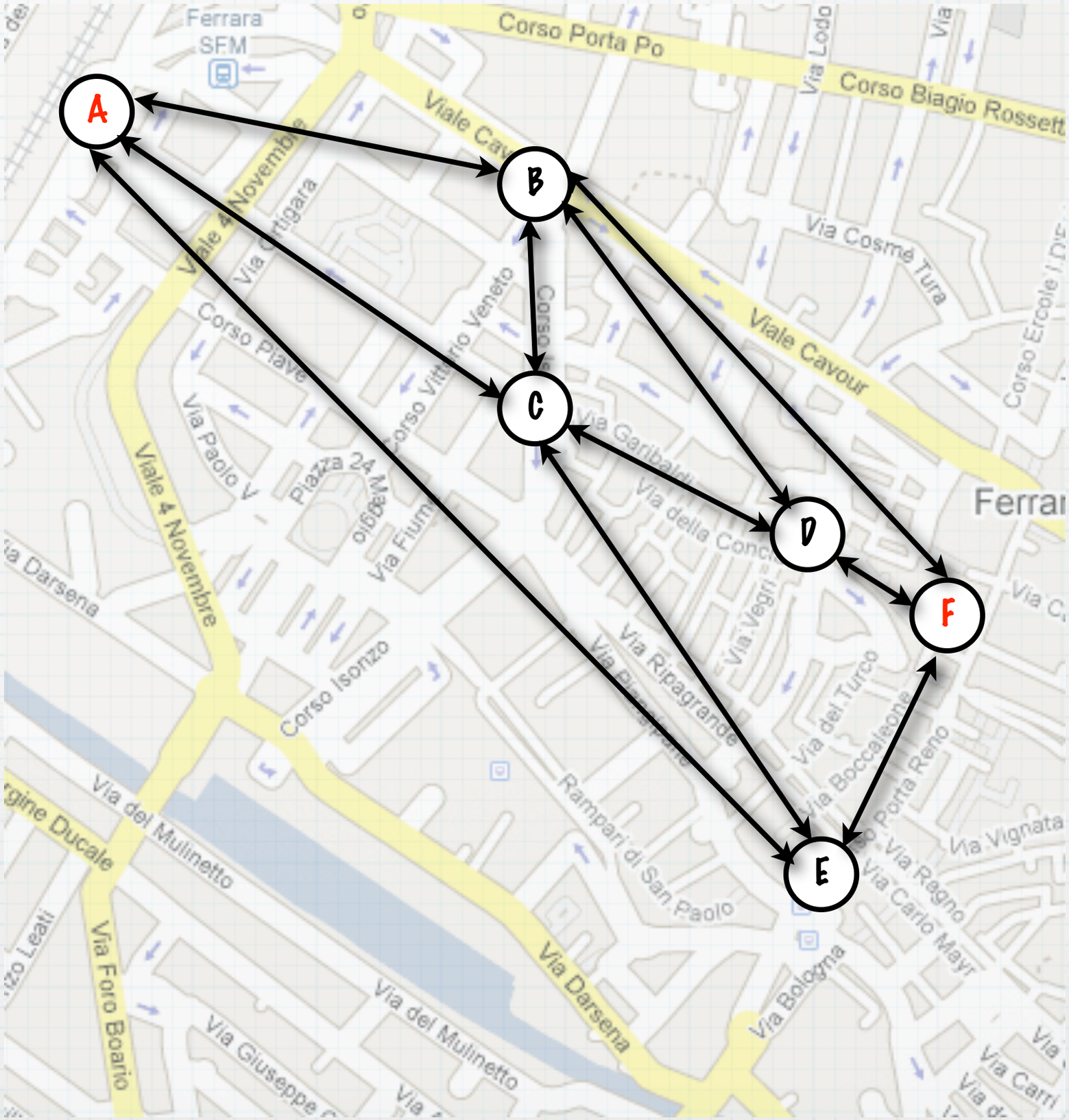
Example

site or intersection

street

A: starting place

F: destination



Example

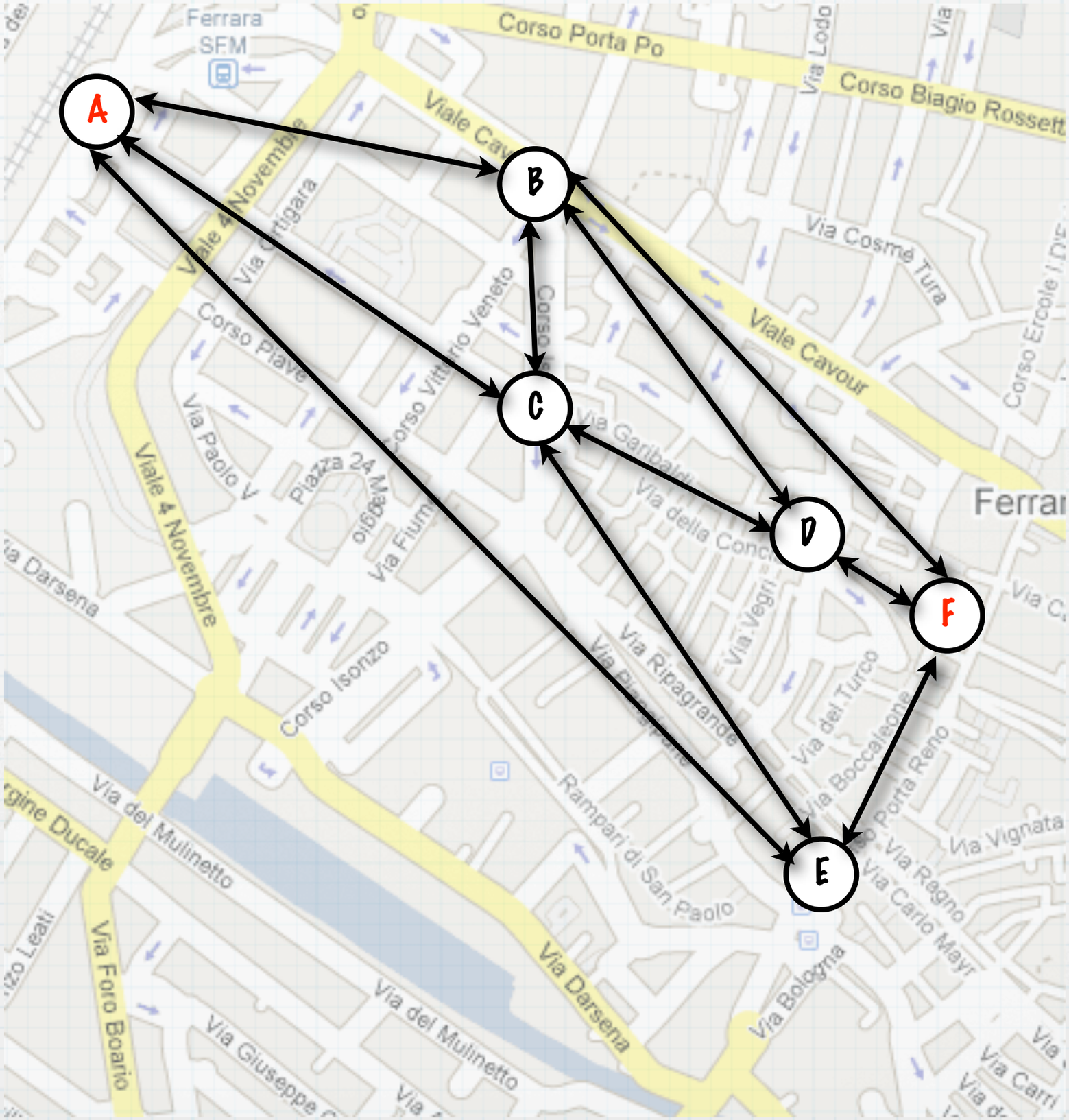
site or intersection

street

5 walking distance (min.)

A: starting place

F: destination



Example

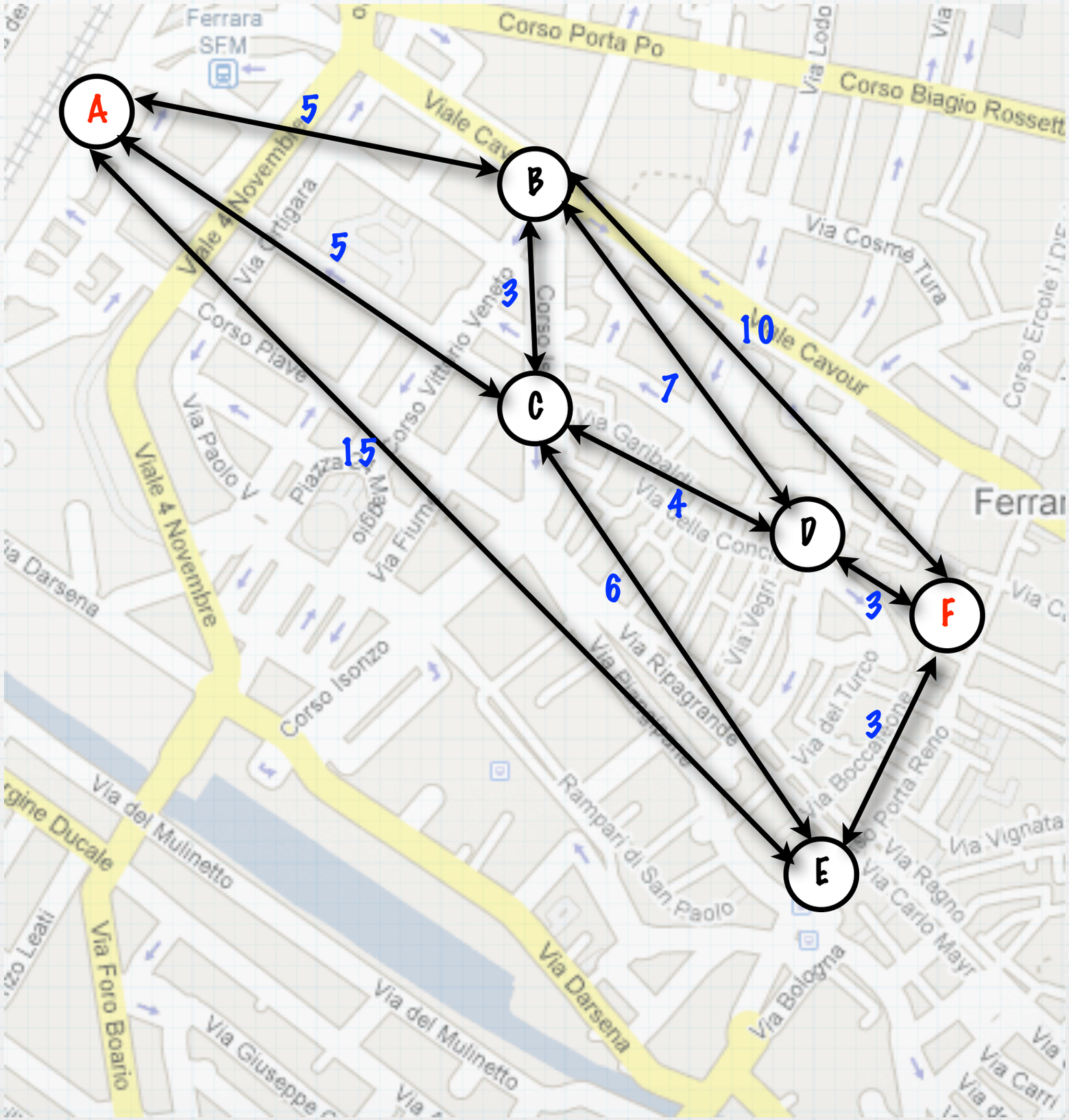
site or intersection

street

5 walking distance (min.)

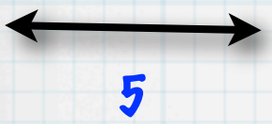
A: starting place

F: destination

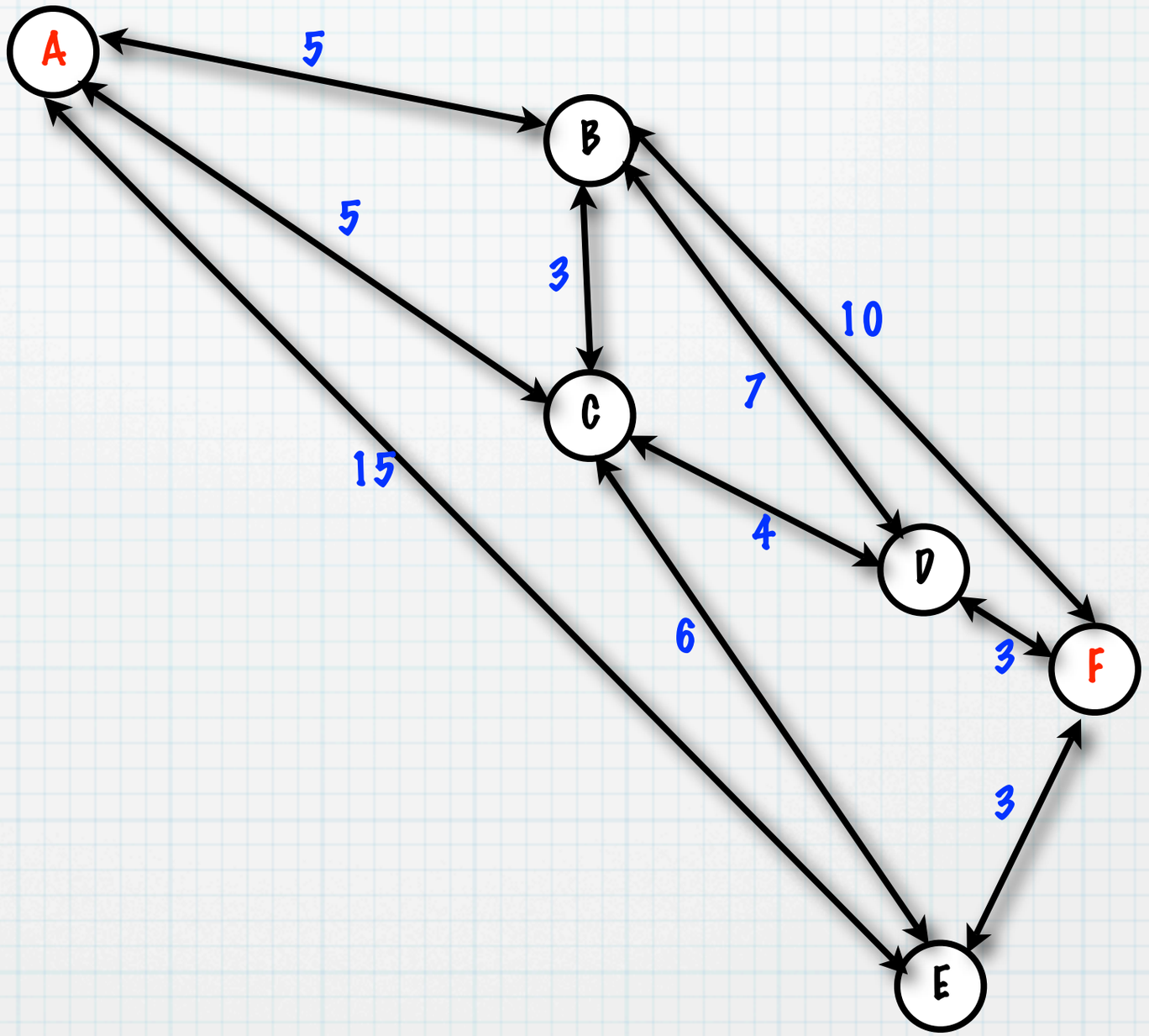


Graph

node / vertex

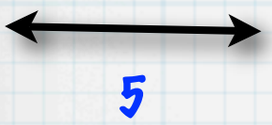


arc / edge
length

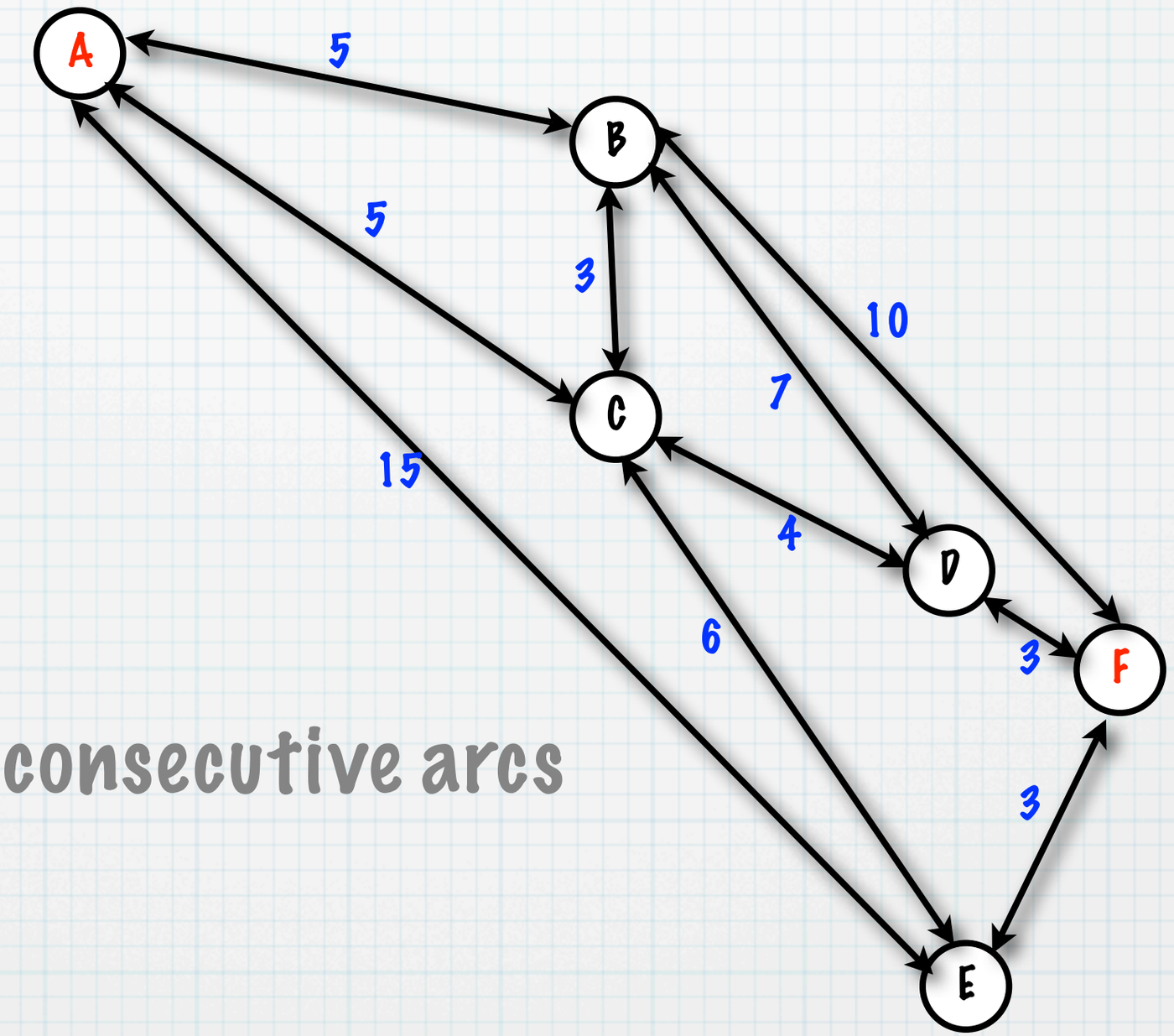


Graph

node / vertex



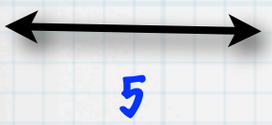
arc / edge
length



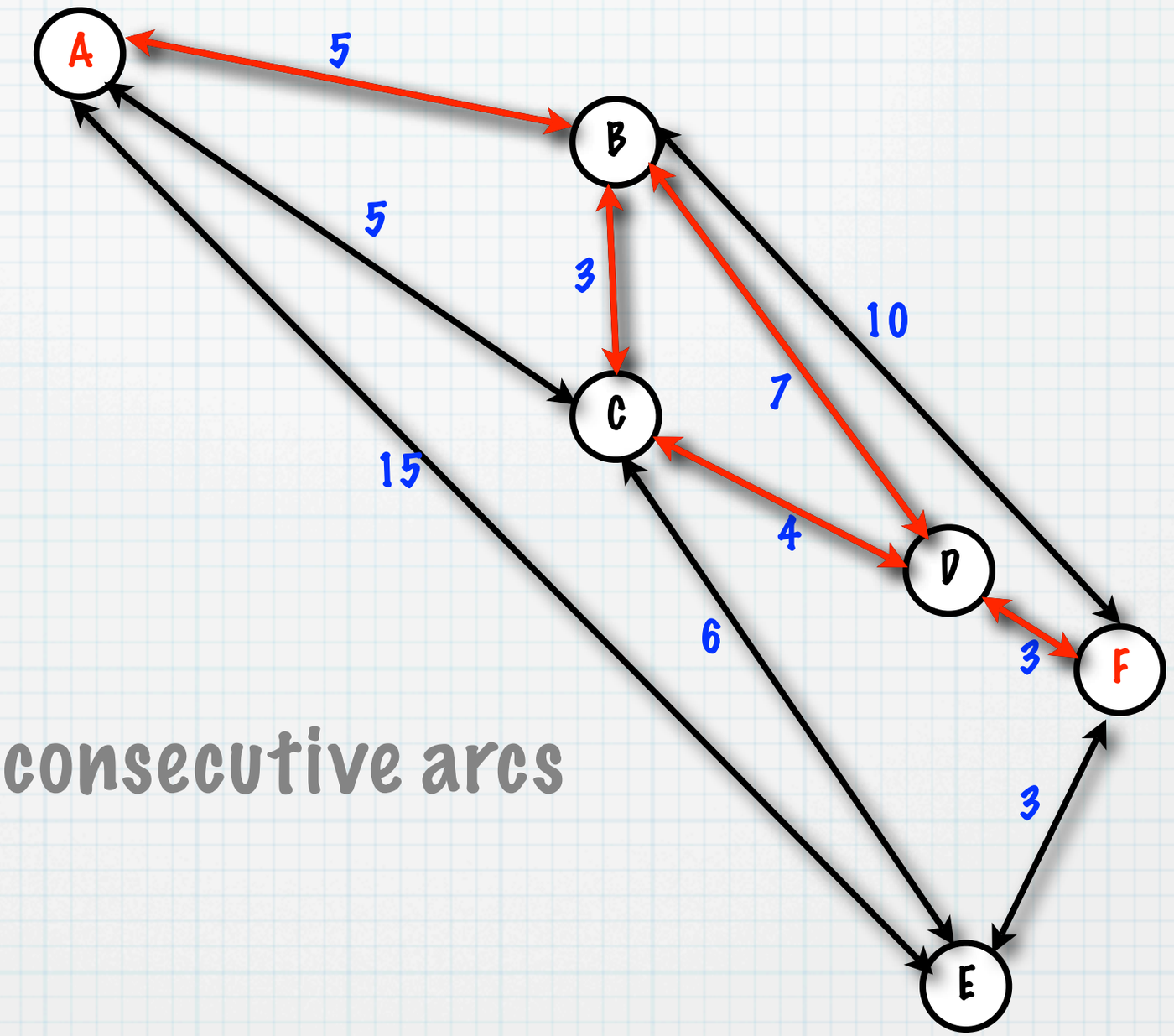
path: sequence of consecutive arcs

Graph

node / vertex



arc / edge
length



path: sequence of consecutive arcs

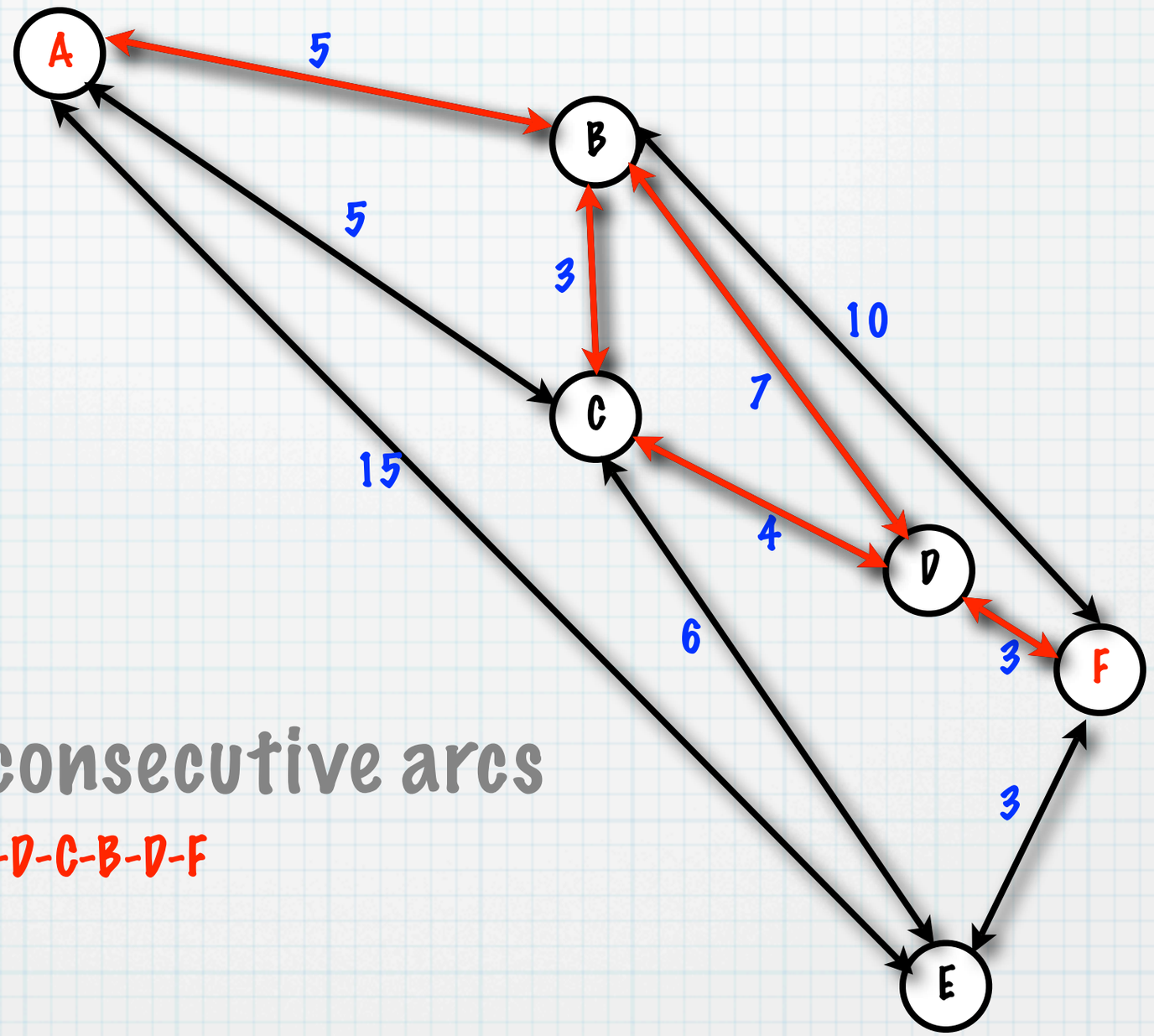
Graph

node / vertex

5

arc / edge

length

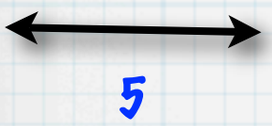


path: sequence of consecutive arcs

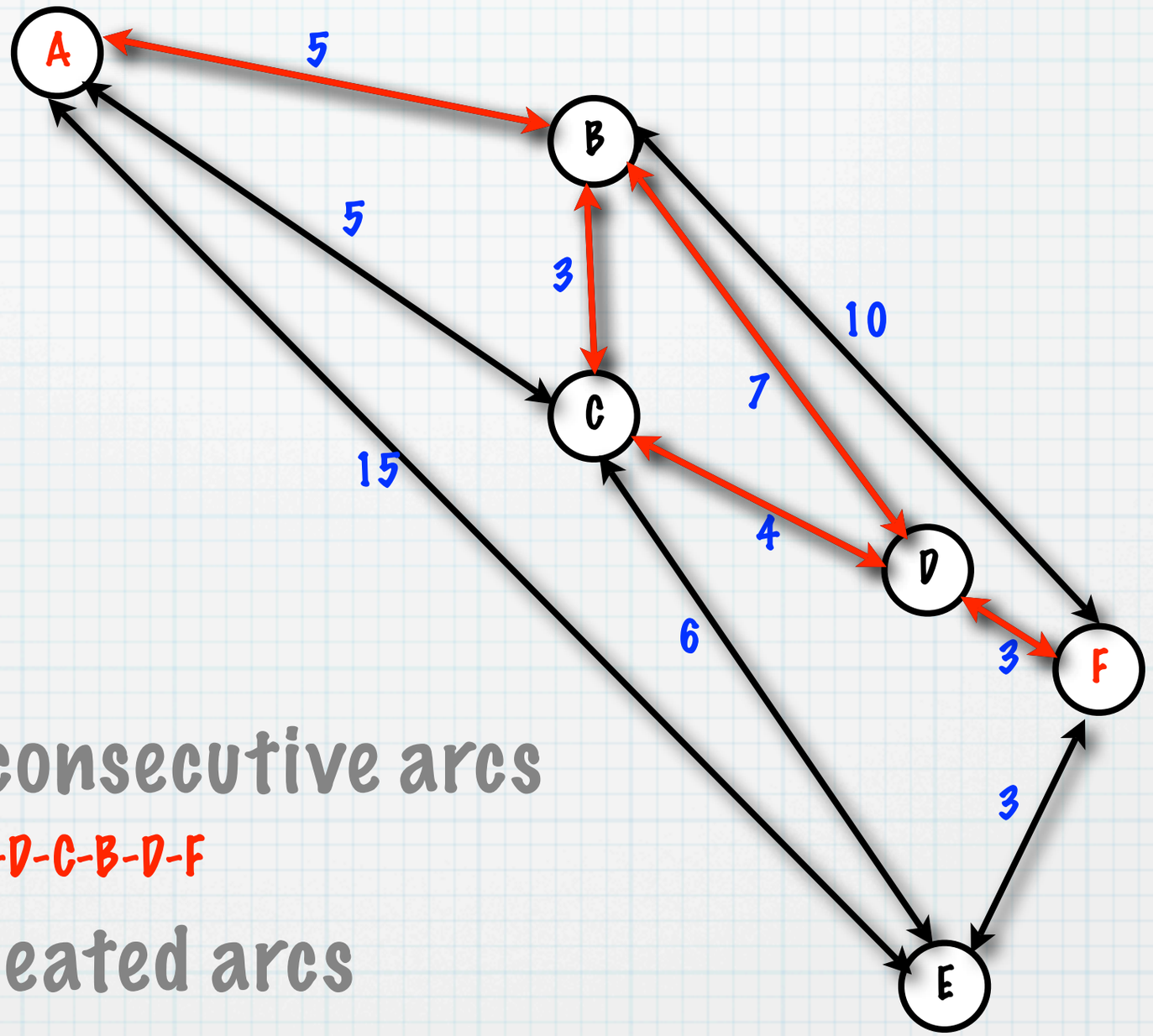
A-B-D-C-B-D-F

Graph

node / vertex



arc / edge
length



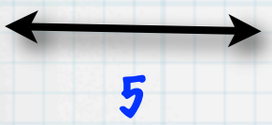
path: sequence of consecutive arcs

A-B-D-C-B-D-F

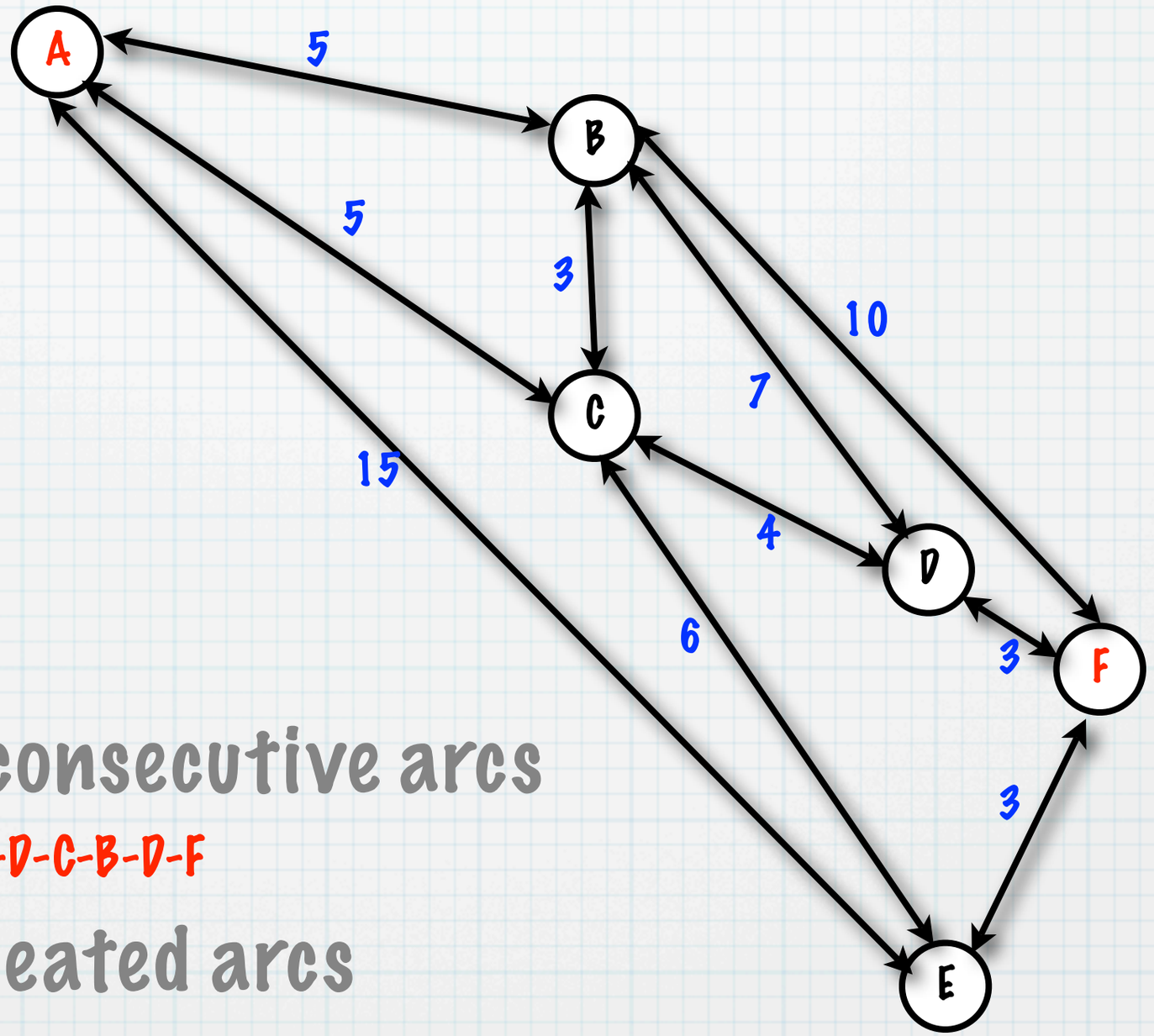
simple path: no repeated arcs

Graph

node / vertex



arc / edge
length



path: sequence of consecutive arcs

A-B-D-C-B-D-F

simple path: no repeated arcs

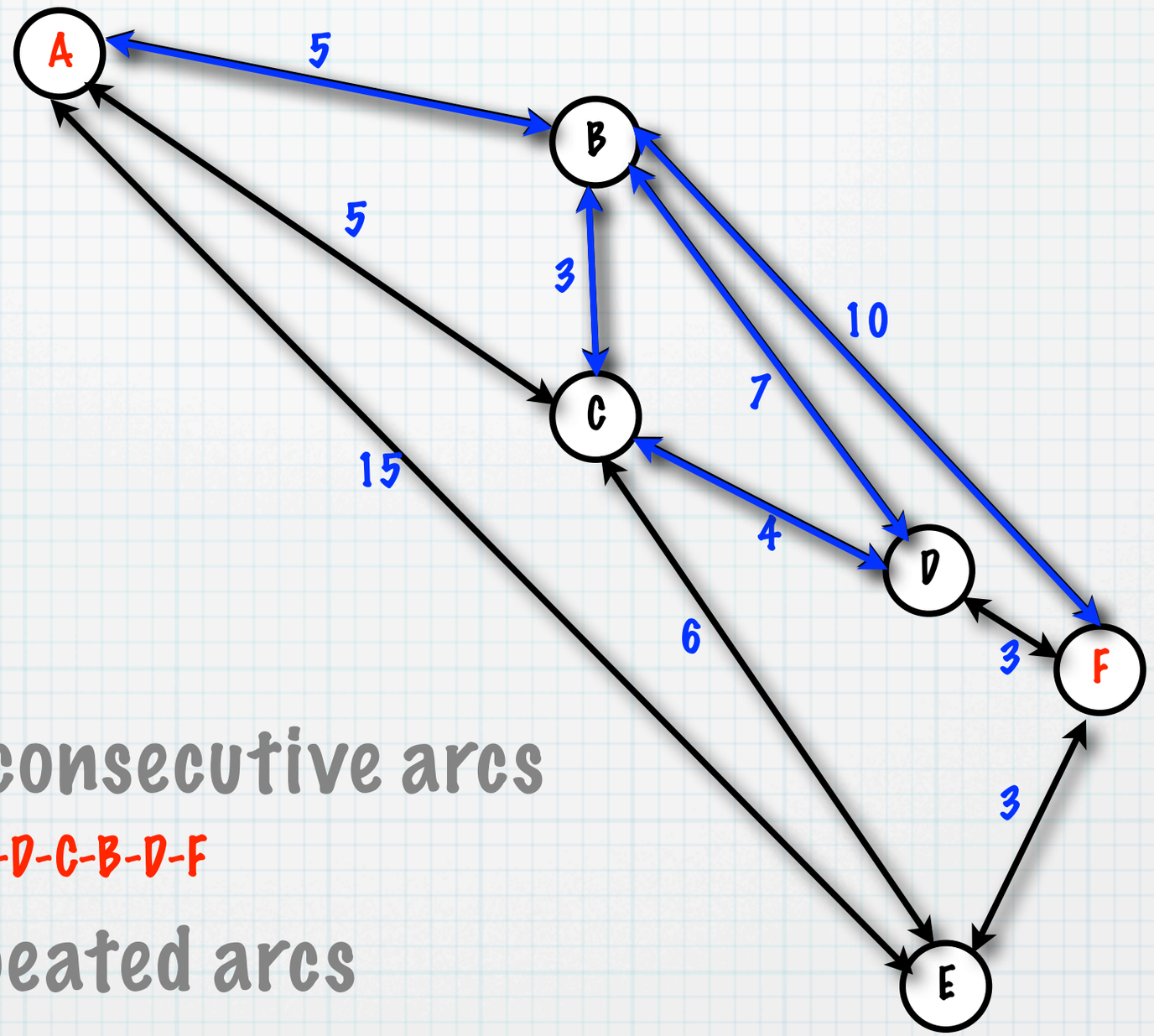
Graph

node / vertex

5

arc / edge

length



path: sequence of consecutive arcs

A-B-D-C-B-D-F

simple path: no repeated arcs

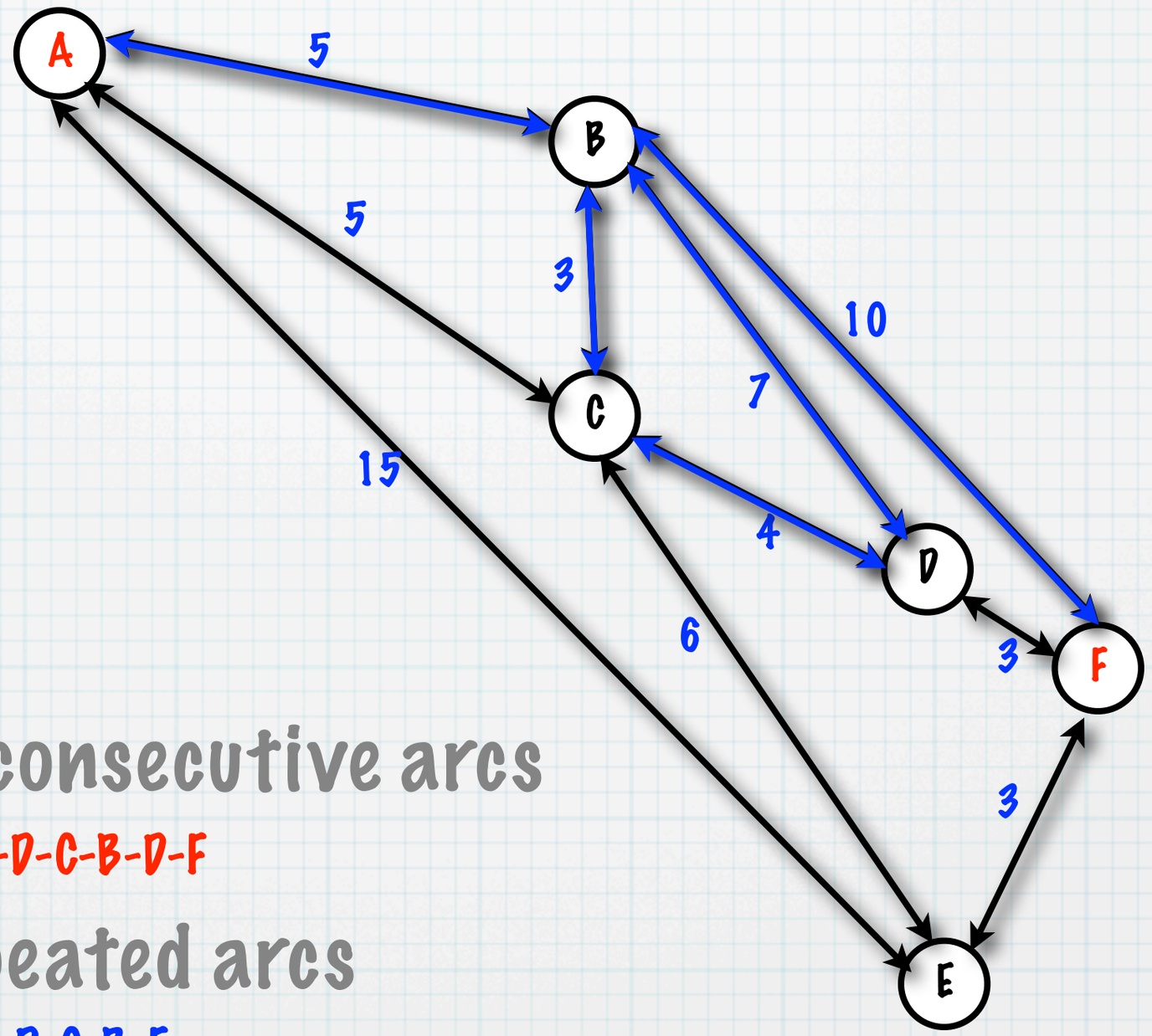
Graph

node / vertex

5

arc / edge

length



path: sequence of consecutive arcs

A-B-D-C-B-D-F

simple path: no repeated arcs

A-B-D-C-B-F

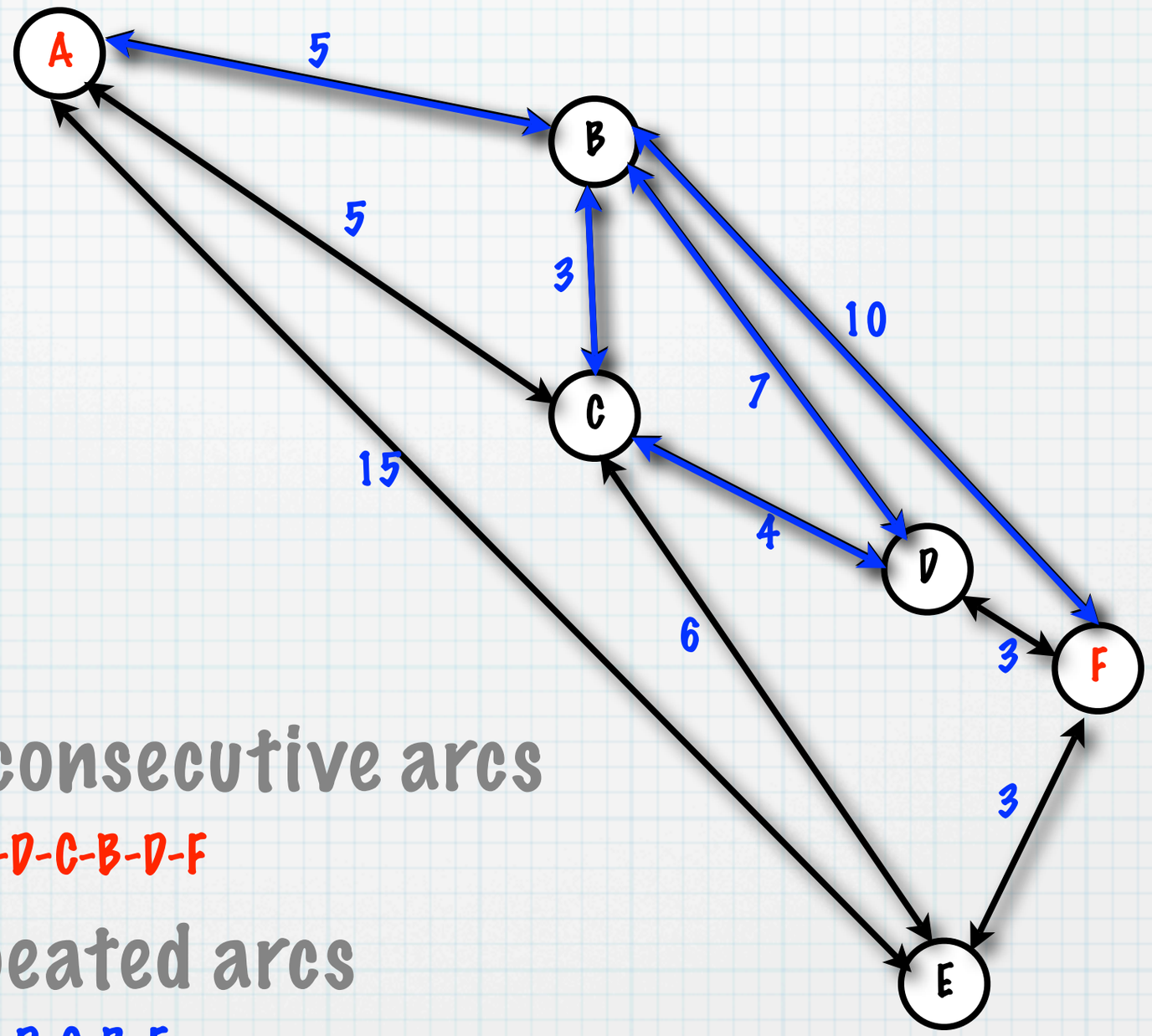
Graph

node / vertex

5

arc / edge

length



path: sequence of consecutive arcs

A-B-D-C-B-D-F

simple path: no repeated arcs

A-B-D-C-B-F

elementary path: no repeated nodes

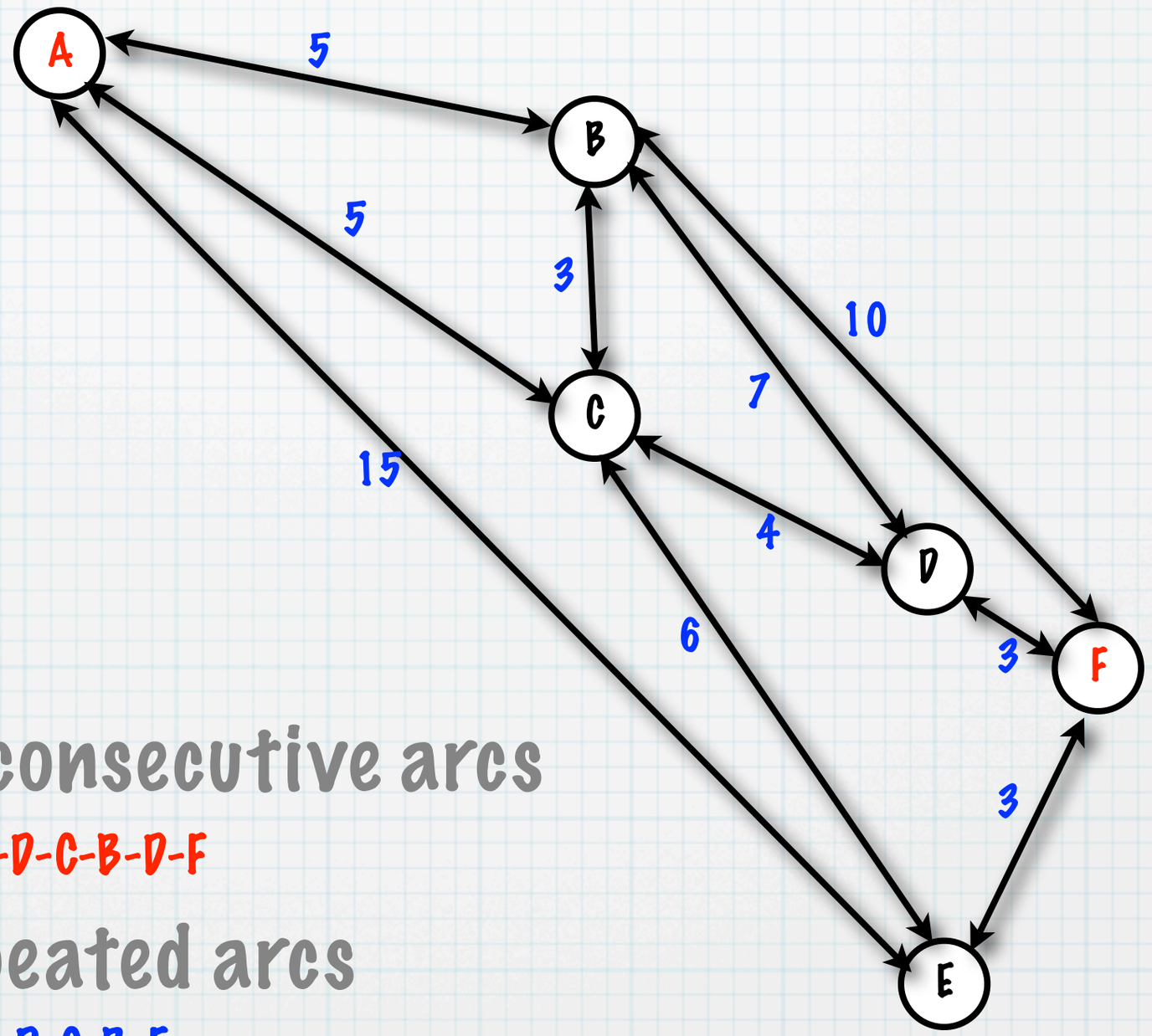
Graph

node / vertex

5

arc / edge

length



path: sequence of consecutive arcs

A-B-D-C-B-D-F

simple path: no repeated arcs

A-B-D-C-B-F

elementary path: no repeated nodes

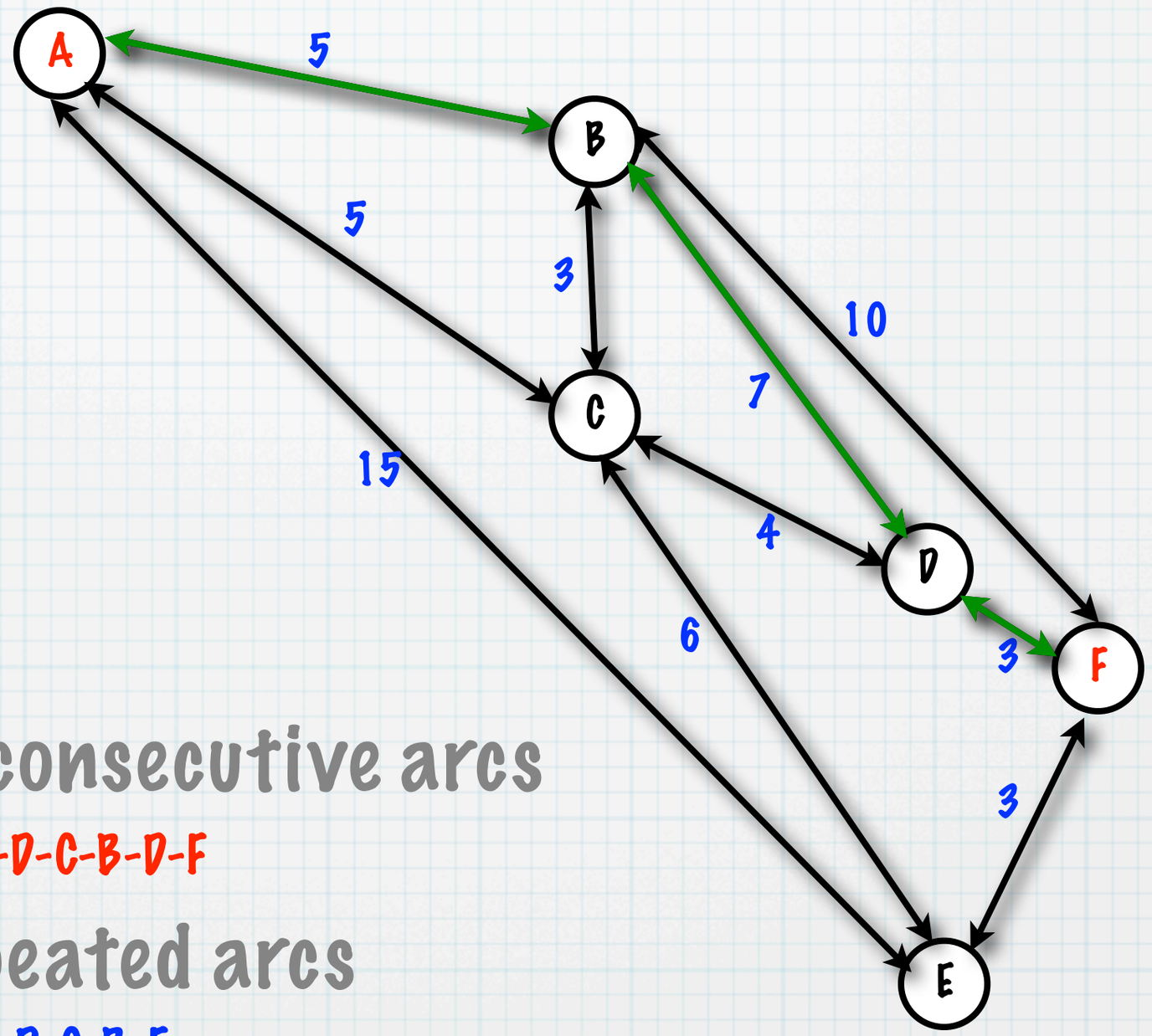
Graph

node / vertex

5

arc / edge

length



path: sequence of consecutive arcs

A-B-D-C-B-D-F

simple path: no repeated arcs

A-B-D-C-B-F

elementary path: no repeated nodes

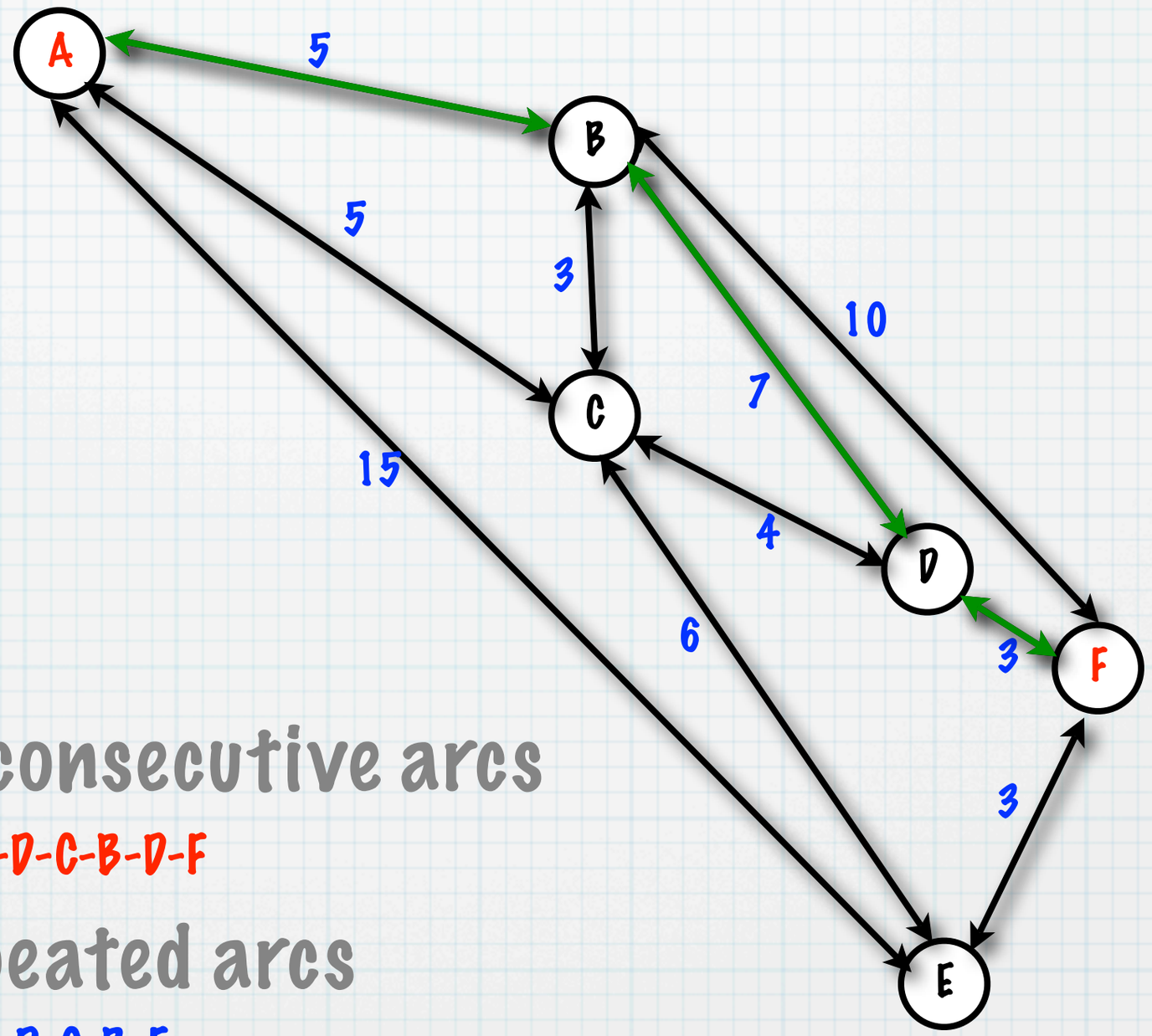
Graph

node / vertex

5

arc / edge

length



path: sequence of consecutive arcs

A-B-D-C-B-D-F

simple path: no repeated arcs

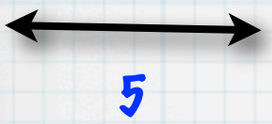
A-B-D-C-B-F

elementary path: no repeated nodes

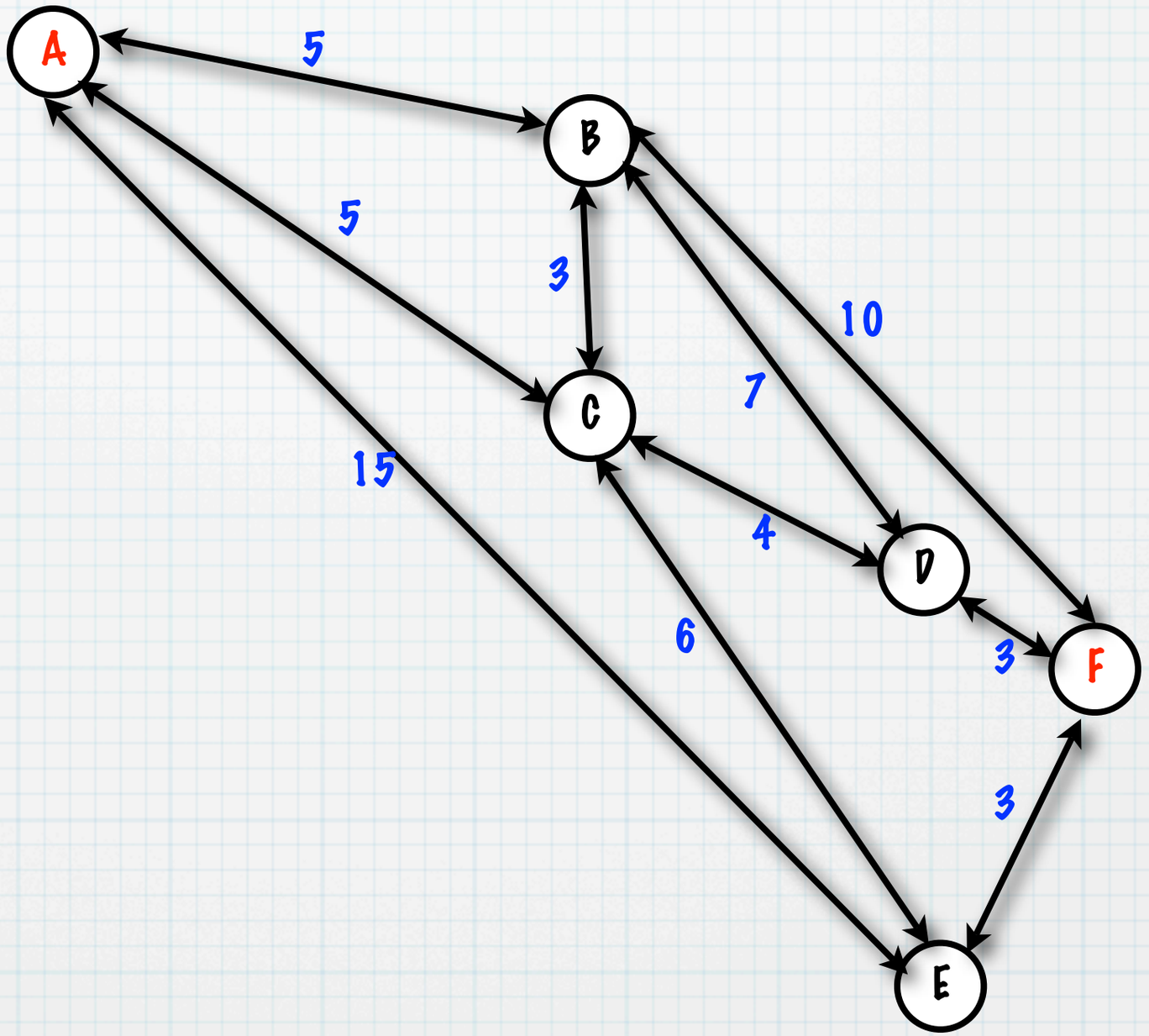
A-B-D-F

Graph

node / vertex

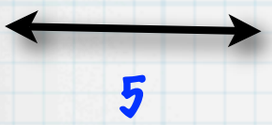


arc / edge
length



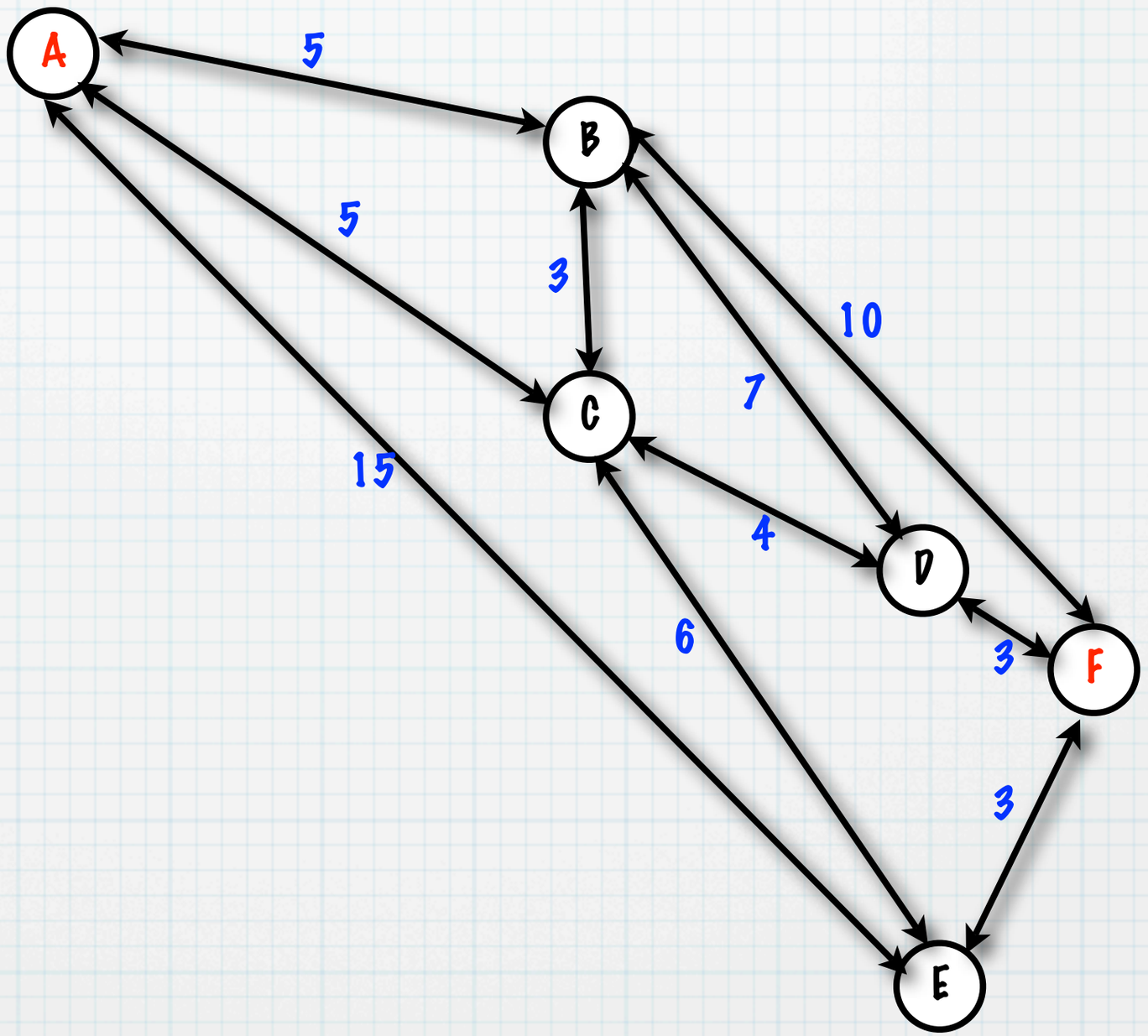
Graph

node / vertex

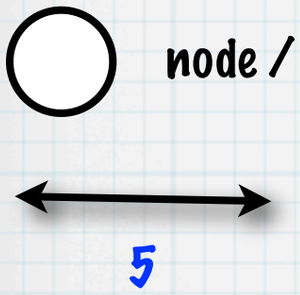


arc / edge
length

cycle: closed path

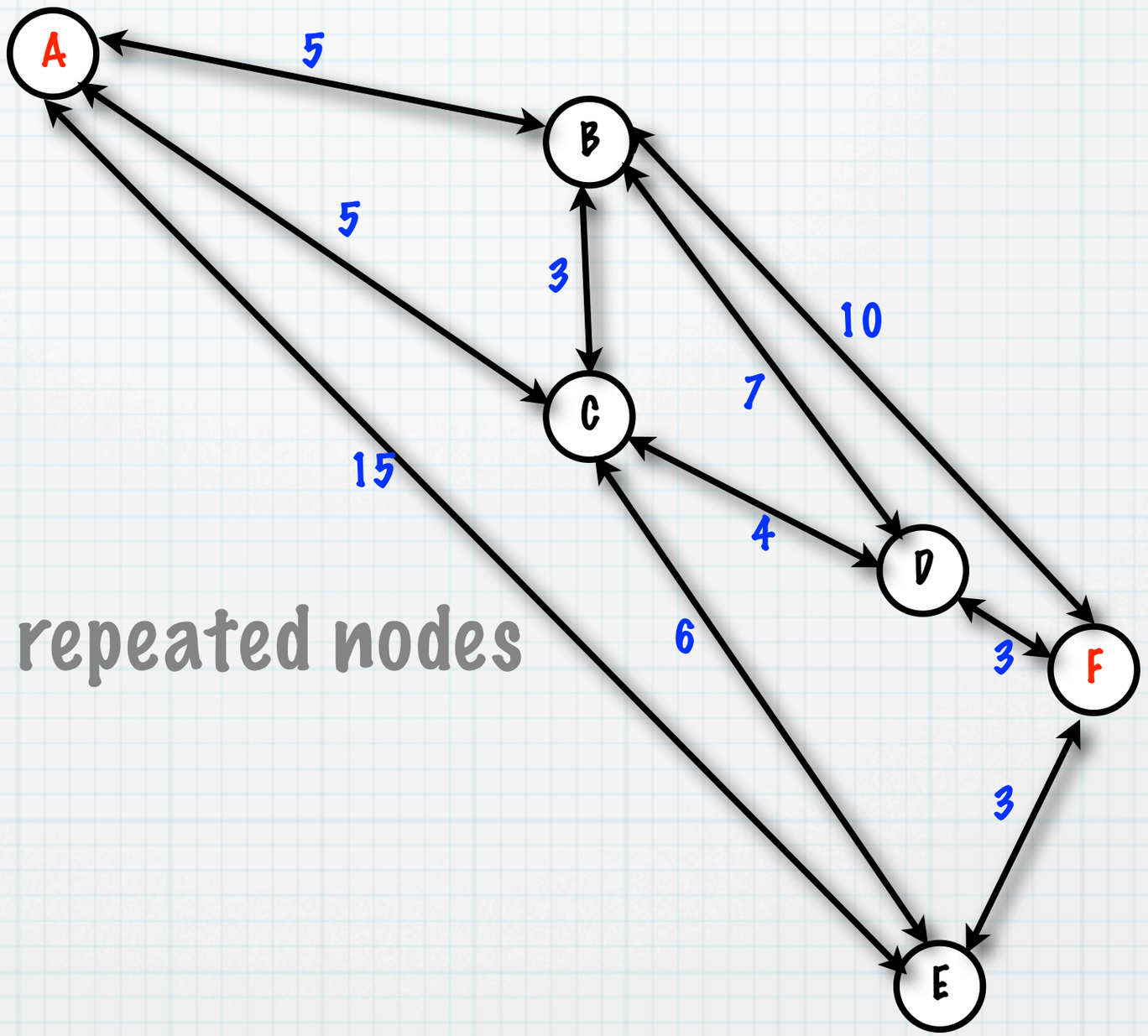


Graph

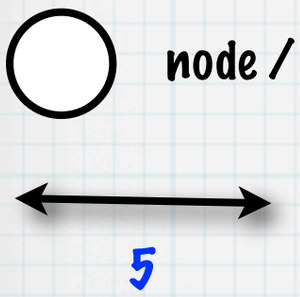


cycle: closed path

elementary cycle: no repeated nodes



Graph

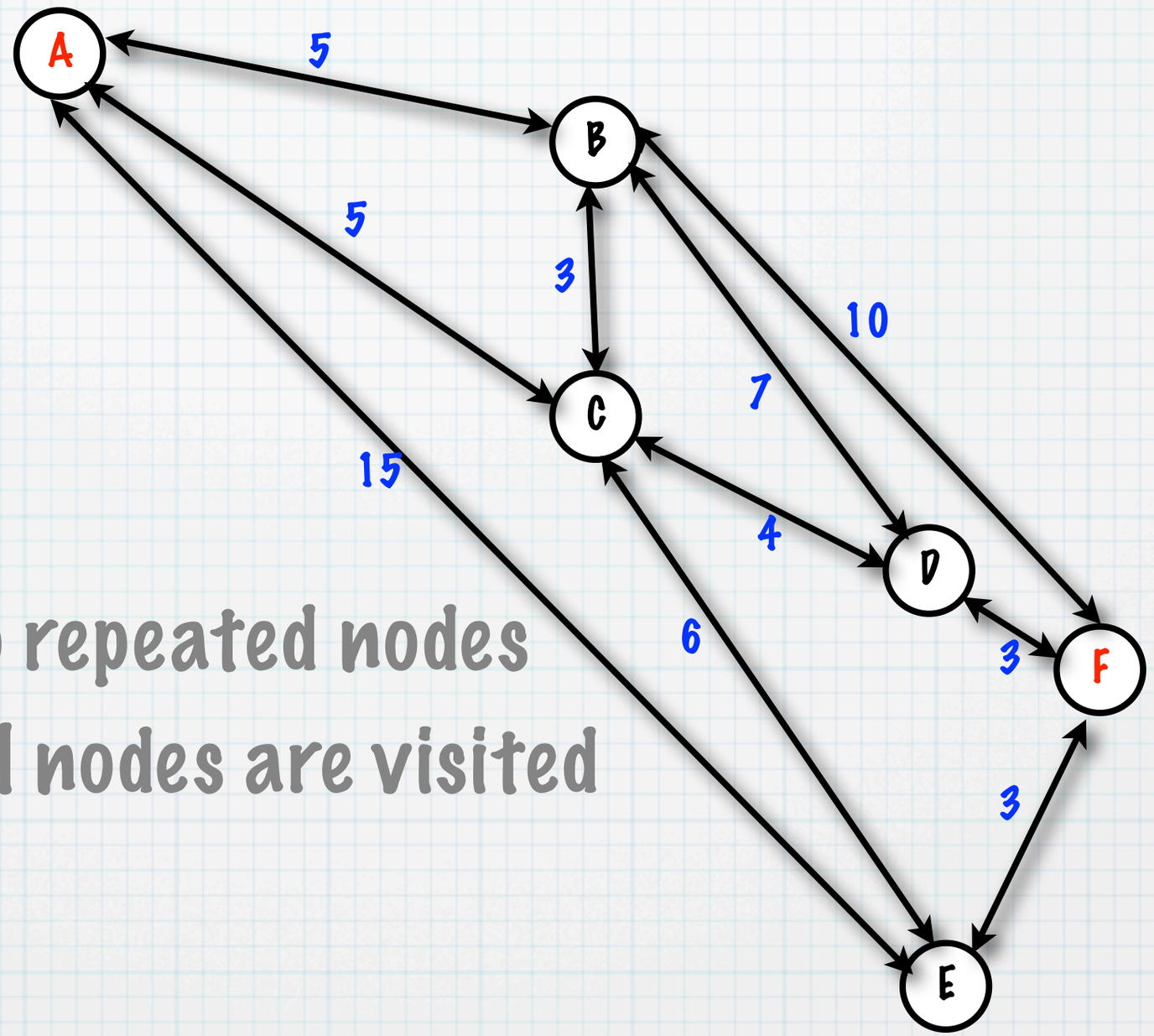


arc / edge
length

cycle: closed path

elementary cycle: no repeated nodes

Hamiltonian cycle: all nodes are visited



Graph

node / vertex

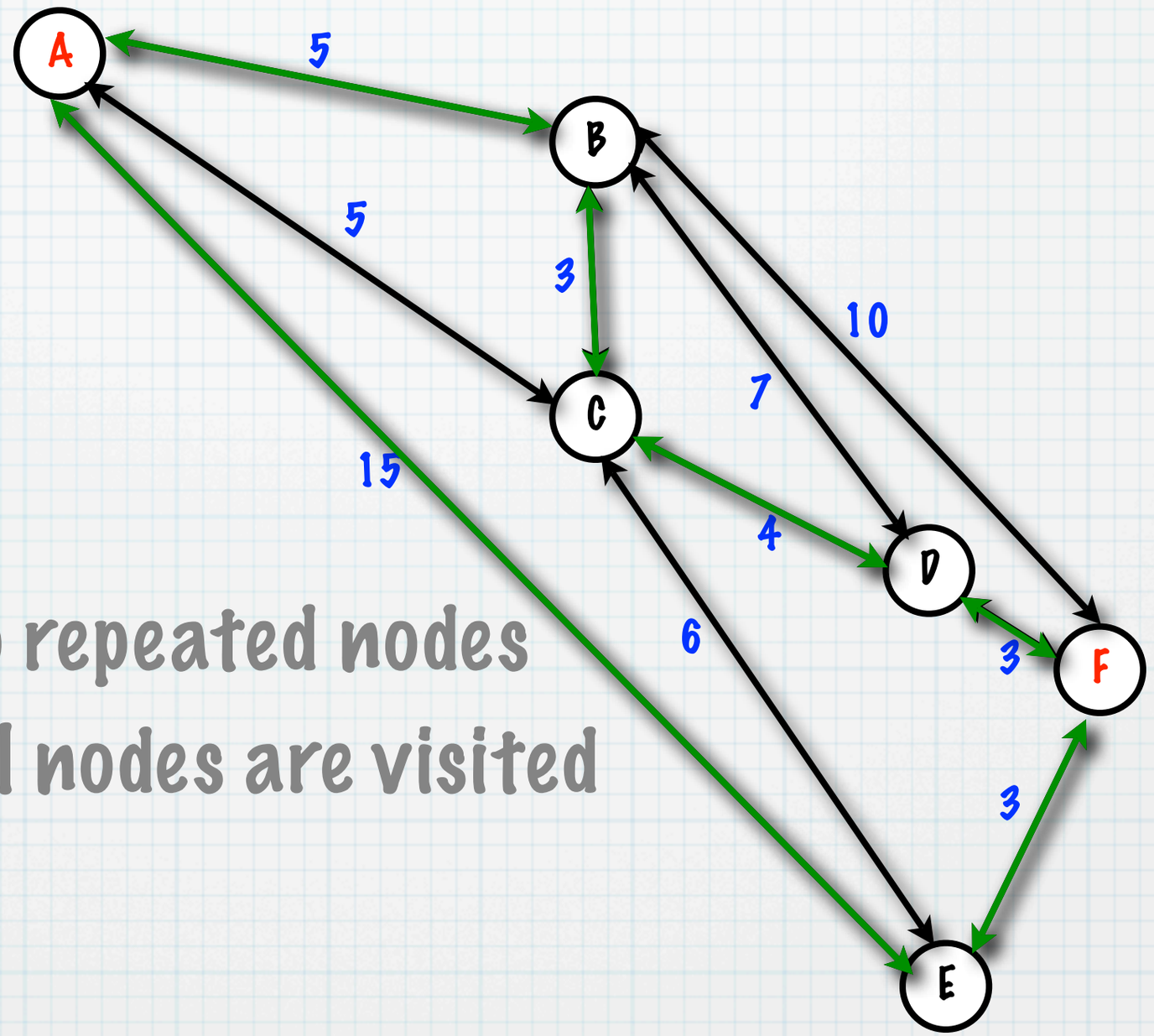
5

arc / edge
length

cycle: closed path

elementary cycle: no repeated nodes

Hamiltonian cycle: all nodes are visited



Graph

node / vertex

5

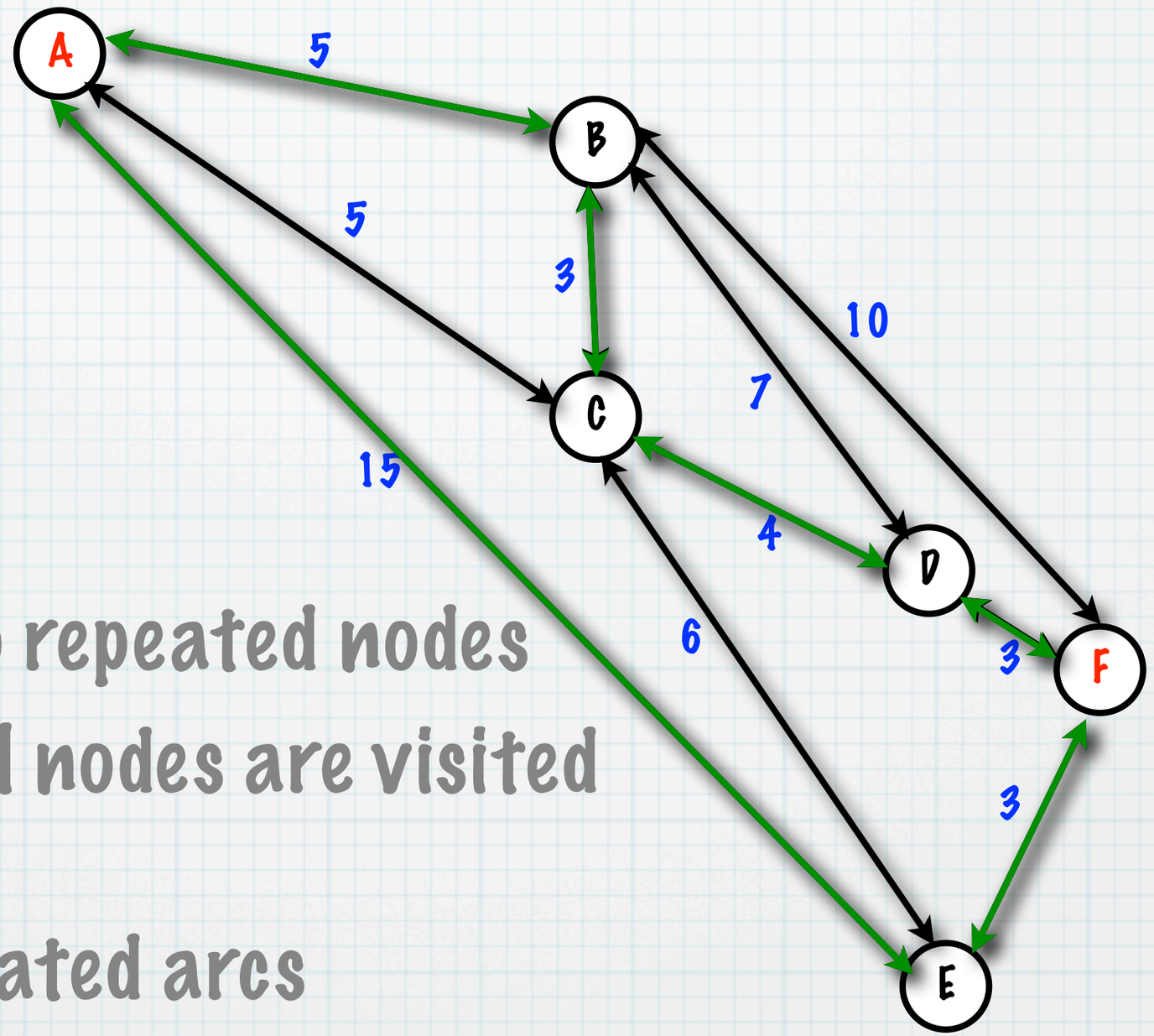
arc / edge
length

cycle: closed path

elementary cycle: no repeated nodes

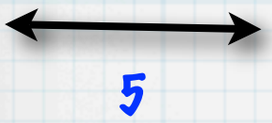
Hamiltonian cycle: all nodes are visited

simple cycle: no repeated arcs



Graph

node / vertex



arc / edge
length

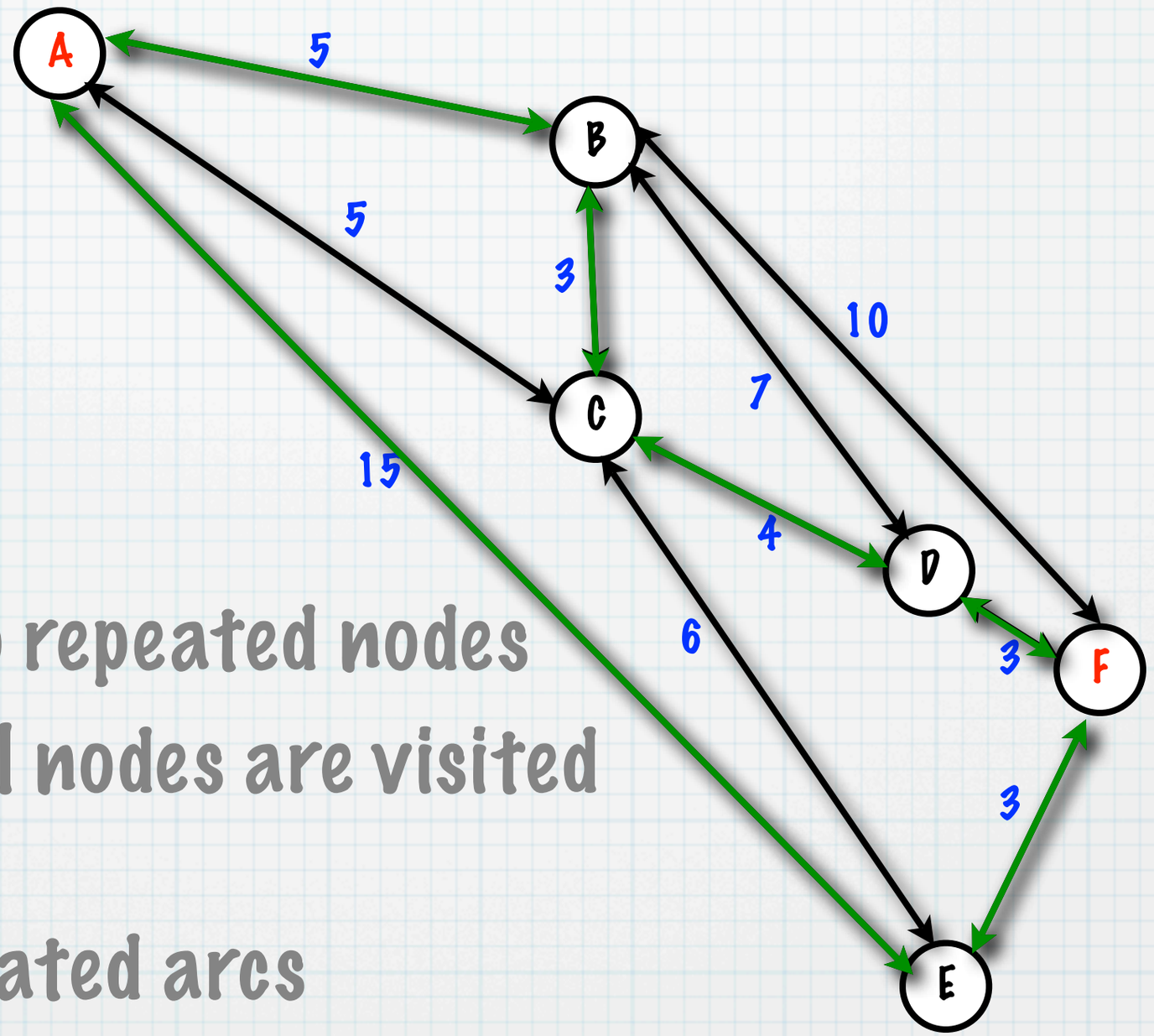
cycle: closed path

elementary cycle: no repeated nodes

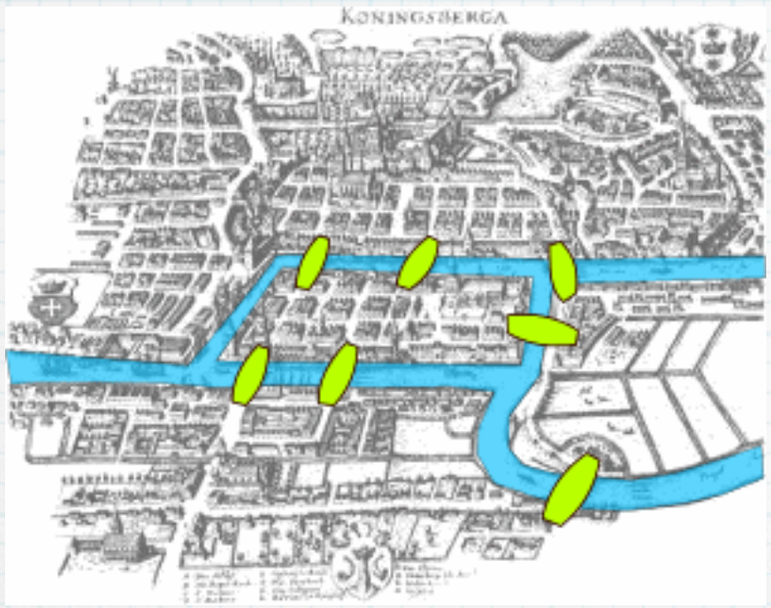
Hamiltonian cycle: all nodes are visited

simple cycle: no repeated arcs

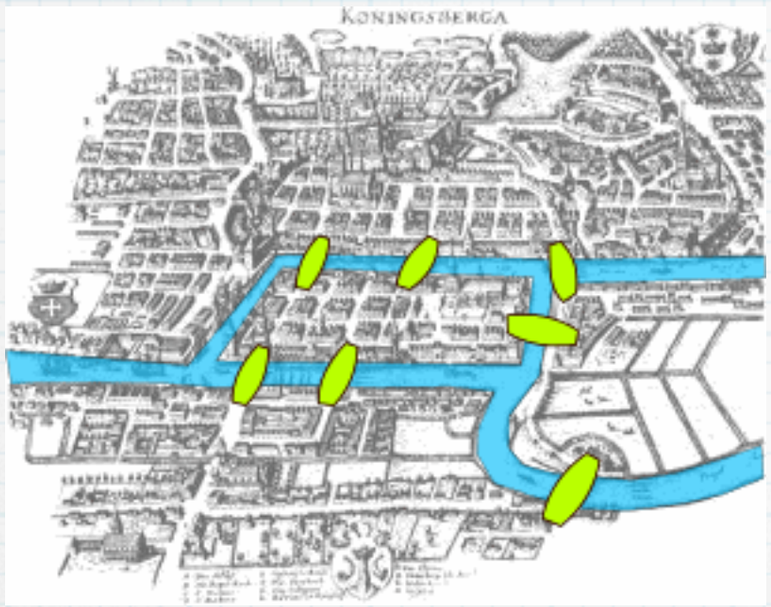
Eulerian tour: all arcs are visited

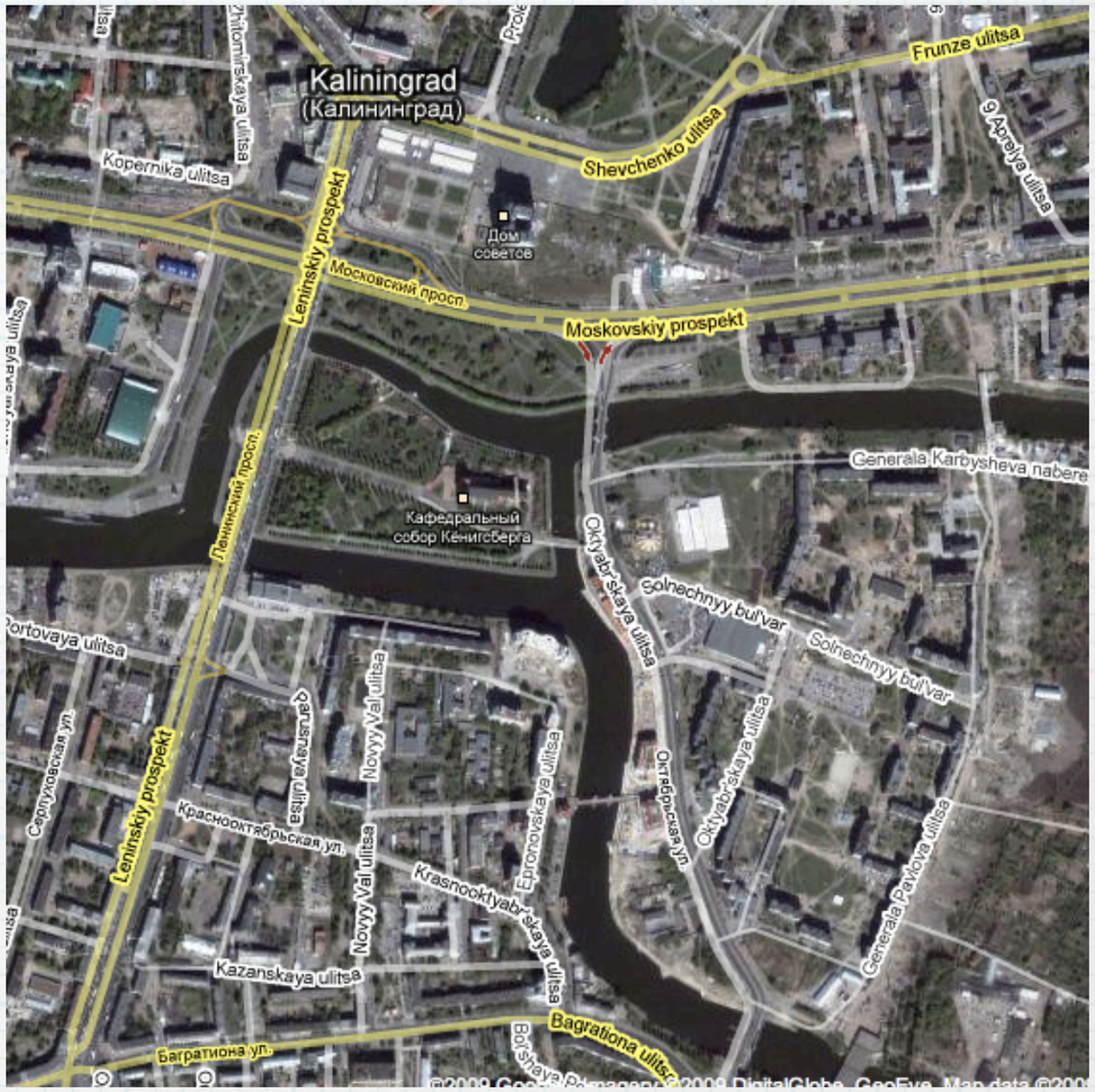


Curiosity: Königsberg bridges

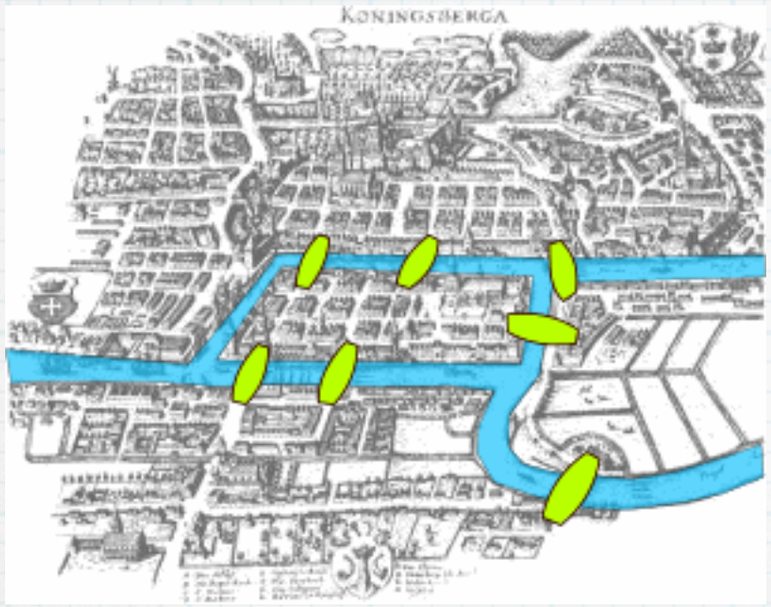


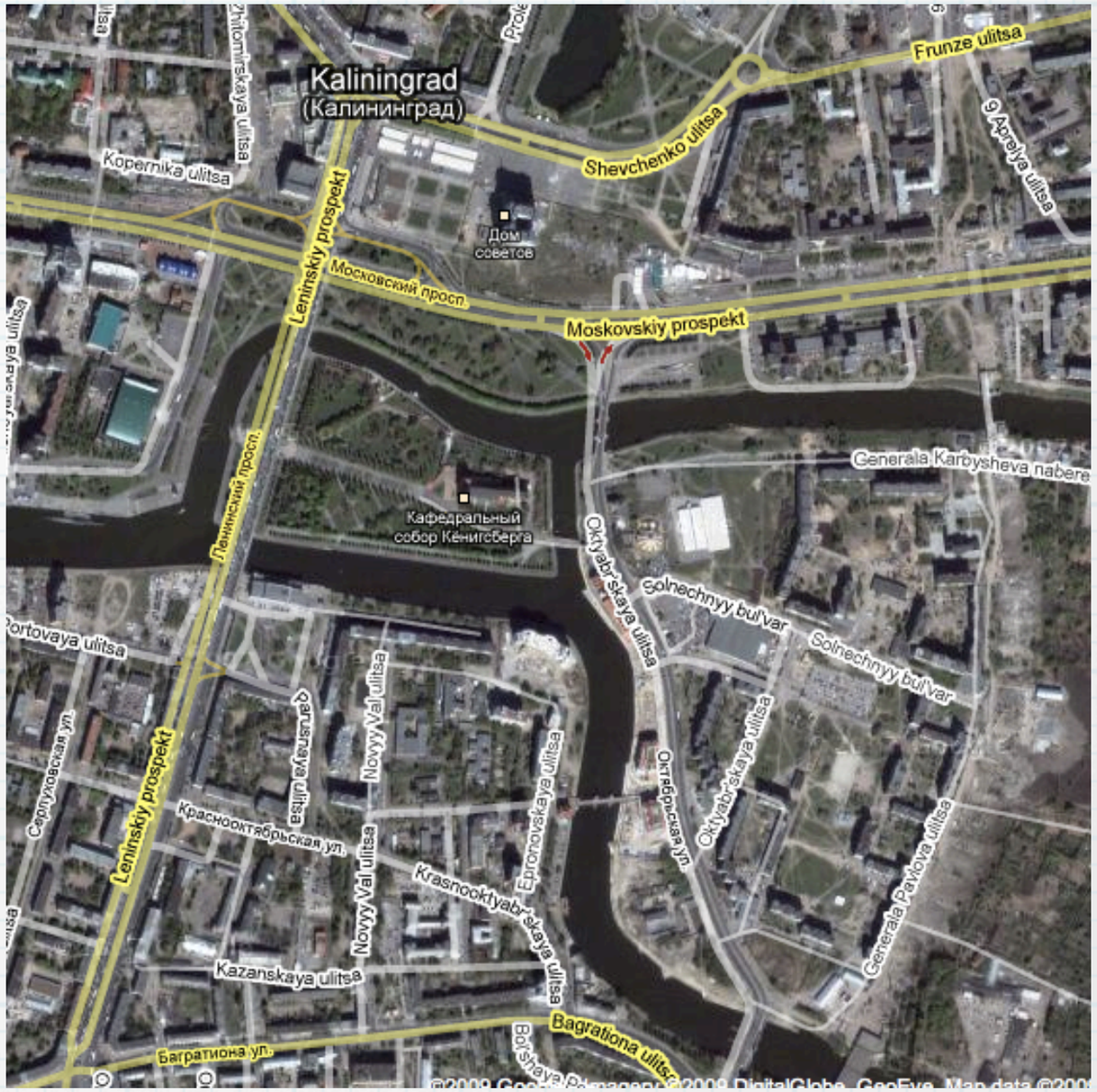
Curiosity: Königsberg bridges





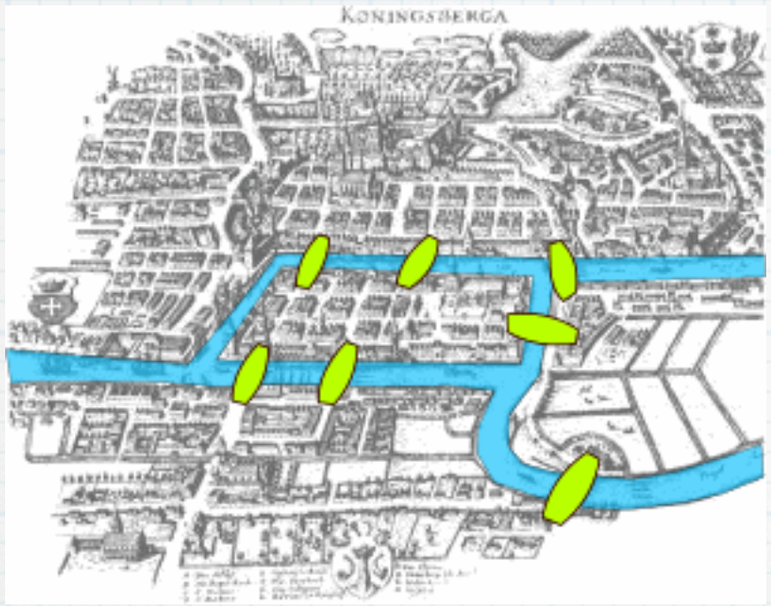
Curiosity: Königsberg bridges





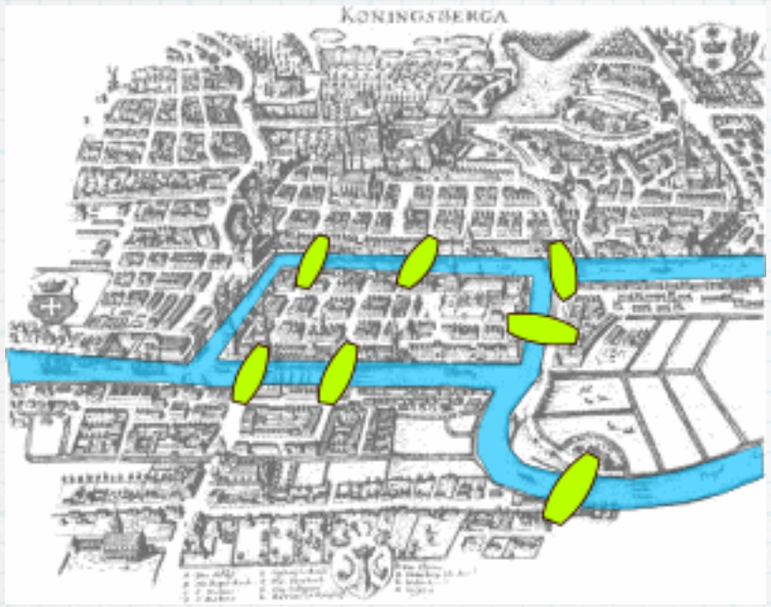
Is it possible to walk over the seven bridges and end up in the starting place?

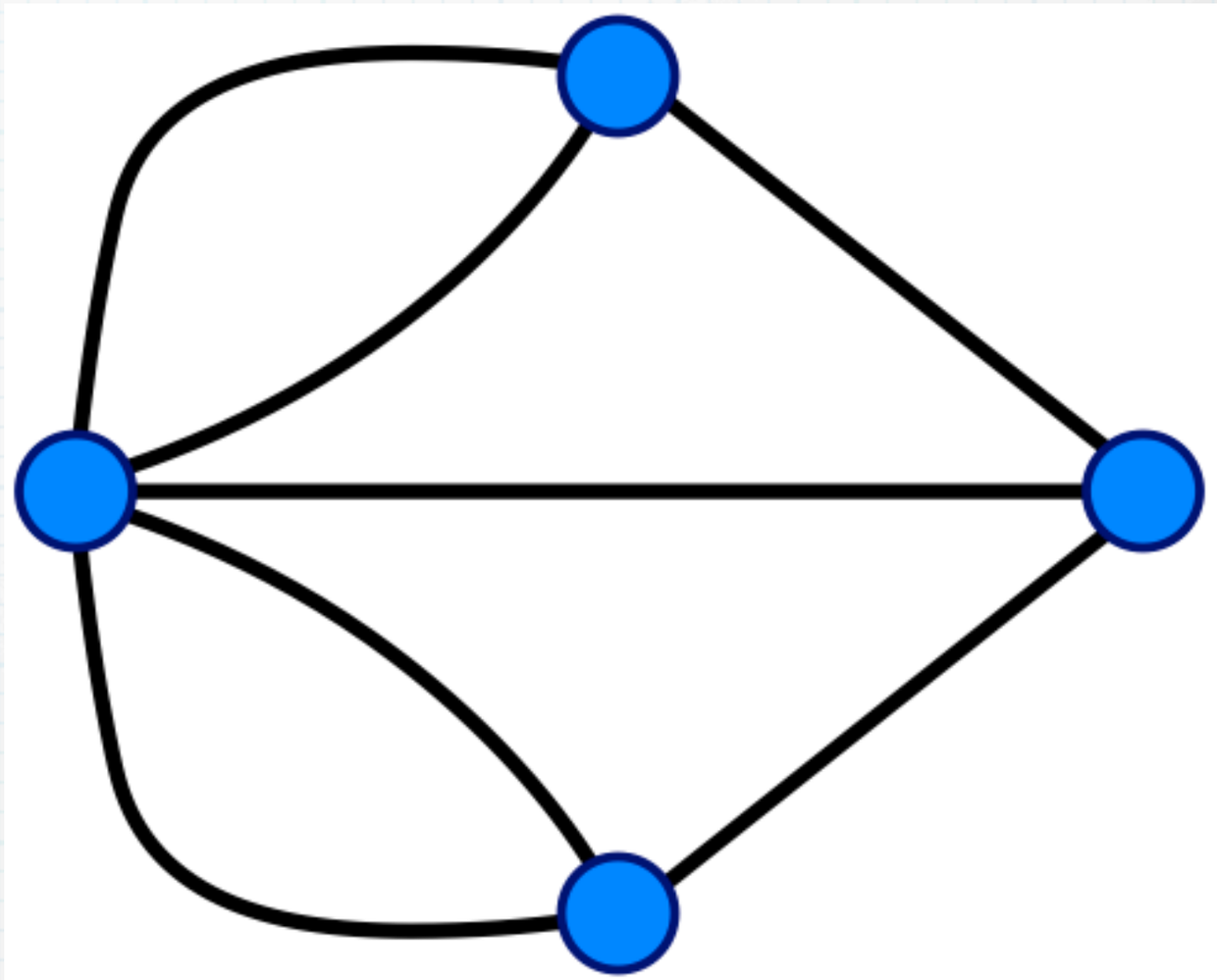
Curiosity: Königsberg bridges



Is it possible to walk over the seven bridges and end up in the starting place?

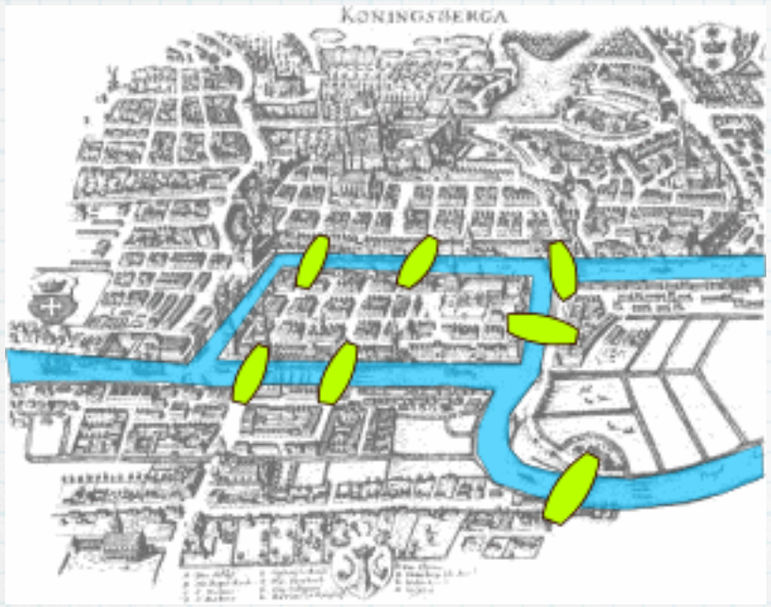
Curiosity: Königsberg bridges

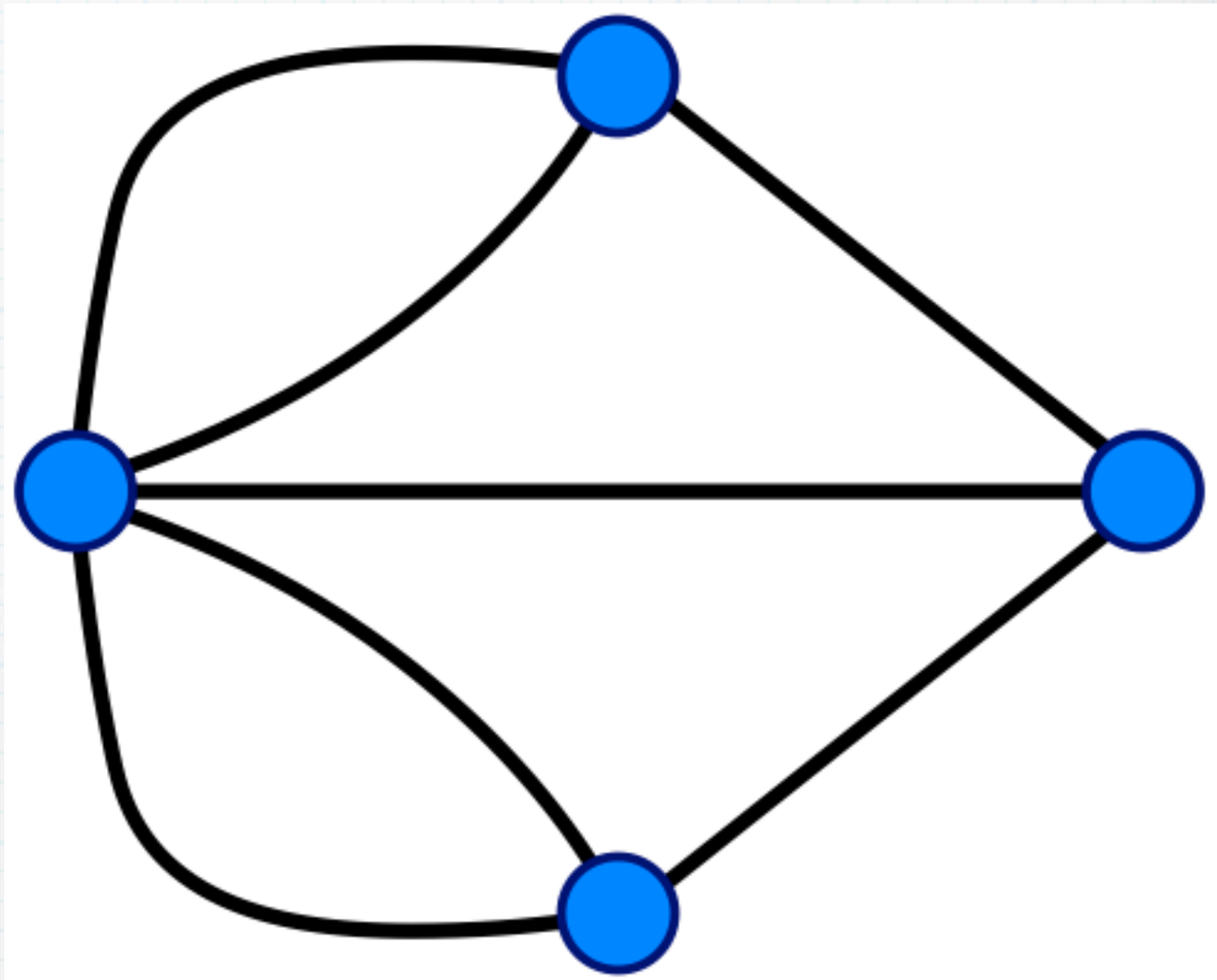




Is it possible to walk over the seven bridges and end up in the starting place?

Curiosity: Königsberg bridges

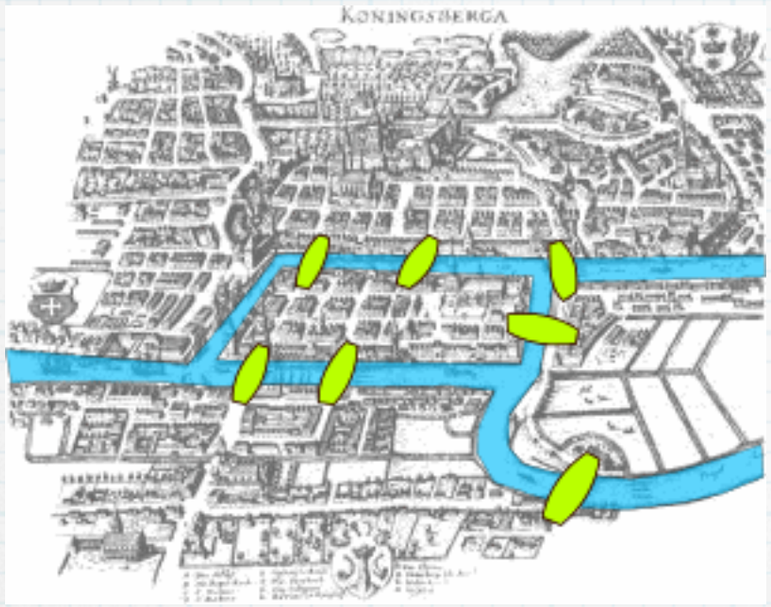


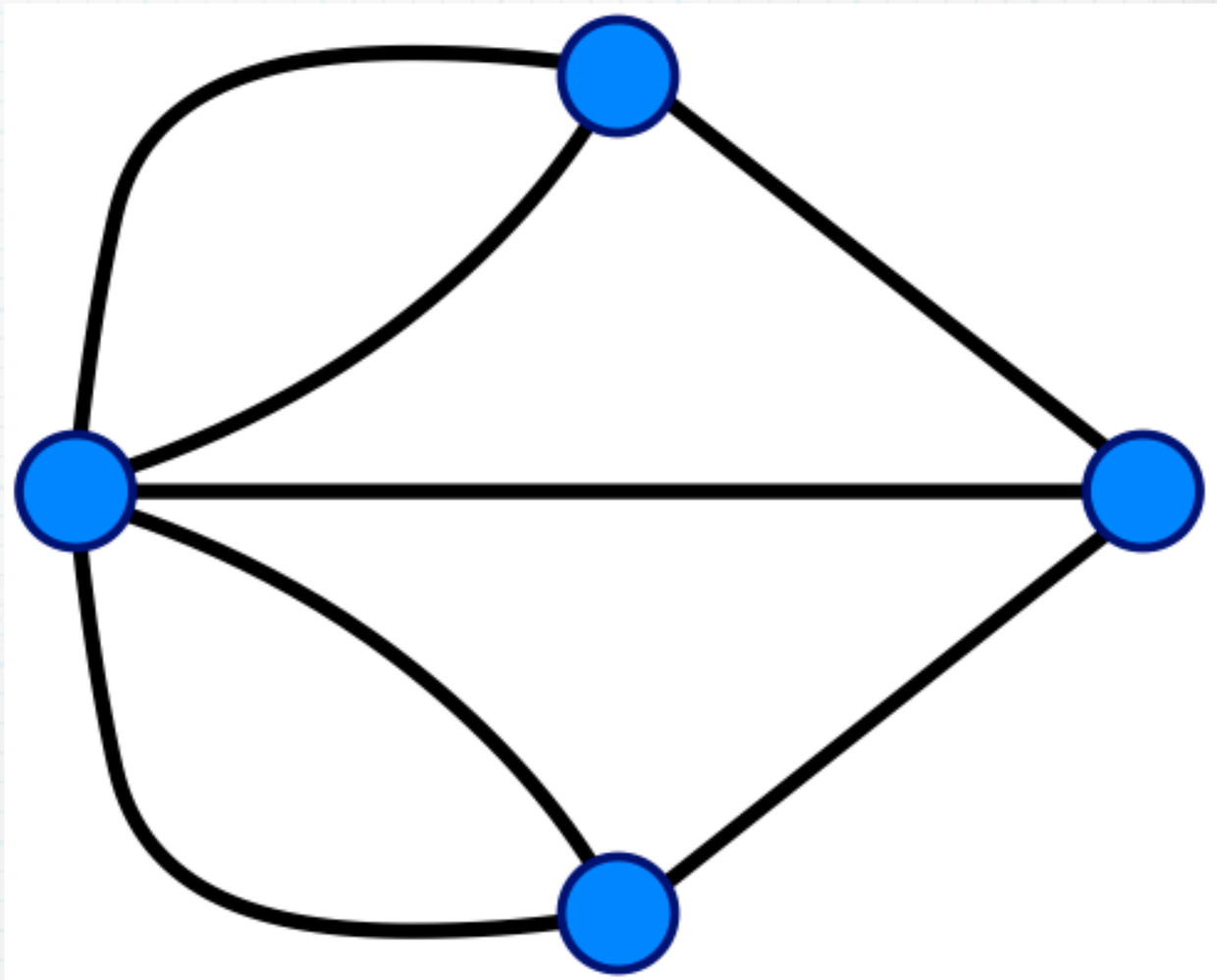


Is it possible to walk over the seven bridges and end up in the starting place?

Is the graph **Eulerian**?

Curiosity: Königsberg bridges



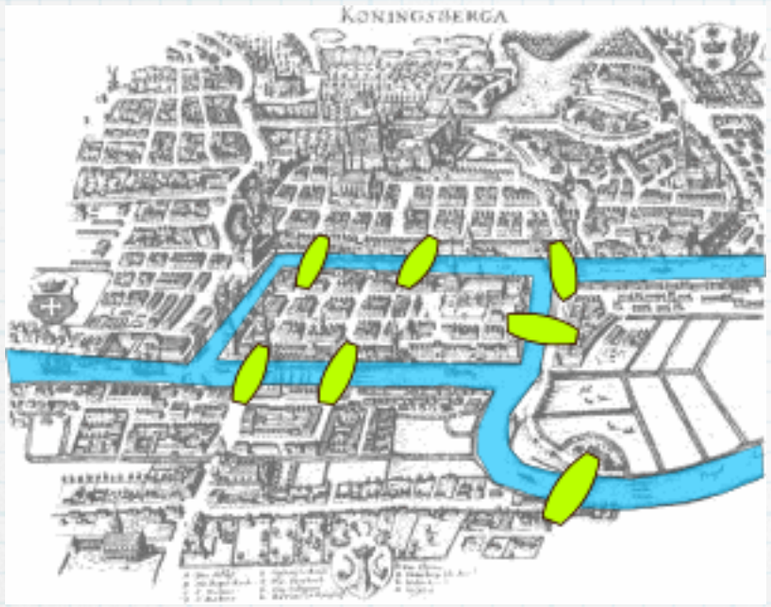


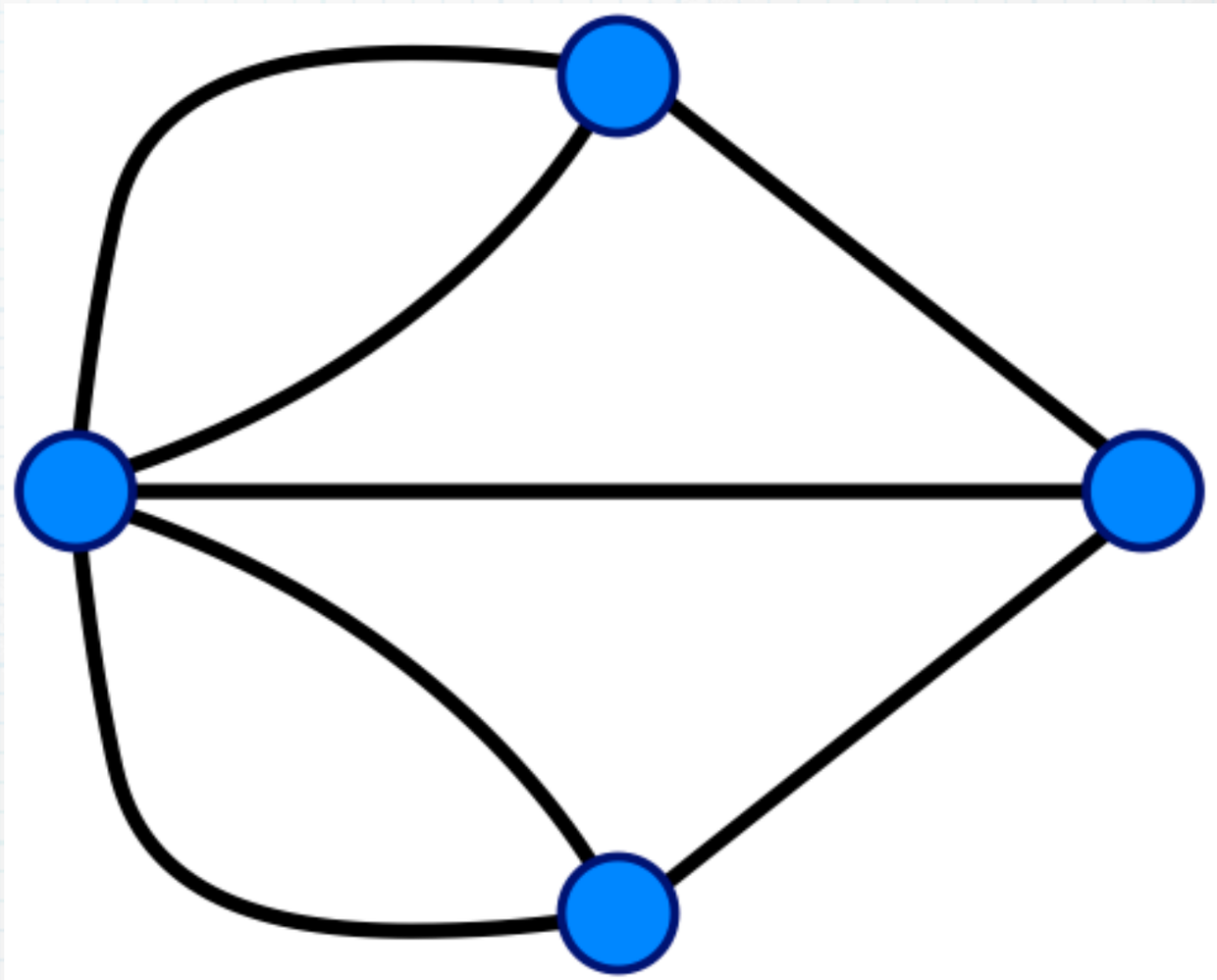
Is it possible to walk over the seven bridges and end up in the starting place?

Is the graph **Eulerian**?

Necessary and sufficient condition on **node degrees** (number of incident arcs)

Curiosity: Königsberg bridges





Is it possible to walk over the seven bridges and end up in the starting place?

Is the graph **Eulerian**?

Necessary and sufficient condition on **node degrees** (number of incident arcs)

Starting milestone of graph theory

Short movie on graph modeling

http://www.dfg-science-tv.de/en/projects/discrete-optimisers/2009-08-06

Undirected vs. directed graphs

Undirected vs. directed graphs

$$G = (N, A)$$

N Set of nodes

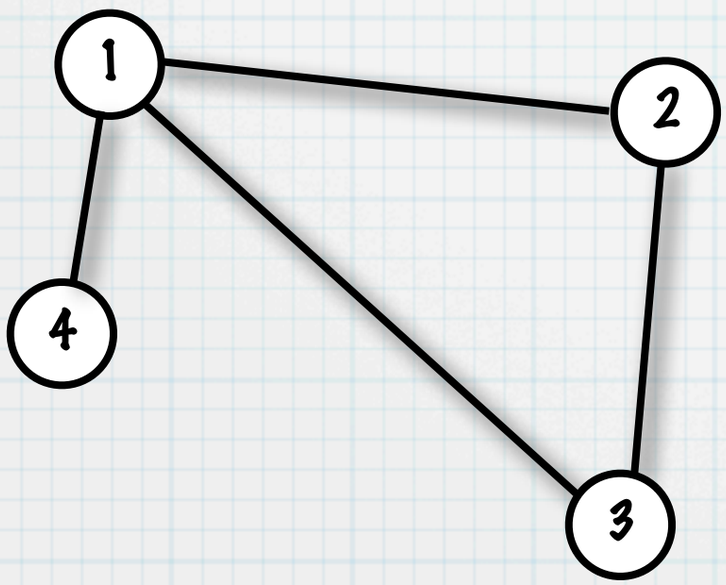
A Set of arcs (pairs of nodes)

Undirected vs. directed graphs

$$G = (N, A)$$

N Set of nodes

A Set of arcs (pairs of nodes)



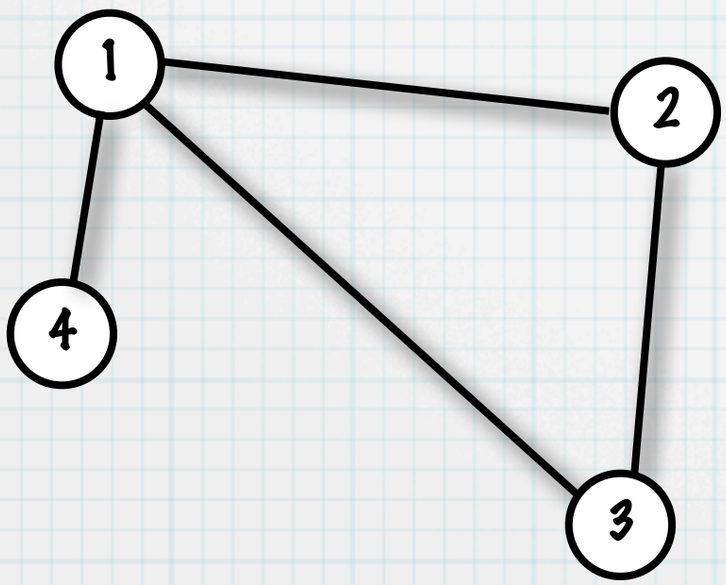
Undirected vs. directed graphs

$$G = (N, A)$$

N Set of nodes

A Set of arcs (pairs of nodes)

arcs = undirected pairs of nodes



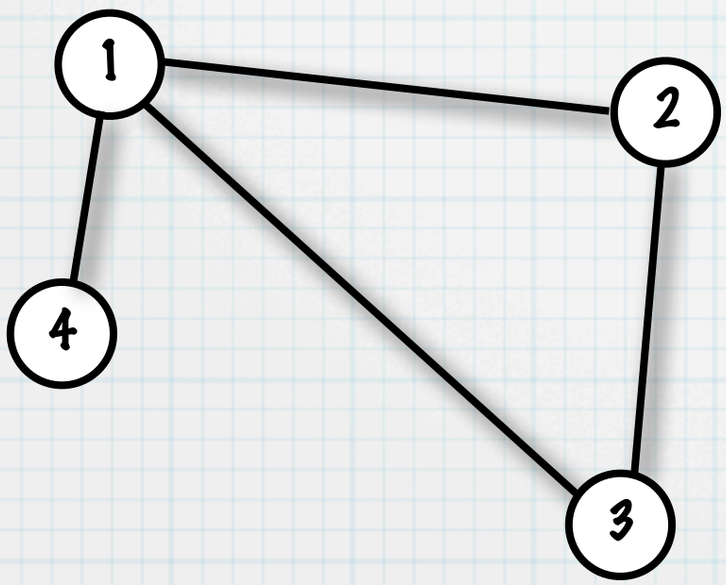
Undirected vs. directed graphs

$$G = (N, A)$$

N Set of nodes

A Set of arcs (pairs of nodes)

arcs = undirected pairs of nodes



$$\{i, j\} = \{j, i\}$$

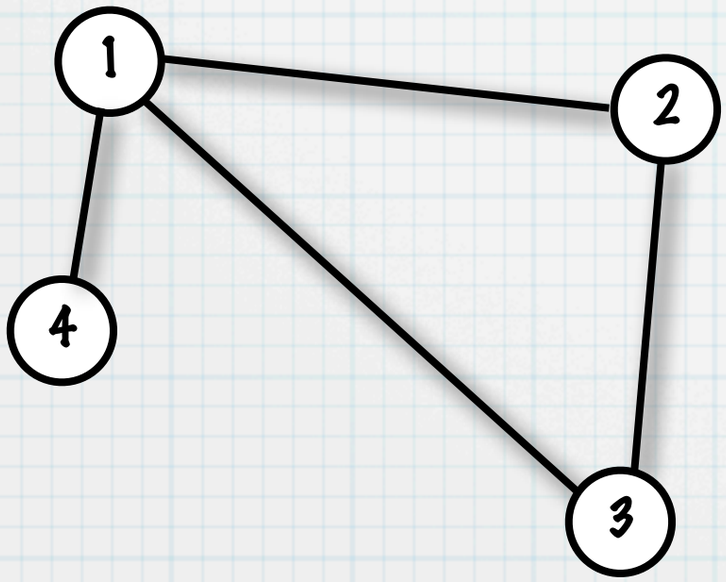
Undirected vs. directed graphs

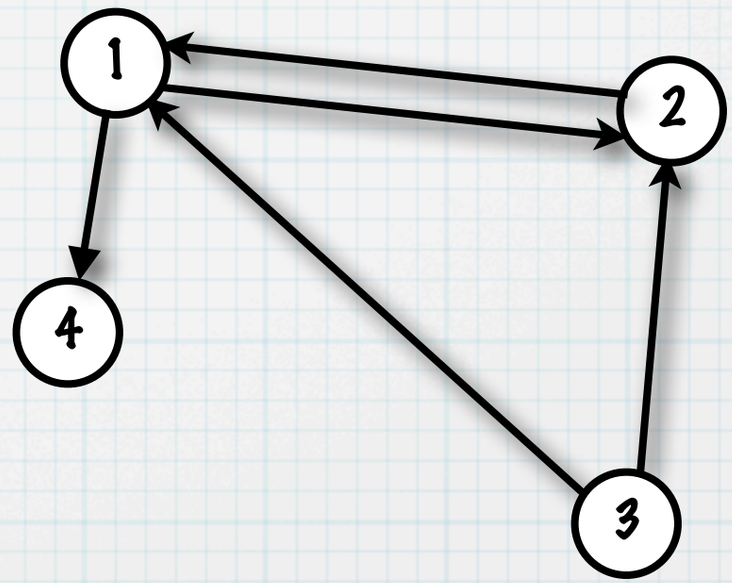
$$G = (N, A)$$

N Set of nodes

A Set of arcs (pairs of nodes)

arcs = undirected pairs of nodes





$\{i, j\} = \{j, i\}$

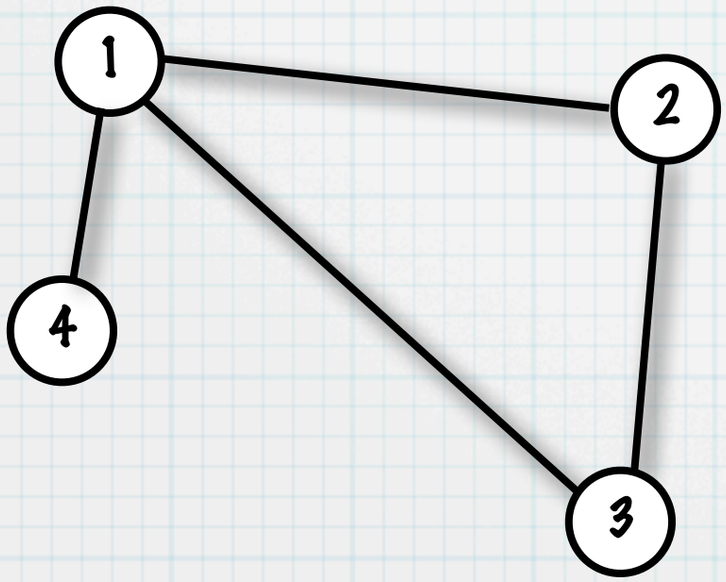
Undirected vs. directed graphs

$$G = (N, A)$$

N Set of nodes

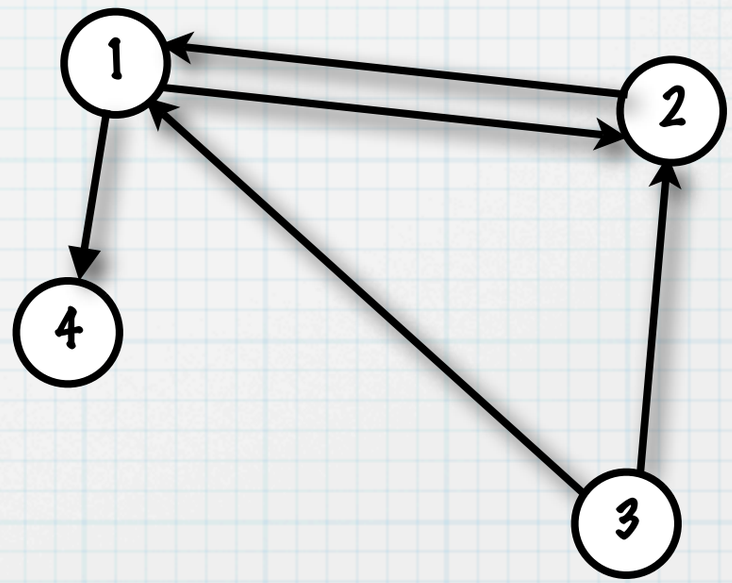
A Set of arcs (pairs of nodes)

arcs = undirected pairs of nodes



$\{i, j\} = \{j, i\}$

arcs = directed pairs of nodes



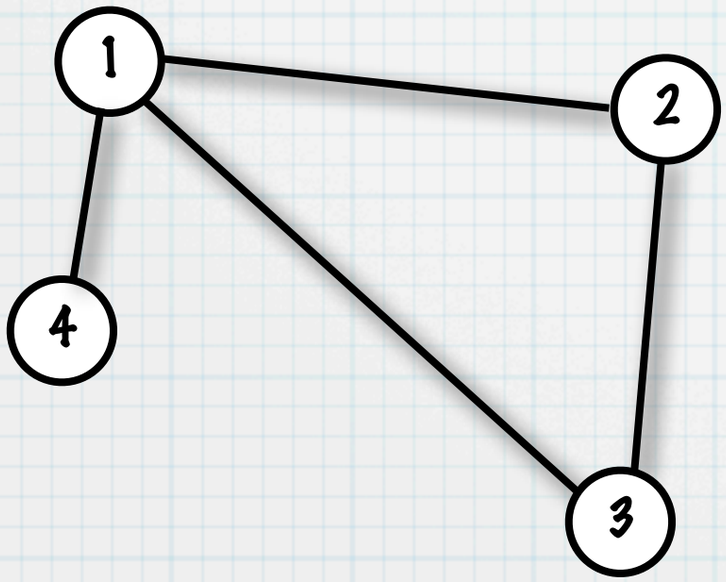
Undirected vs. directed graphs

$$G = (N, A)$$

N Set of nodes

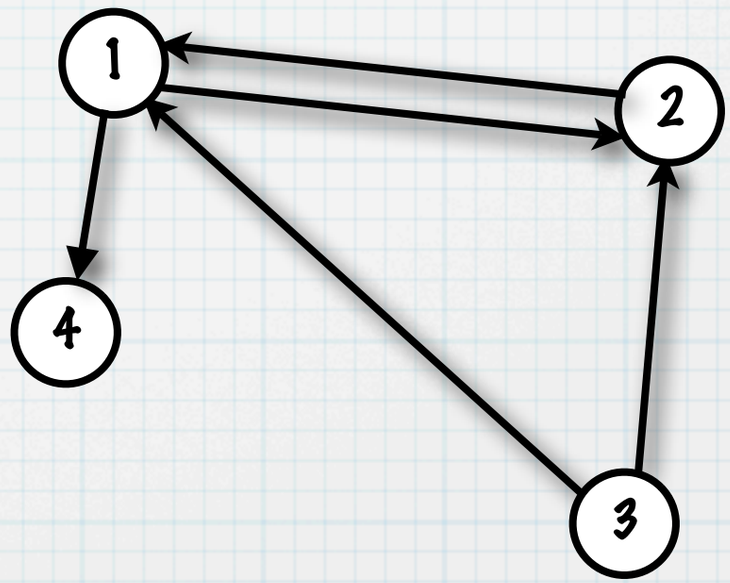
A Set of arcs (pairs of nodes)

arcs = undirected pairs of nodes



$$\{i, j\} = \{j, i\}$$

arcs = directed pairs of nodes



$$(i, j) \neq (j, i)$$

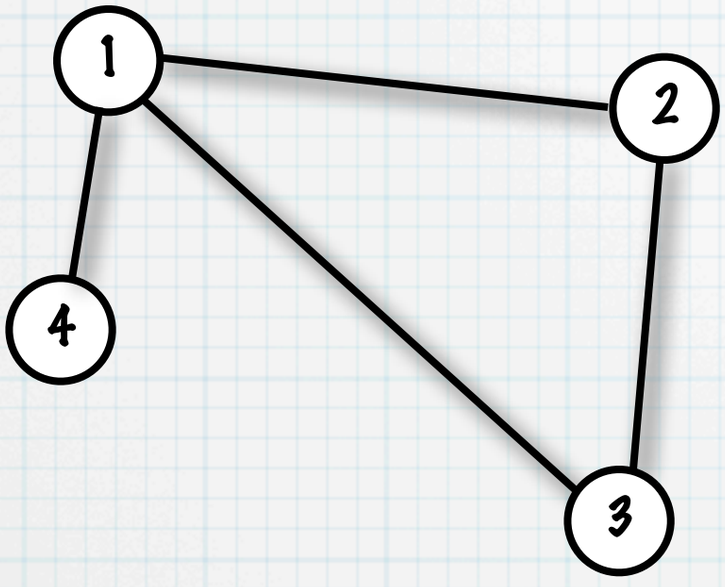
How to represent graphs (1)

How to represent graphs (1)

Adjacency matrix A

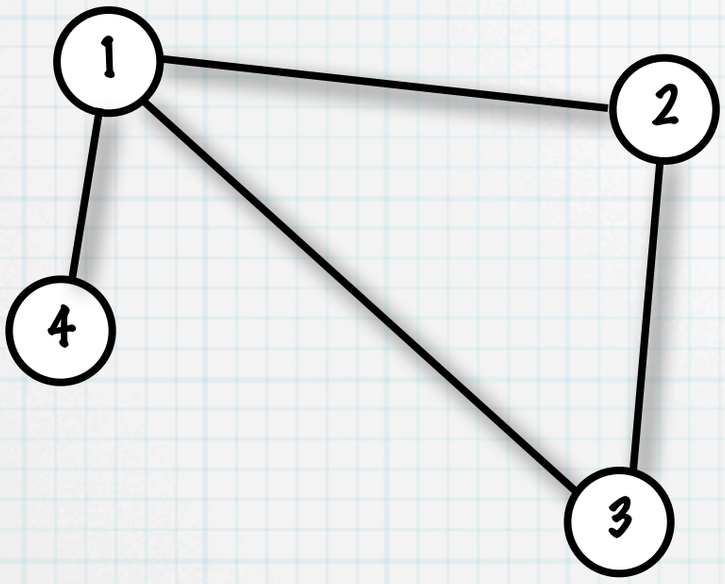
How to represent graphs (1)

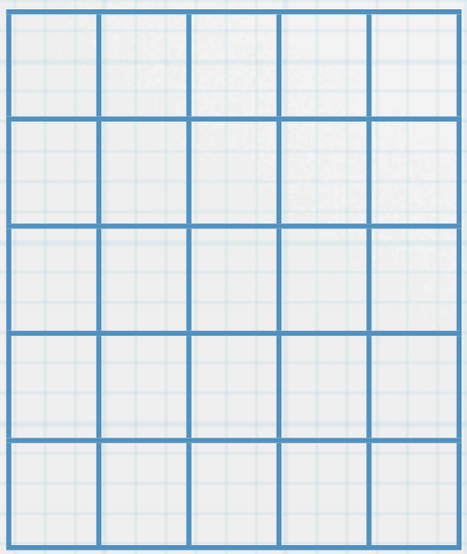
Adjacency matrix A



How to represent graphs (1)

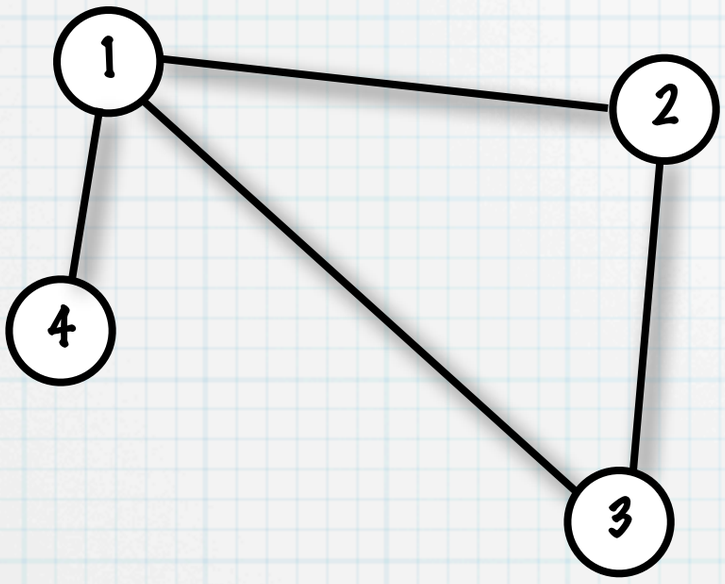
Adjacency matrix A

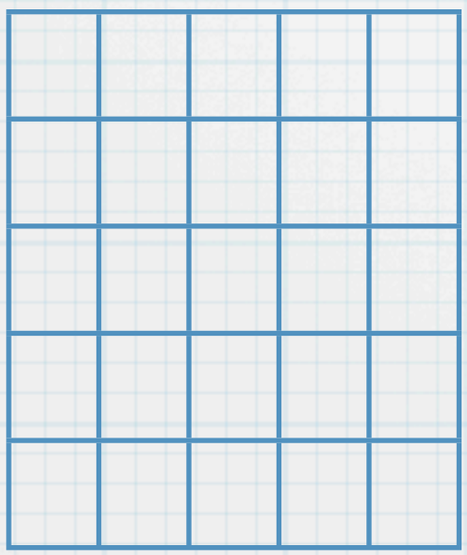




How to represent graphs (1)

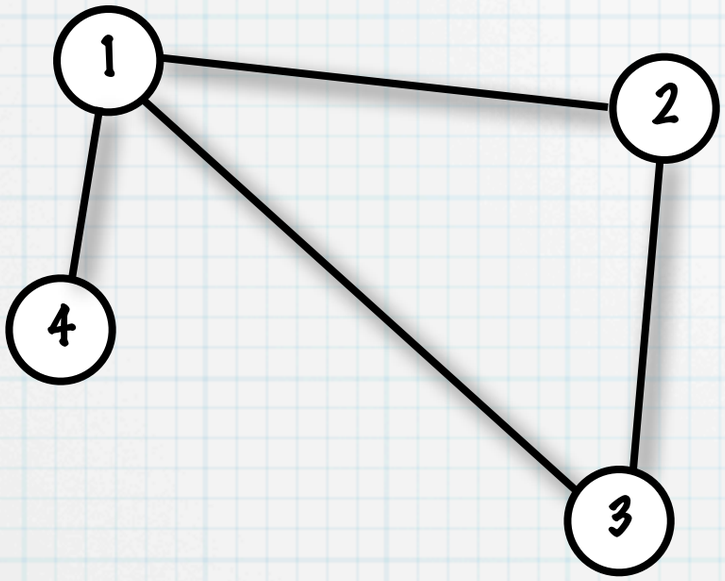
Adjacency matrix A





How to represent graphs (1)

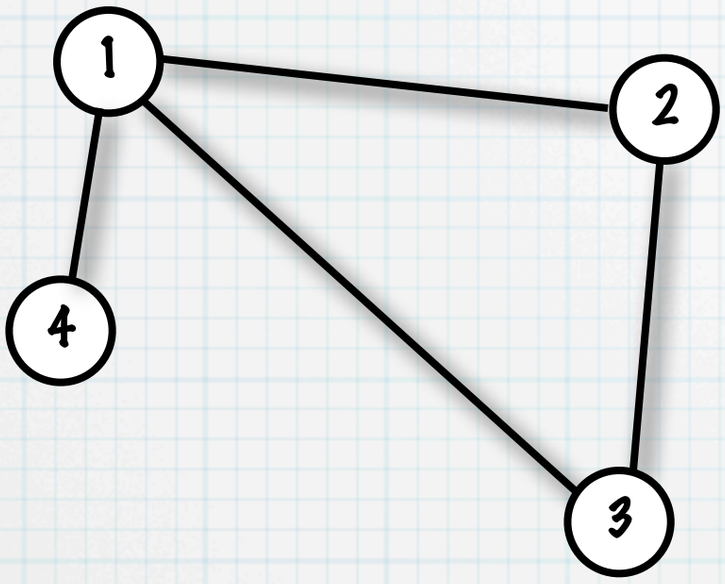
Adjacency matrix A



	1			

How to represent graphs (1)

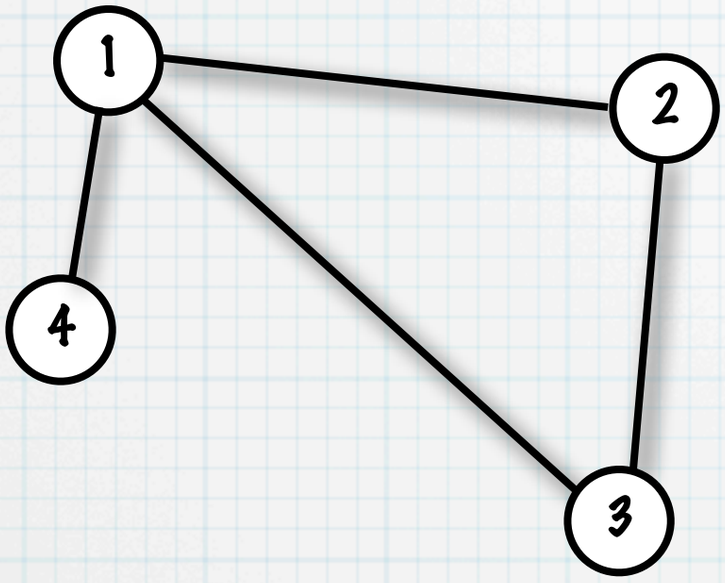
Adjacency matrix A



	1	2		

How to represent graphs (1)

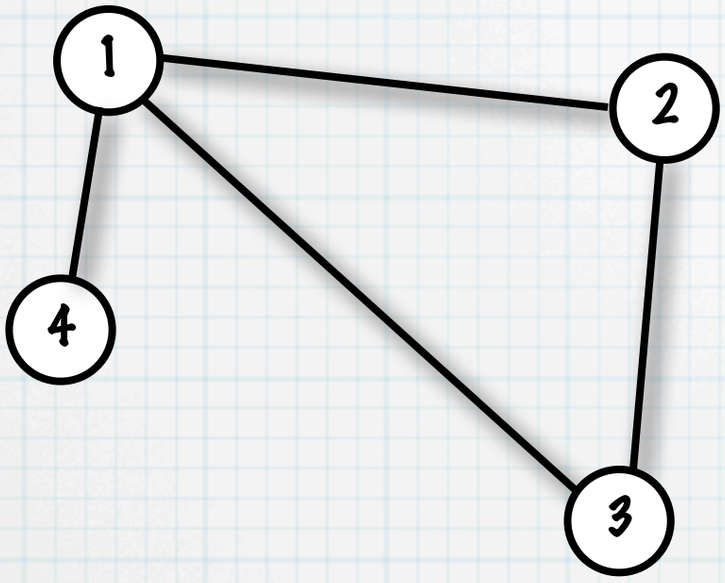
Adjacency matrix A



	1	2	3	

How to represent graphs (1)

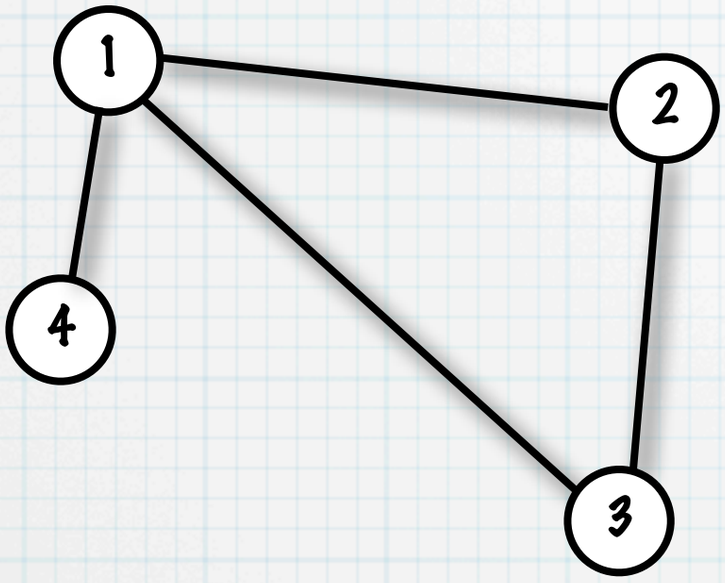
Adjacency matrix A



	1	2	3	4

How to represent graphs (1)

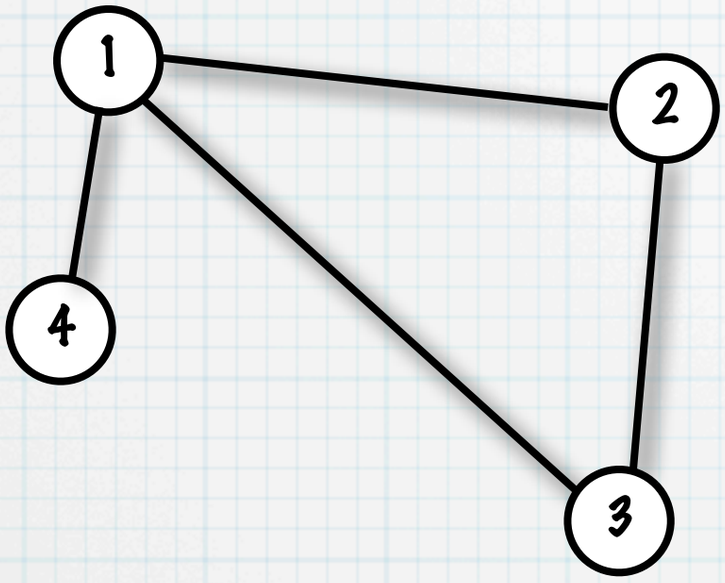
Adjacency matrix A



	1	2	3	4
1				

How to represent graphs (1)

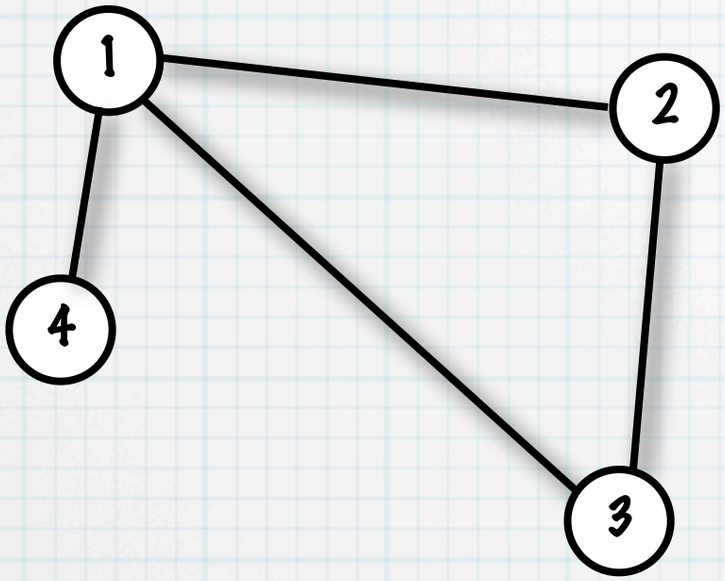
Adjacency matrix A



	1	2	3	4
1				

How to represent graphs (1)

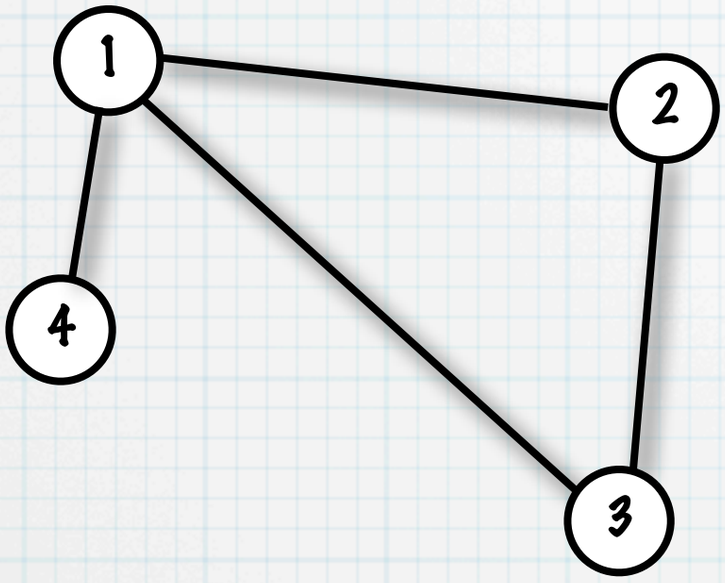
Adjacency matrix A



	1	2	3	4
1		1		

How to represent graphs (1)

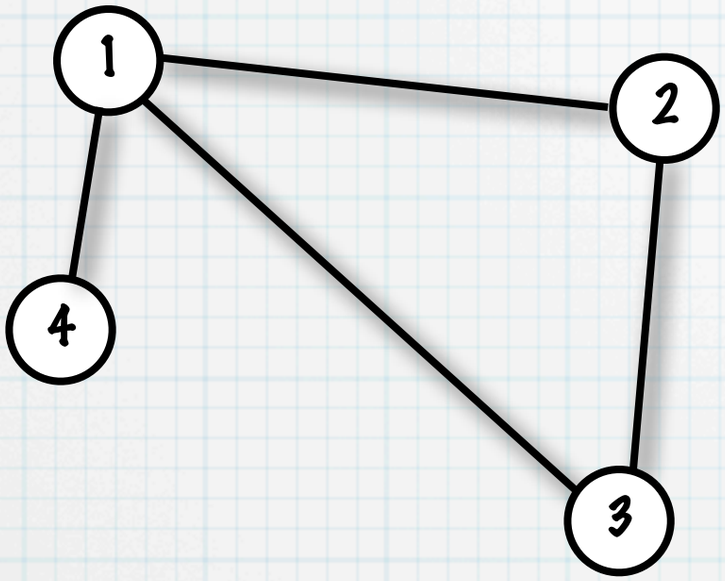
Adjacency matrix A



	1	2	3	4
1		1	1	

How to represent graphs (1)

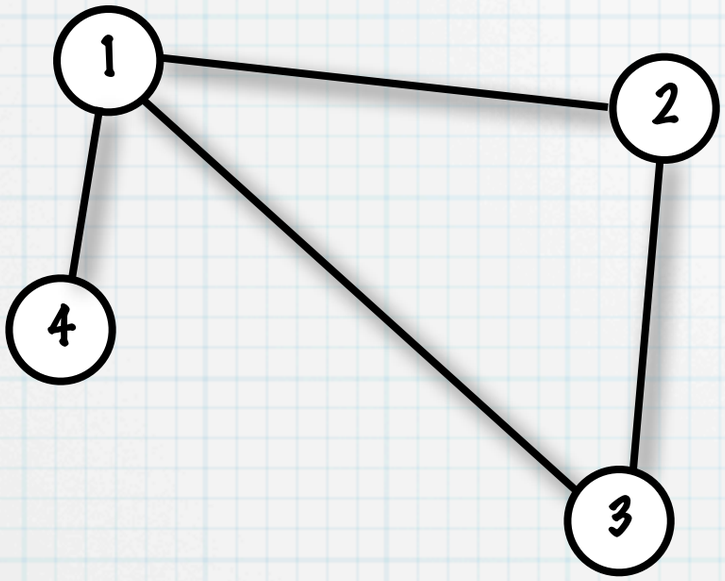
Adjacency matrix A



	1	2	3	4
1		1	1	1

How to represent graphs (1)

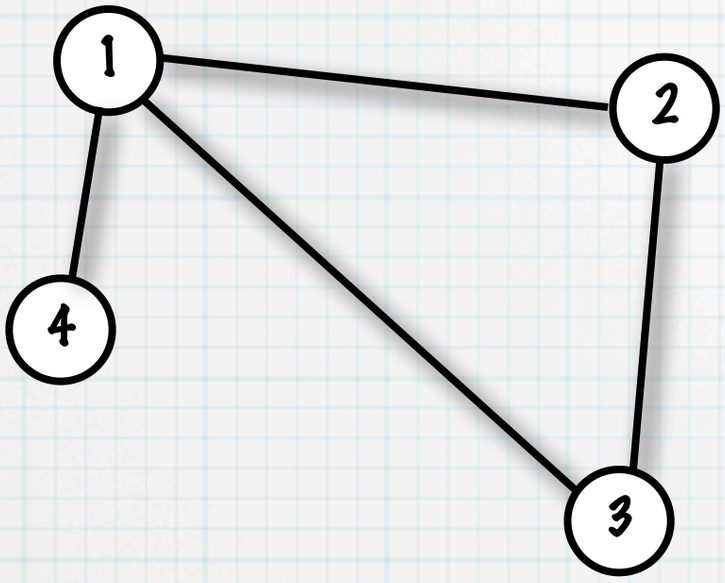
Adjacency matrix A



	1	2	3	4
1		1	1	1
2				

How to represent graphs (1)

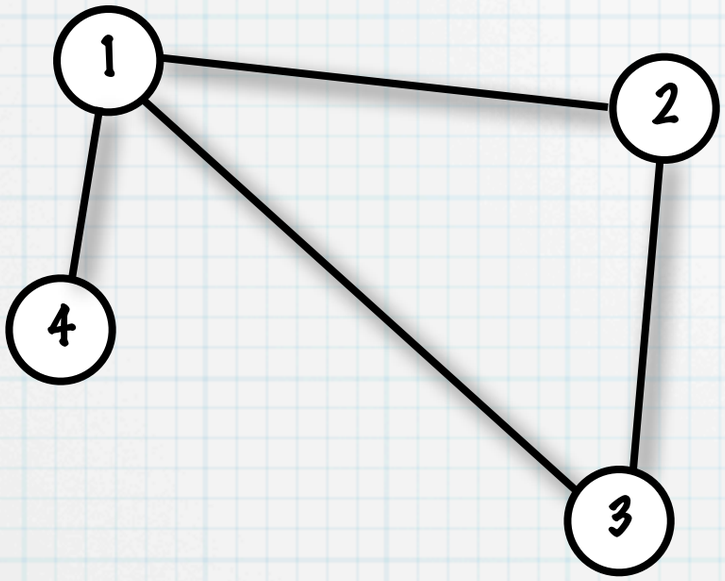
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1			

How to represent graphs (1)

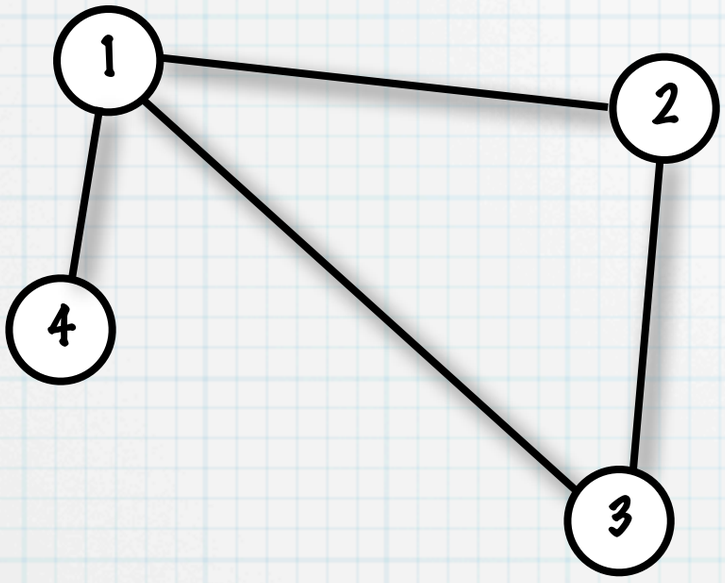
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1			

How to represent graphs (1)

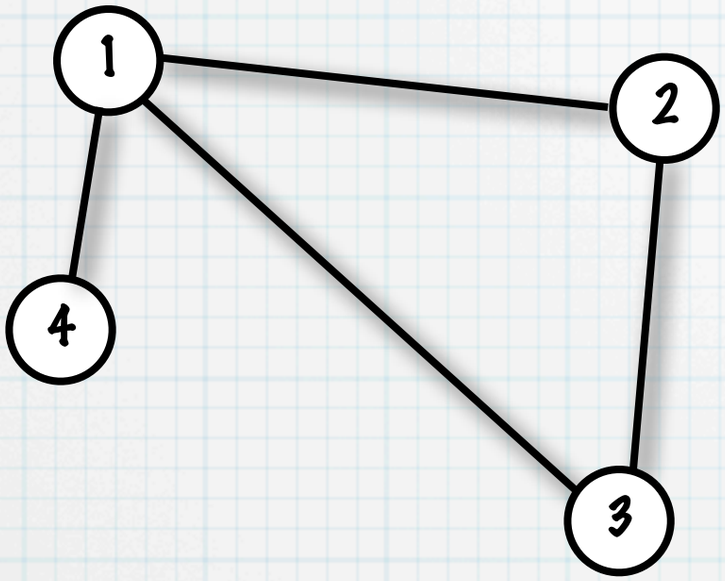
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	

How to represent graphs (1)

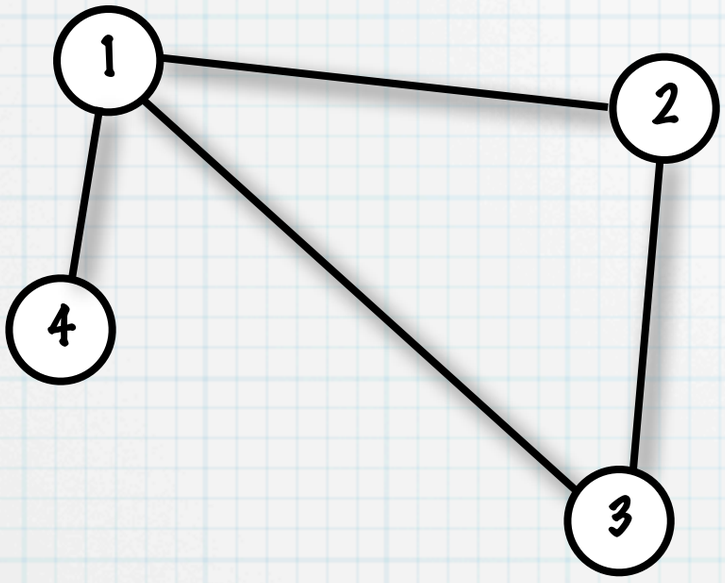
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0

How to represent graphs (1)

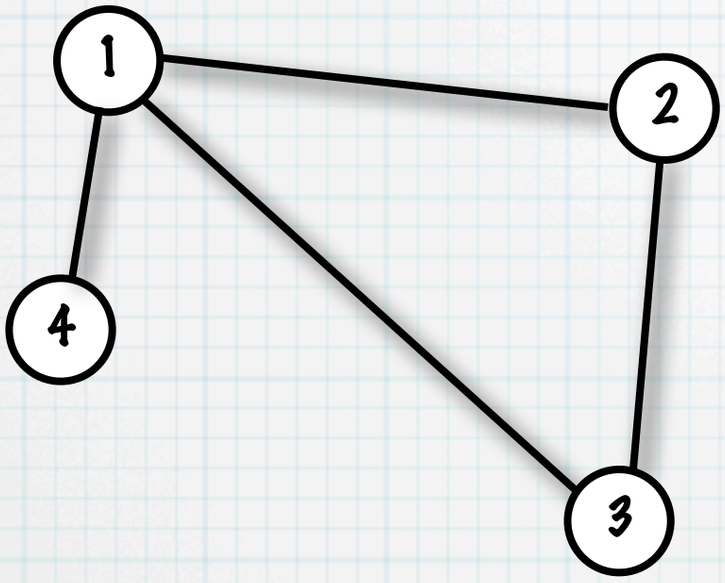
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3				

How to represent graphs (1)

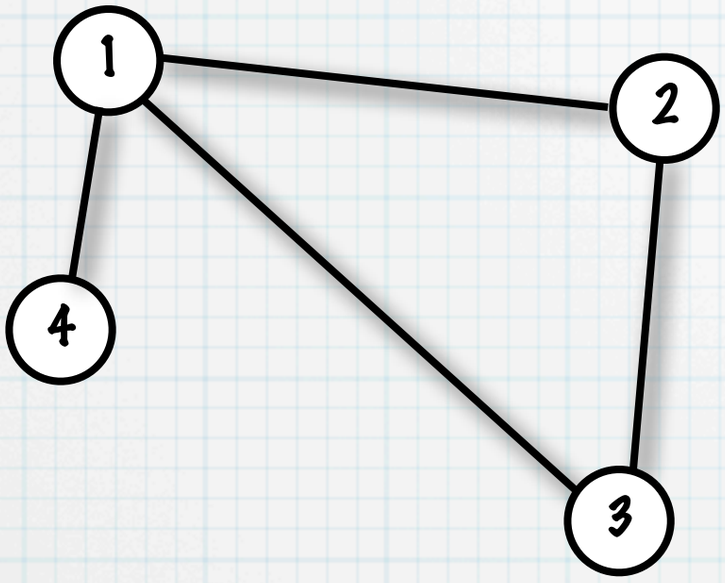
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1			

How to represent graphs (1)

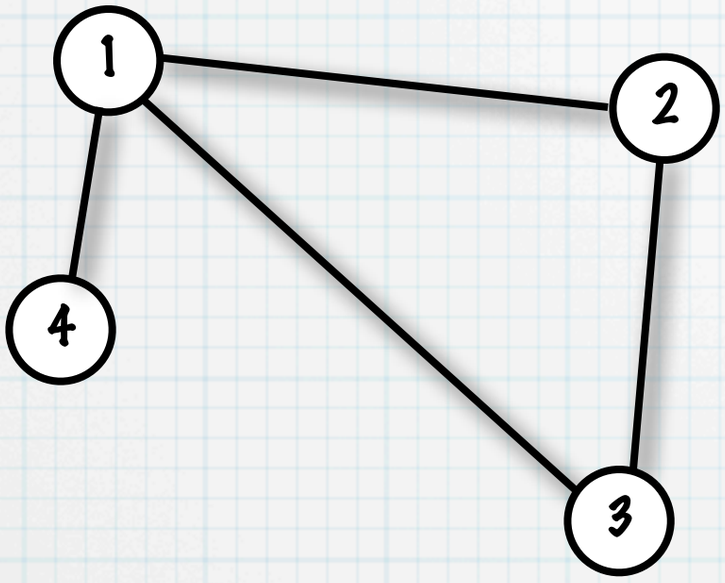
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		

How to represent graphs (1)

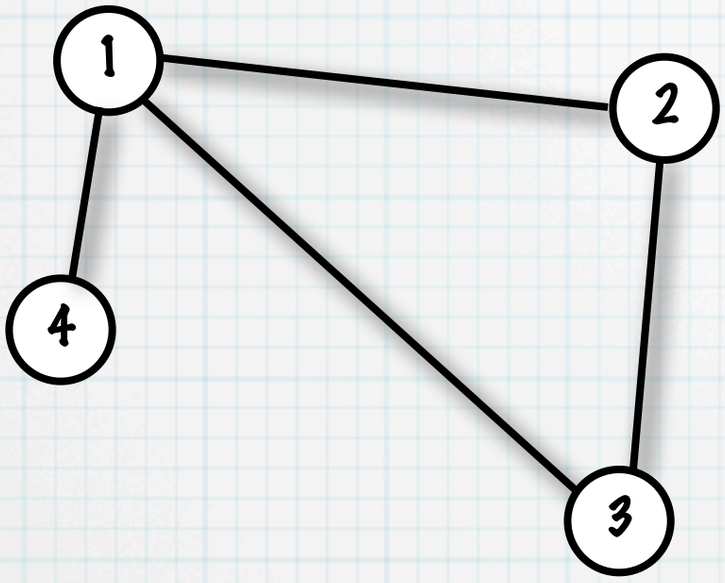
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		

How to represent graphs (1)

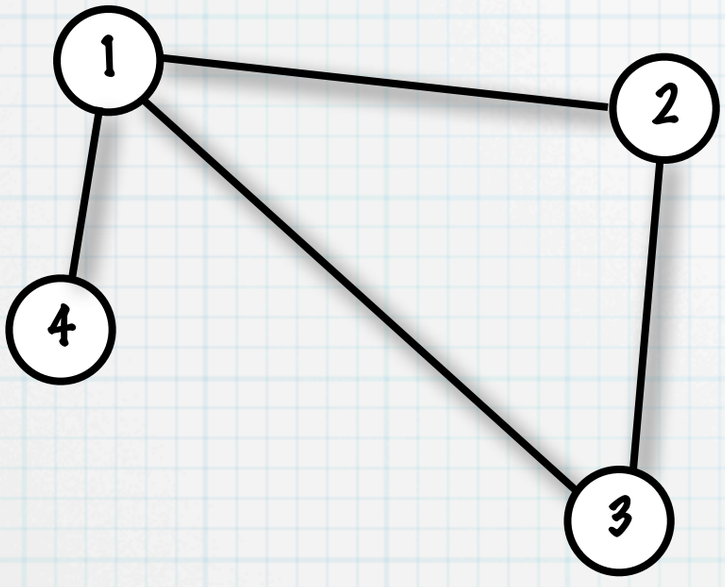
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0

How to represent graphs (1)

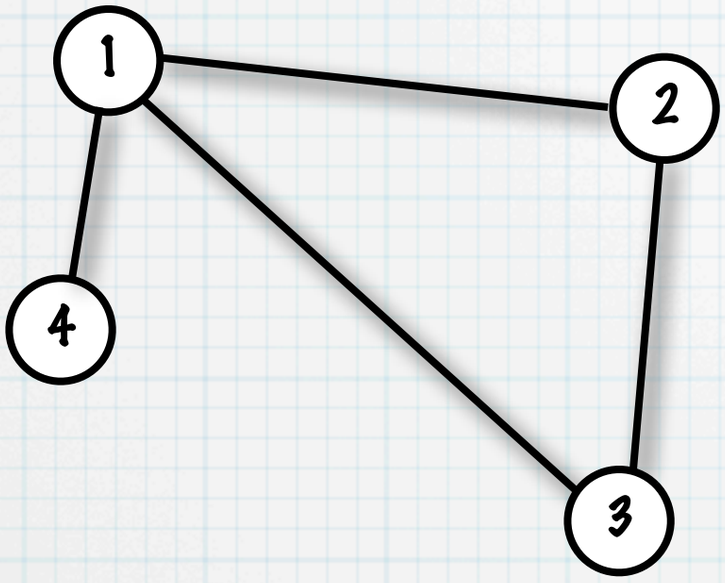
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4				

How to represent graphs (1)

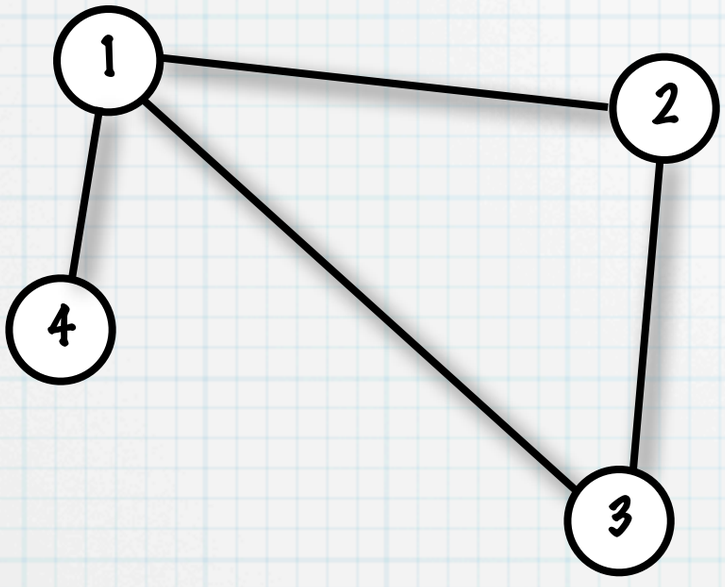
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1			

How to represent graphs (1)

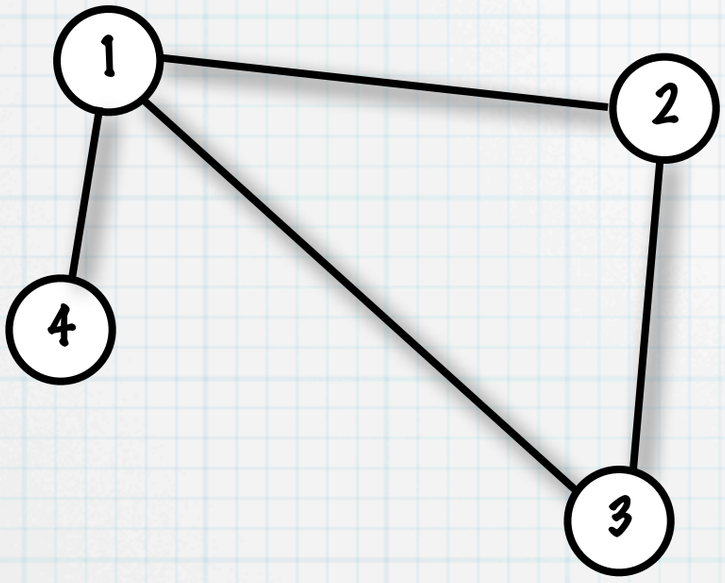
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0		

How to represent graphs (1)

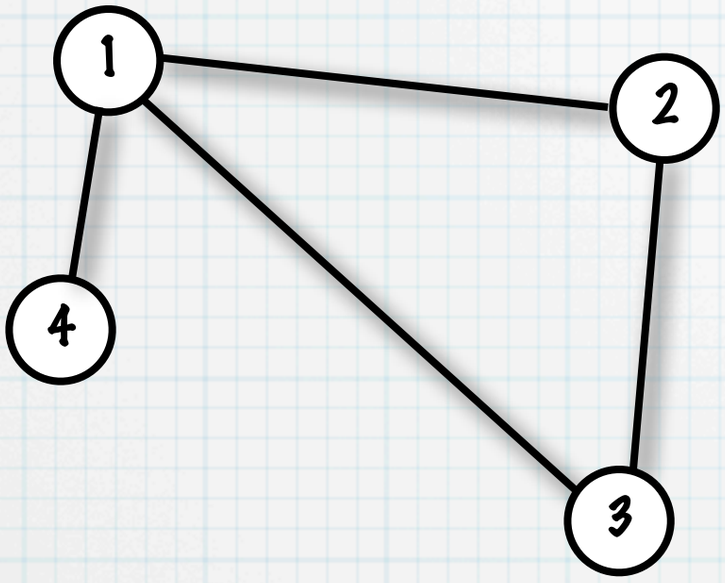
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

How to represent graphs (1)

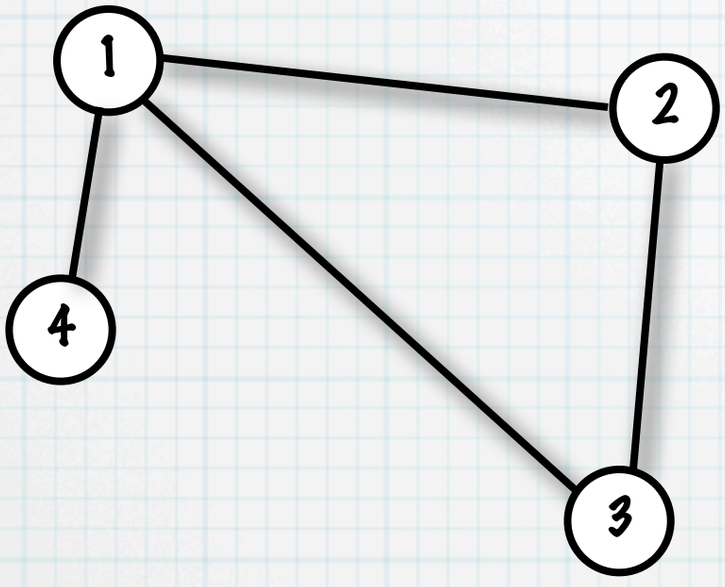
Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

How to represent graphs (1)

Adjacency matrix A

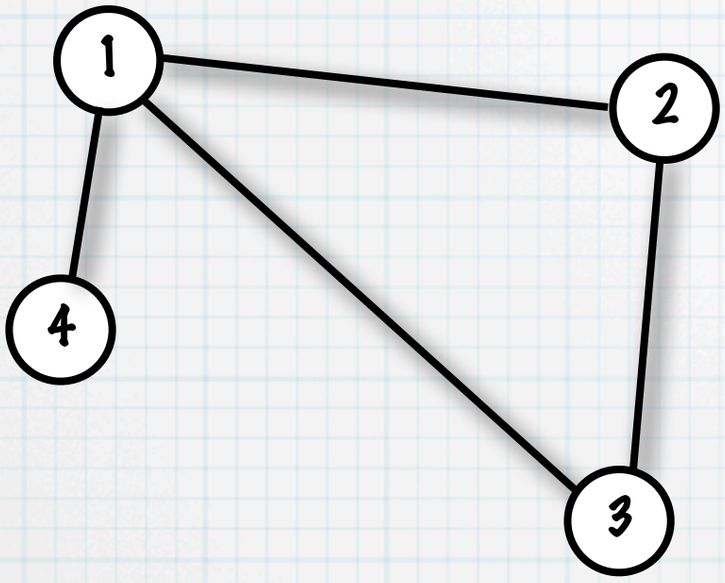


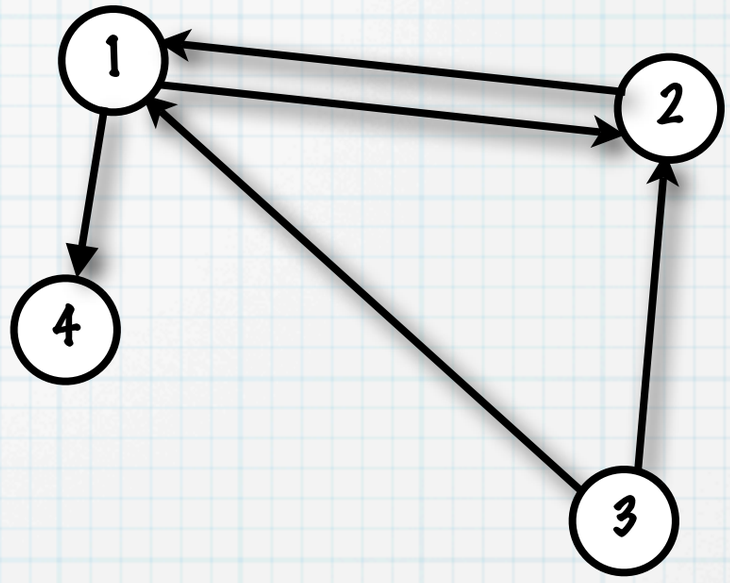
	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored

How to represent graphs (1)

Adjacency matrix A



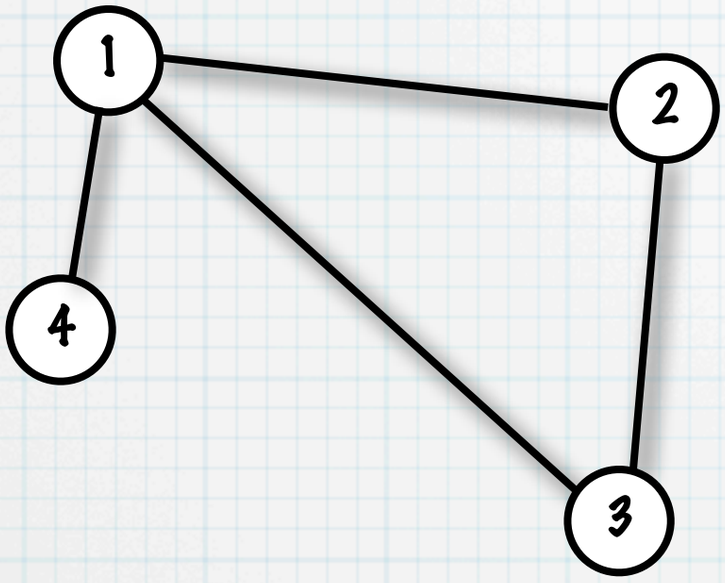


	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper)
triangle can be stored

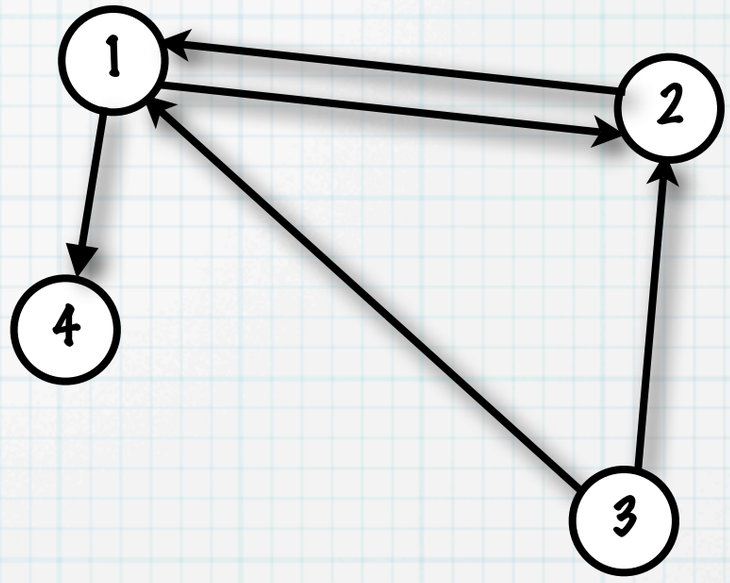
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

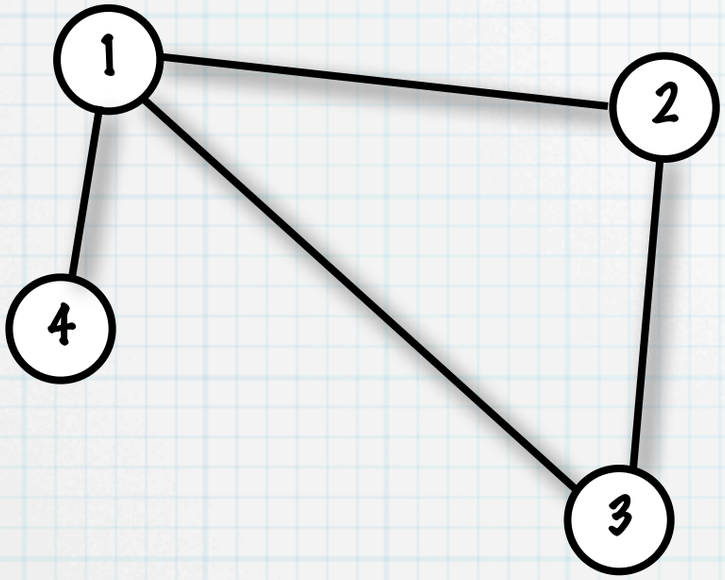
only the lower (or upper) triangle can be stored



tails

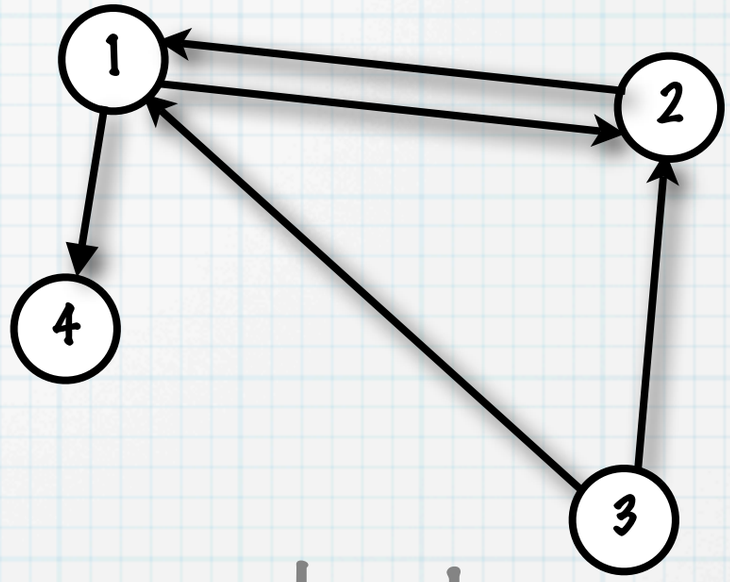
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored

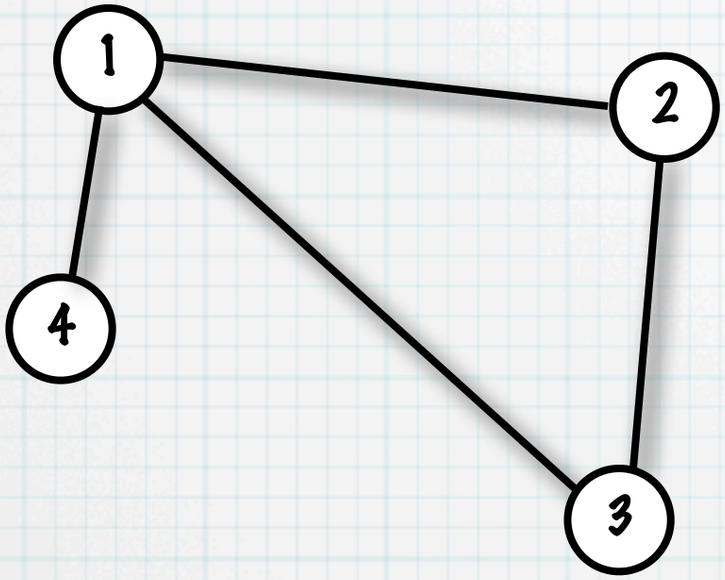


heads

tails

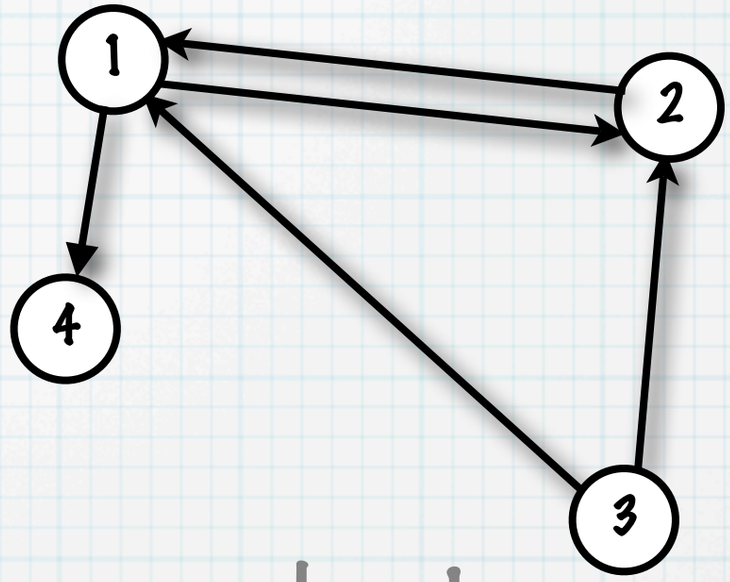
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored

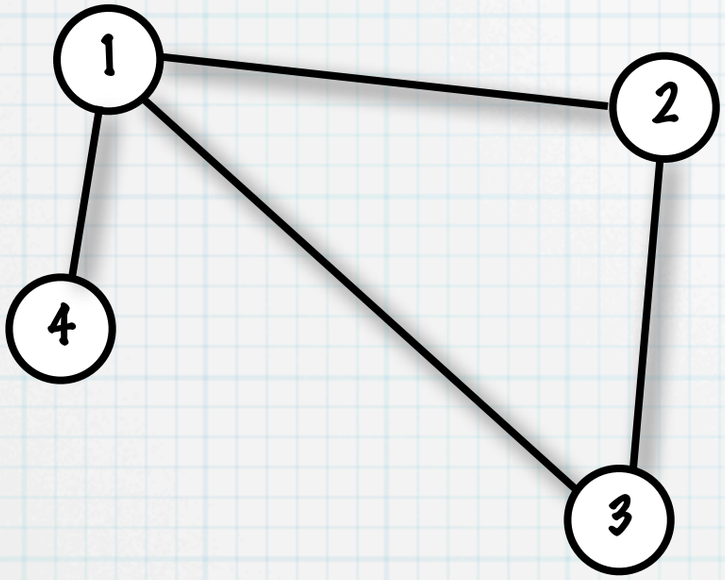


heads

tails

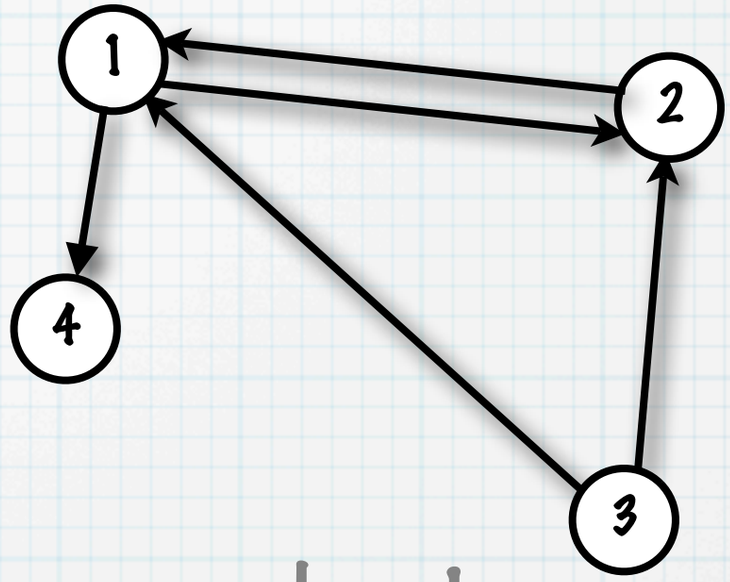
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored

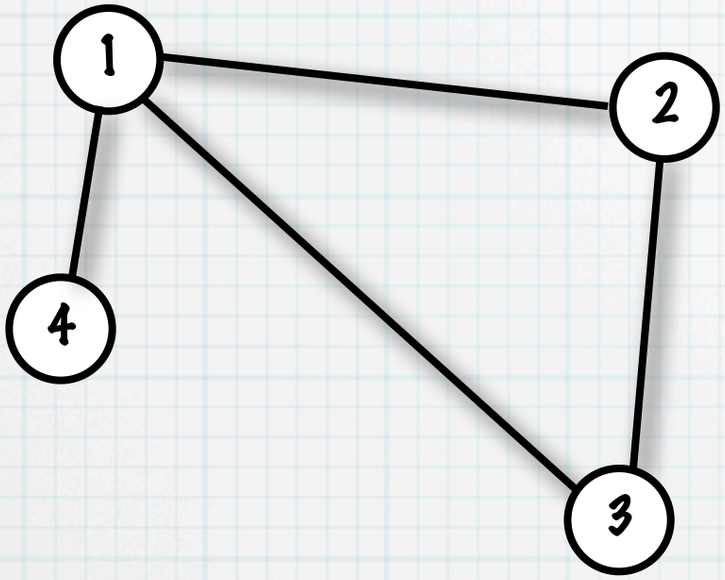


heads

tails

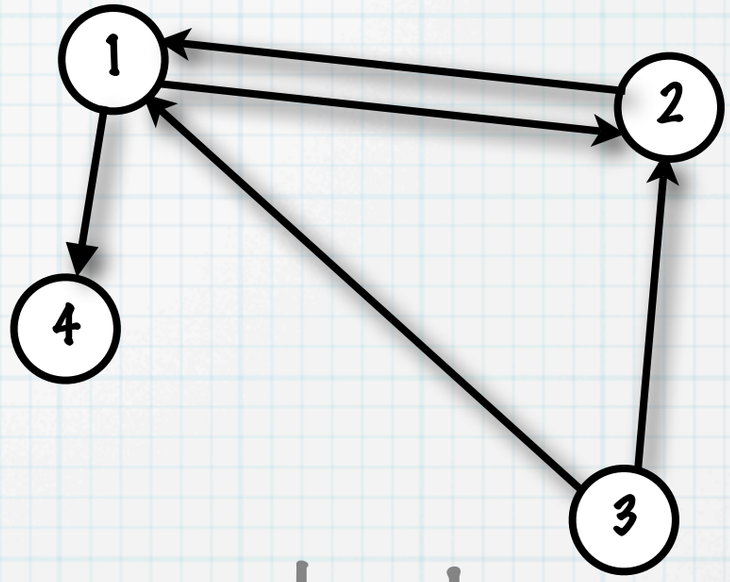
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



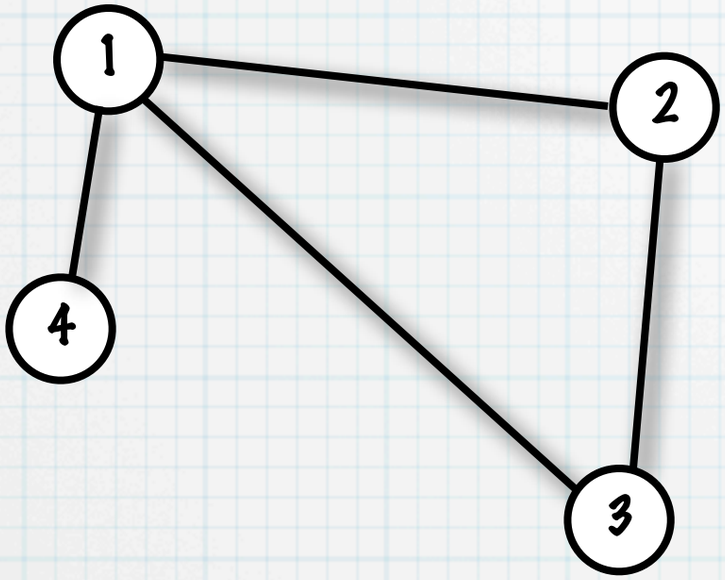
heads

tails

	1			

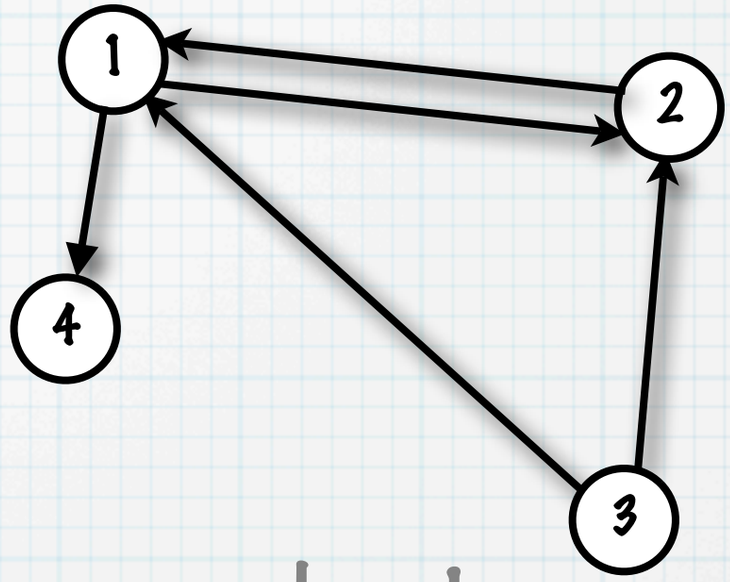
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



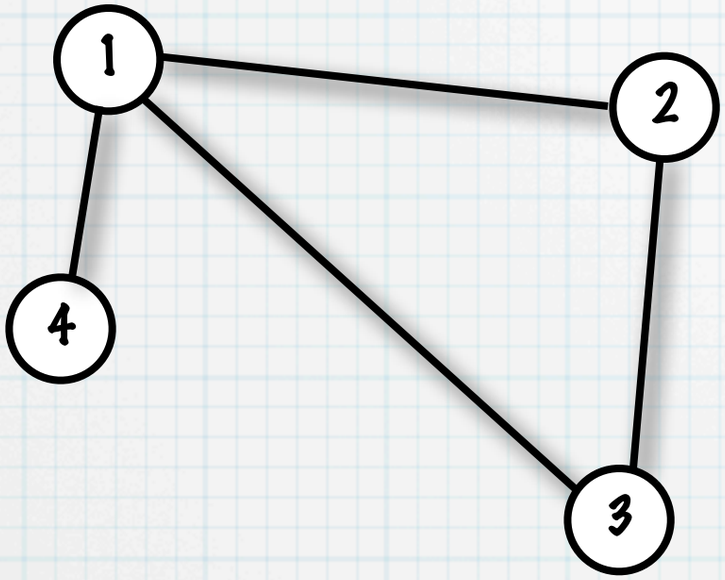
heads

tails

	1	2		

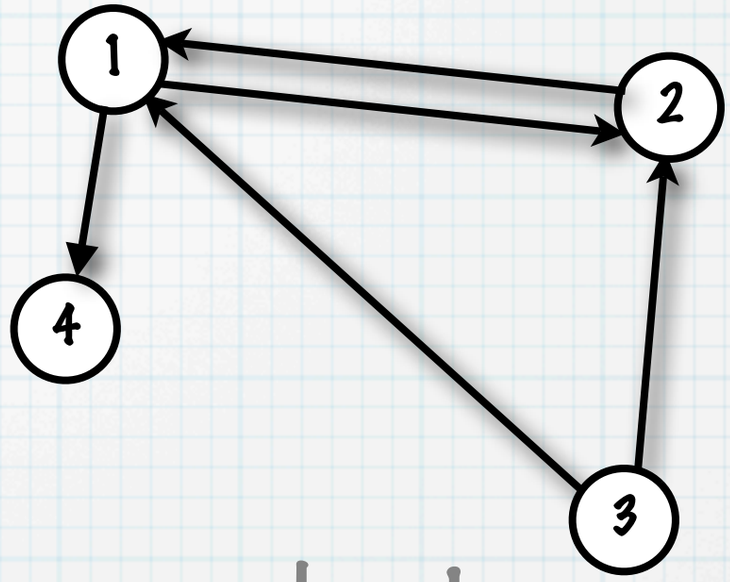
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



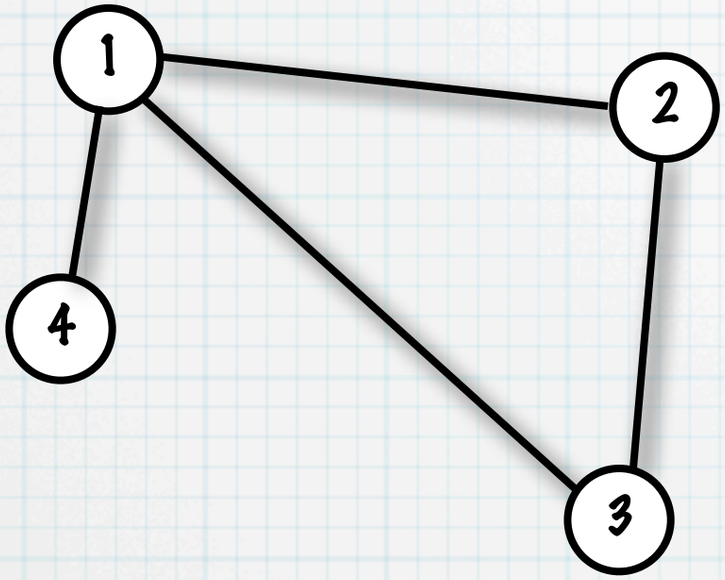
heads

tails

	1	2	3	

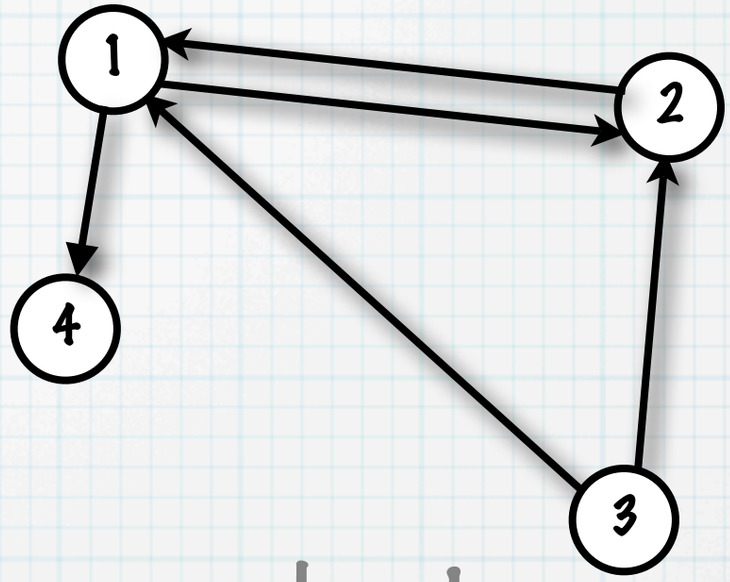
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



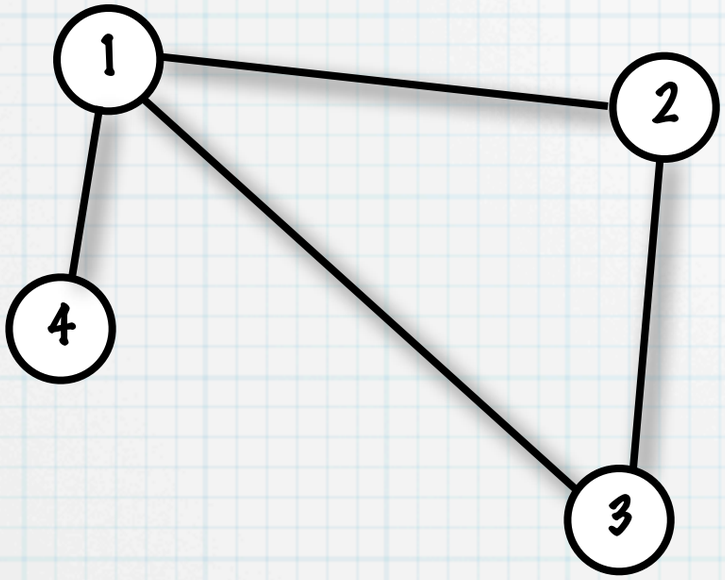
heads

tails

	1	2	3	4

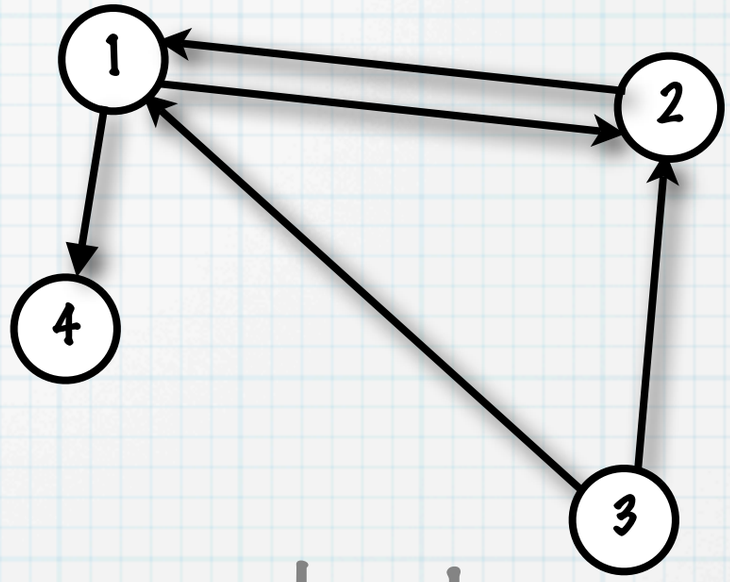
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



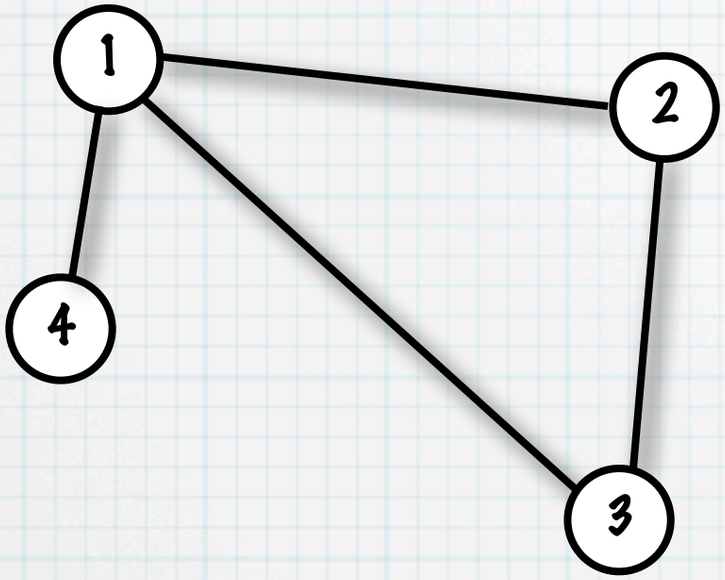
heads

tails

	1	2	3	4
1				

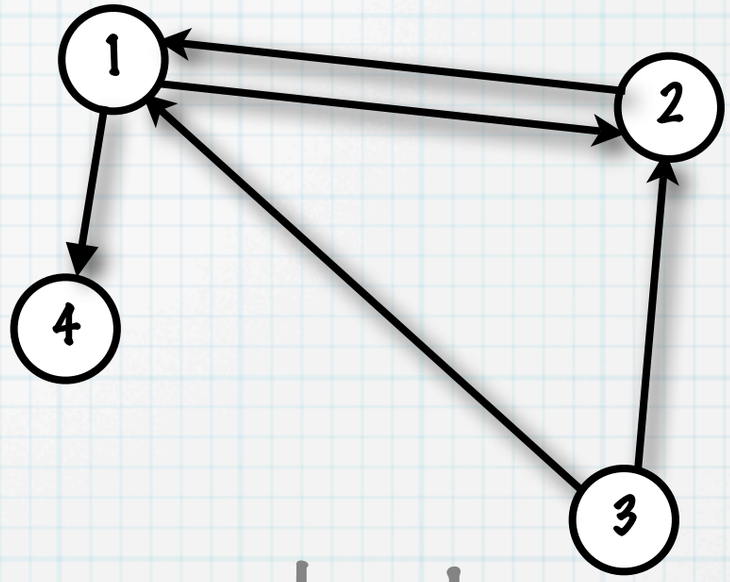
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



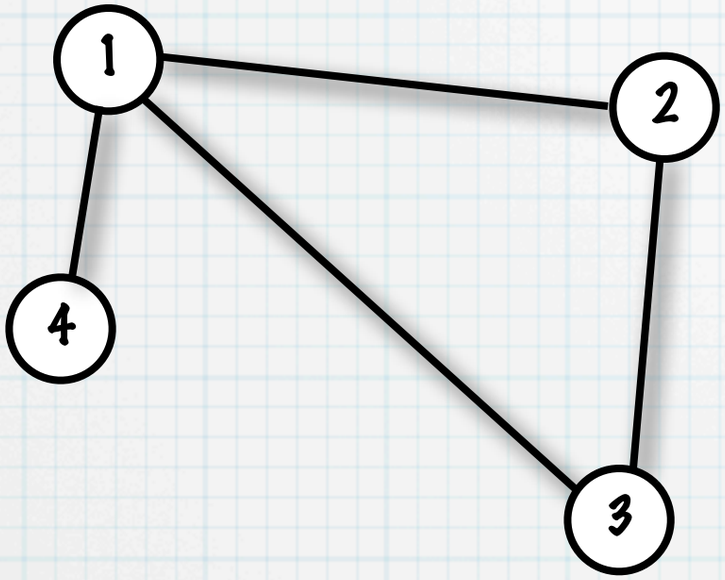
heads

tails

	1	2	3	4
1				

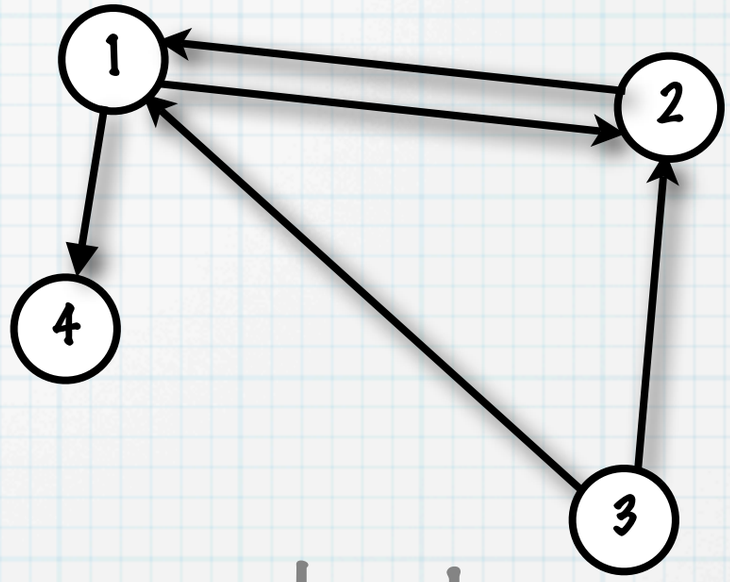
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



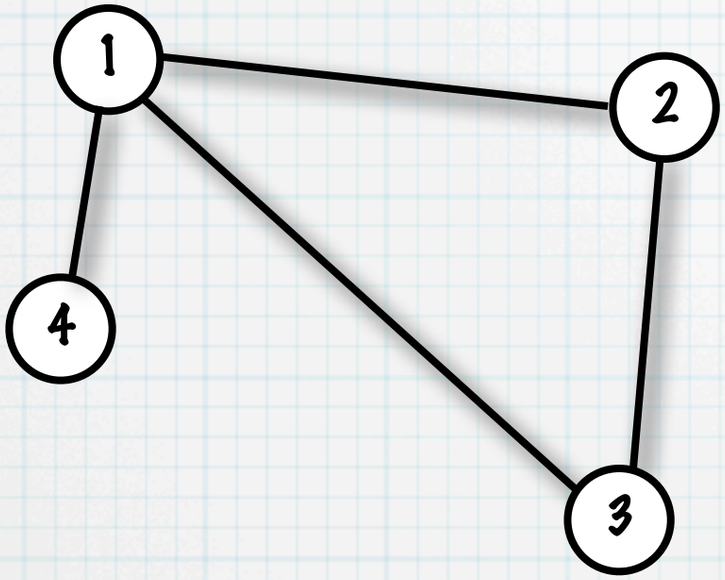
heads

tails

	1	2	3	4
1		1		

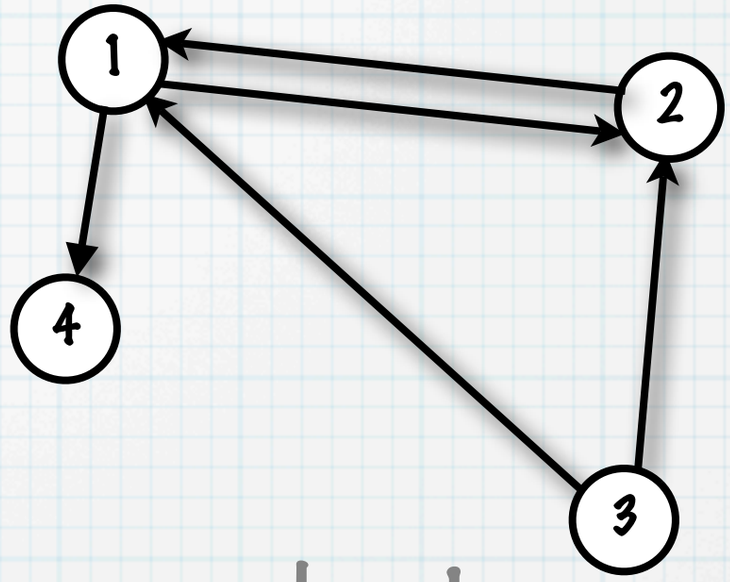
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



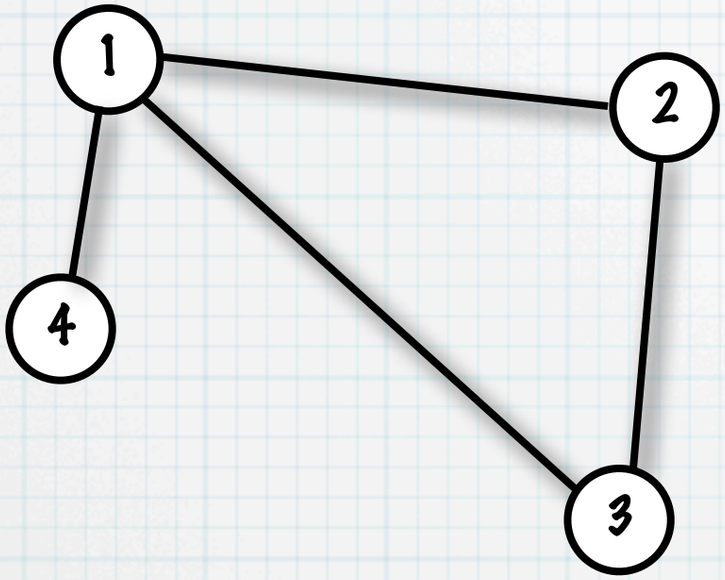
heads

tails

	1	2	3	4
1		1	0	

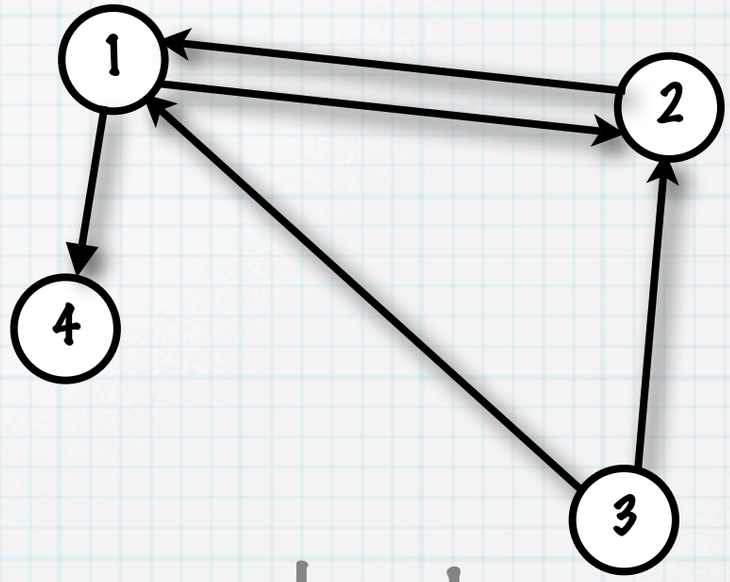
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



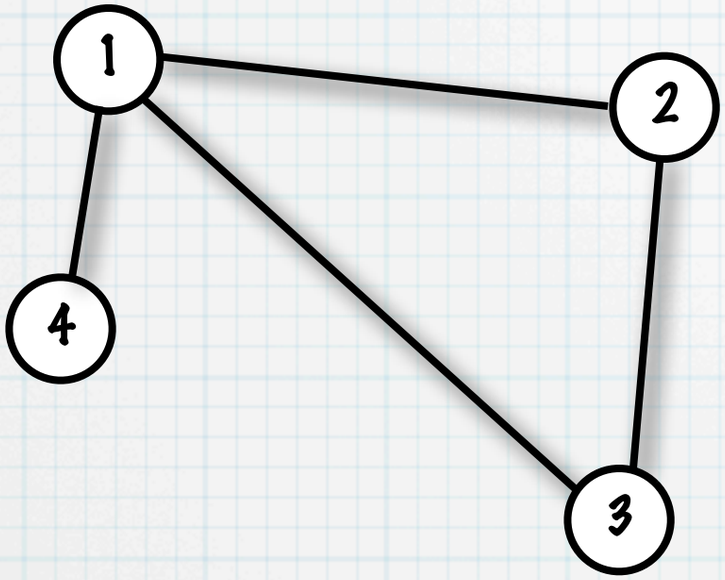
heads

tails

	1	2	3	4
1		1	0	1

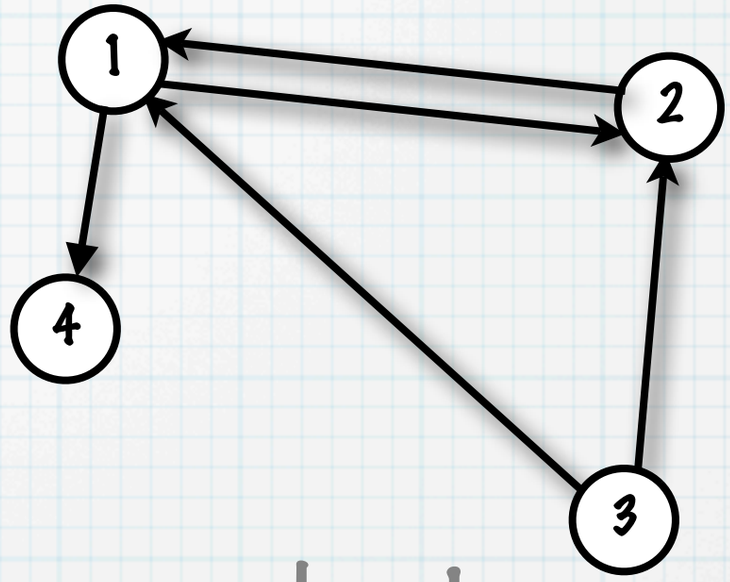
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



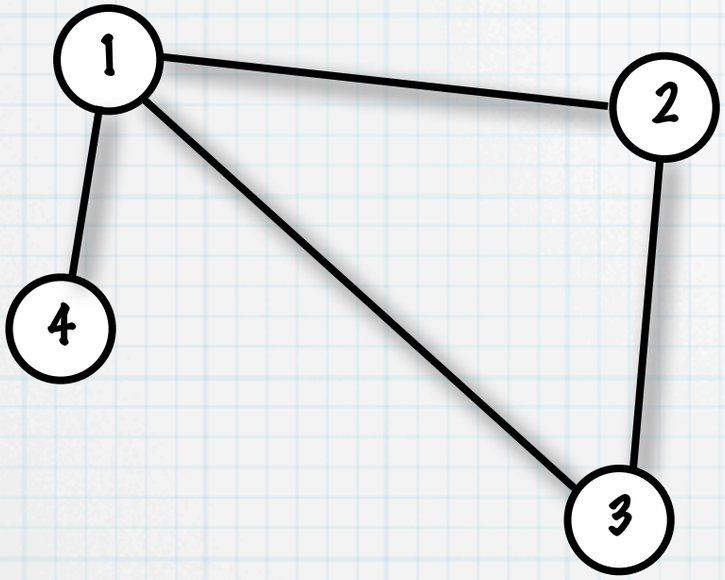
heads

tails

	1	2	3	4
1		1	0	1
2				

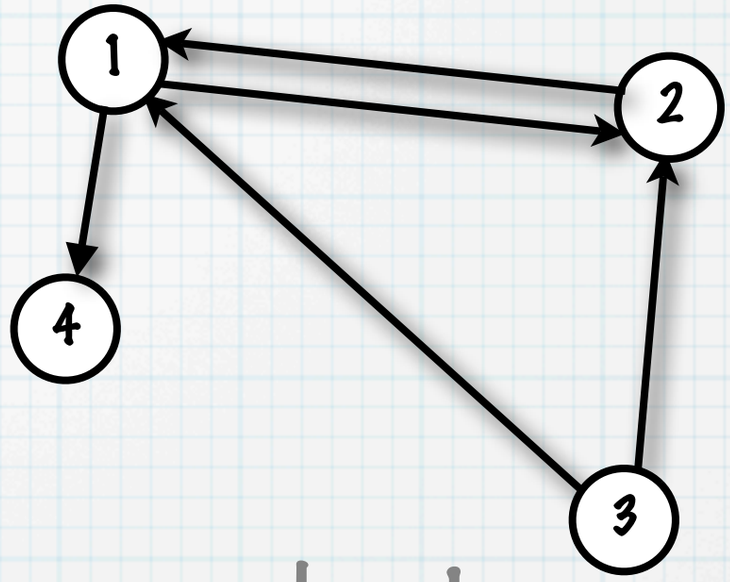
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



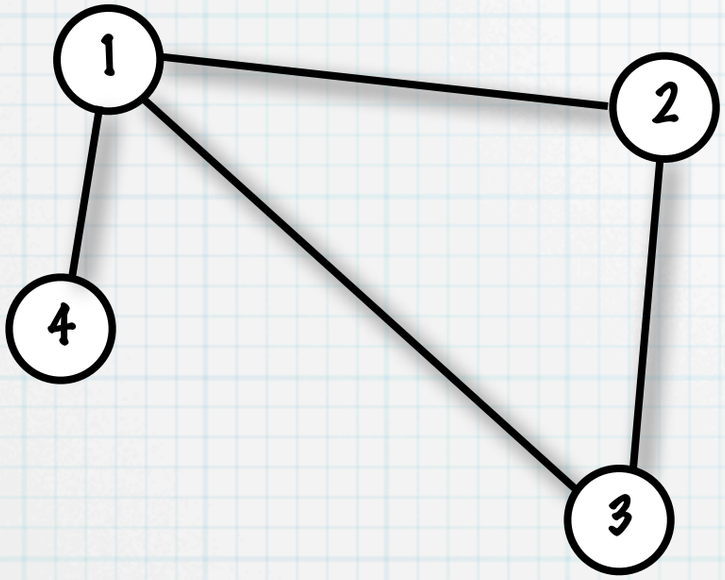
heads

tails

	1	2	3	4
1		1	0	1
2	1			

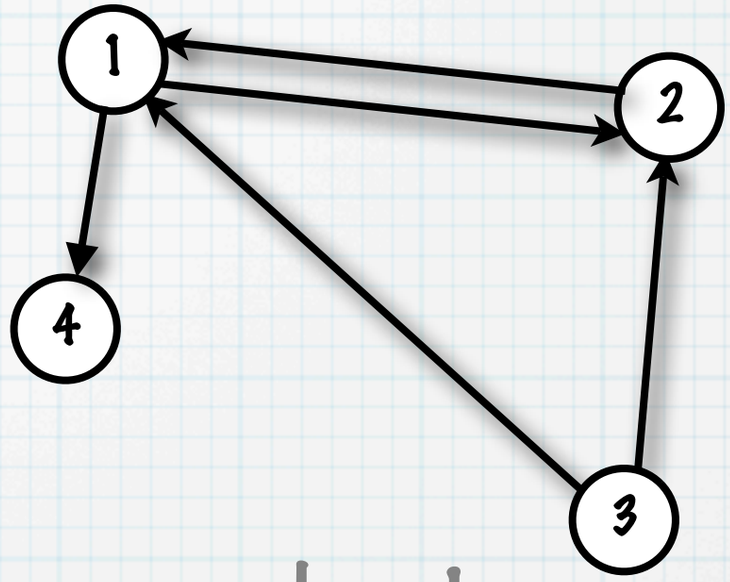
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



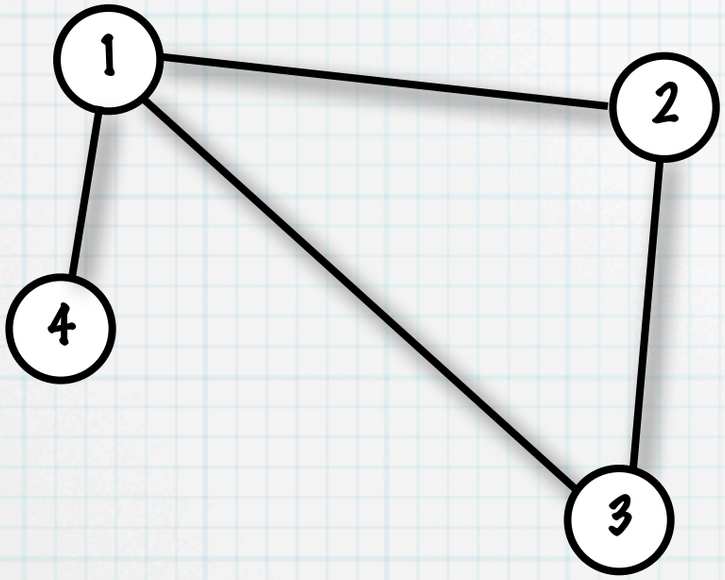
heads

tails

	1	2	3	4
1		1	0	1
2	1			

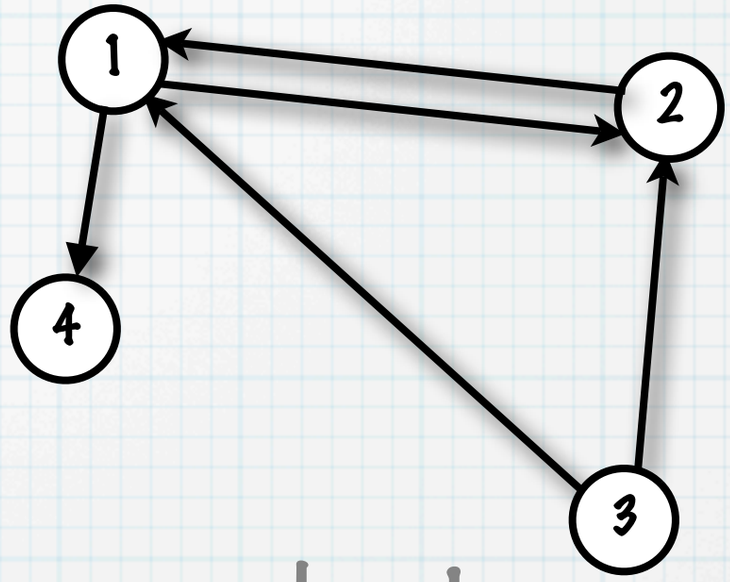
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



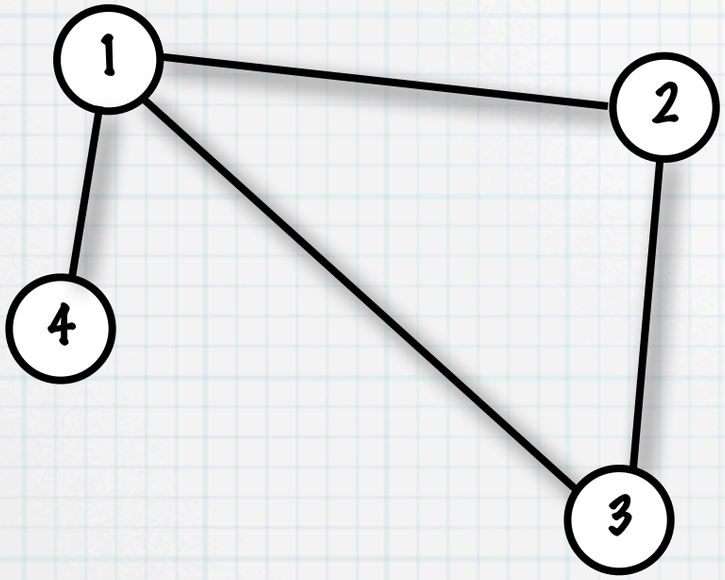
heads

tails

	1	2	3	4
1		1	0	1
2	1		0	

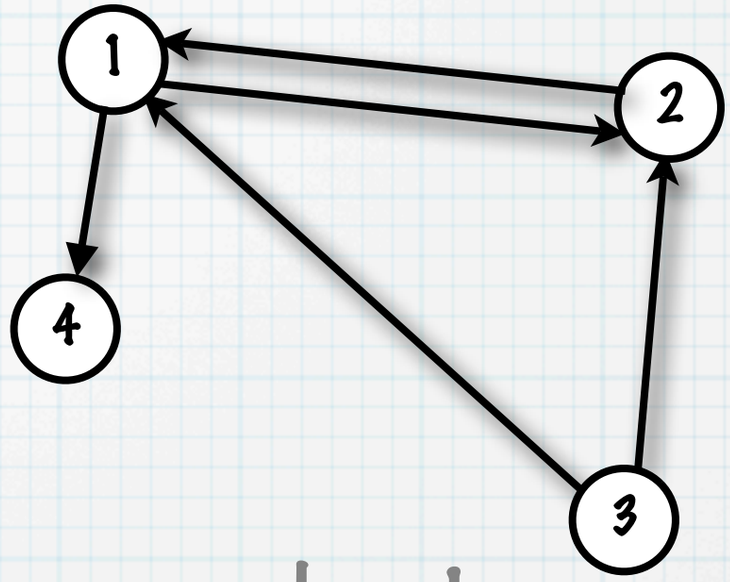
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



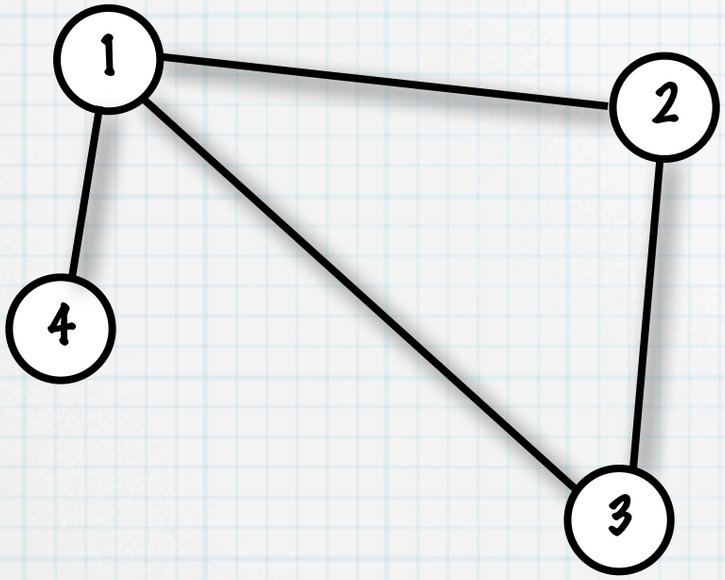
heads

tails

	1	2	3	4
1		1	0	1
2	1		0	0

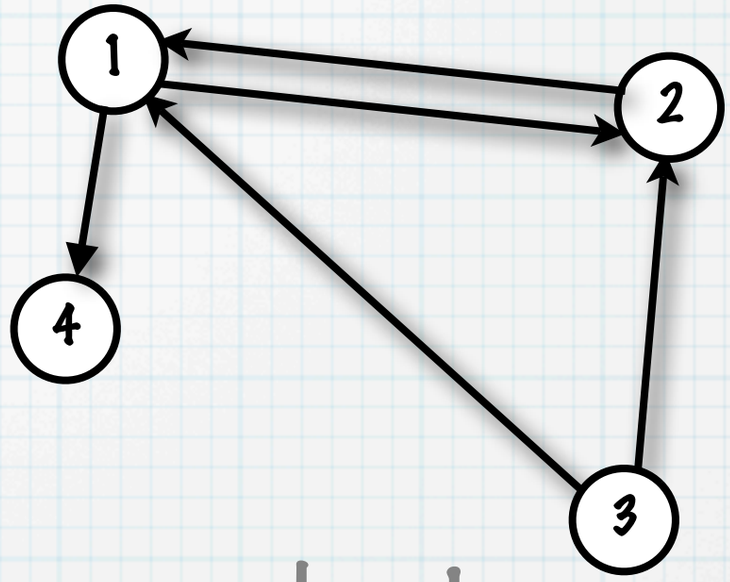
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



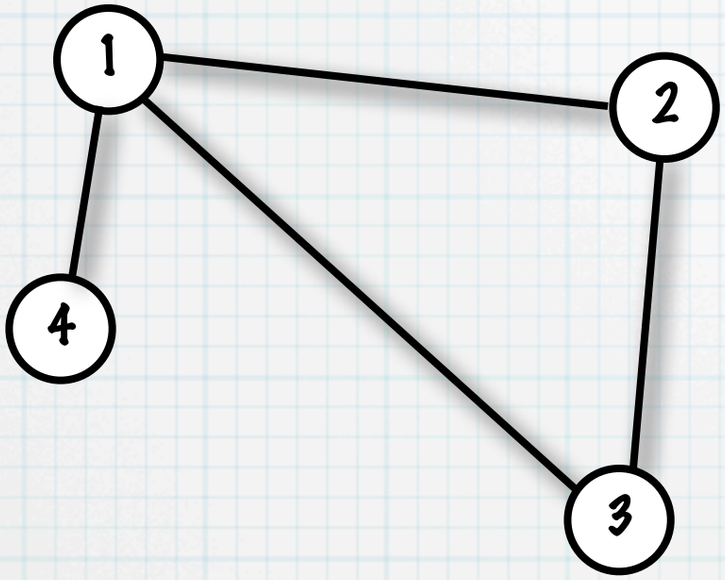
heads

tails

	1	2	3	4
1		1	0	1
2	1		0	0
3				

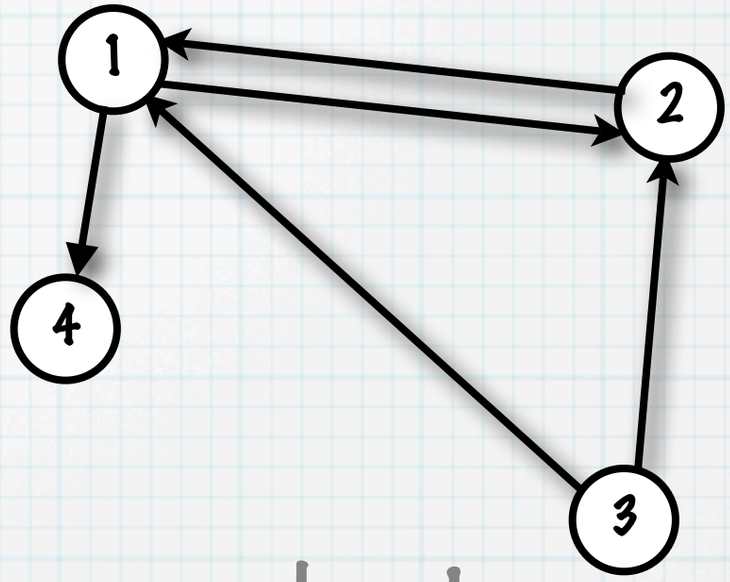
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



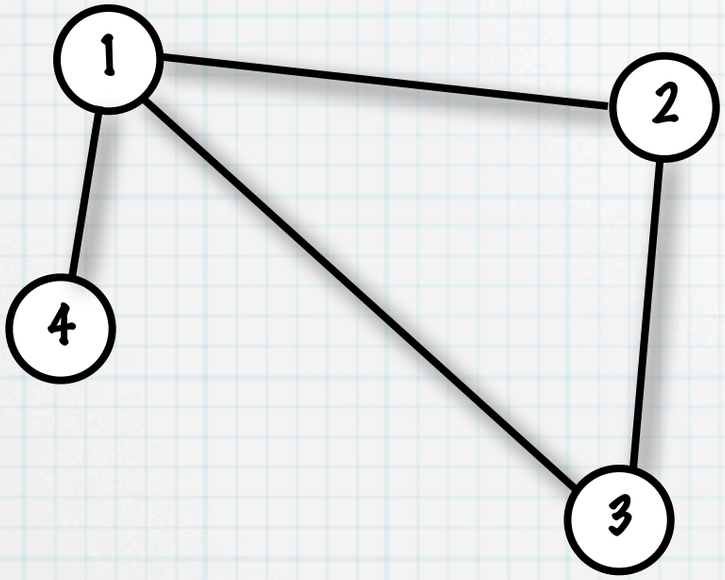
heads

tails

	1	2	3	4
1		1	0	1
2	1		0	0
3	1			

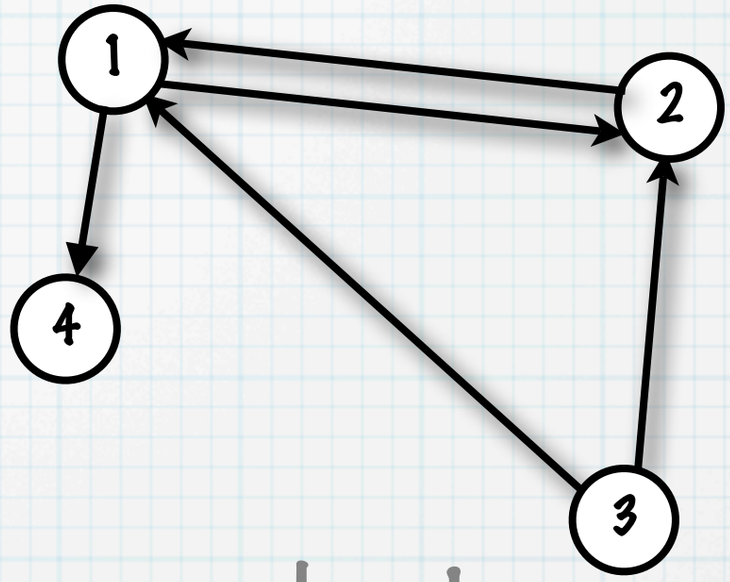
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



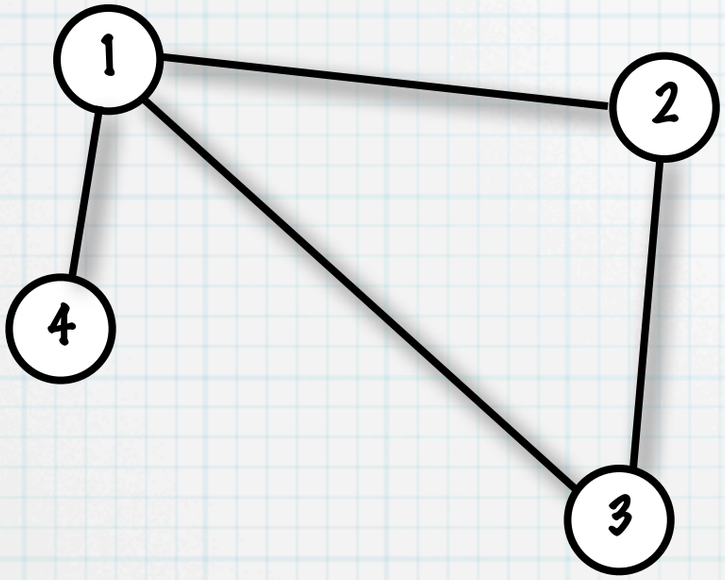
heads

tails

	1	2	3	4
1		1	0	1
2	1		0	0
3	1	1		

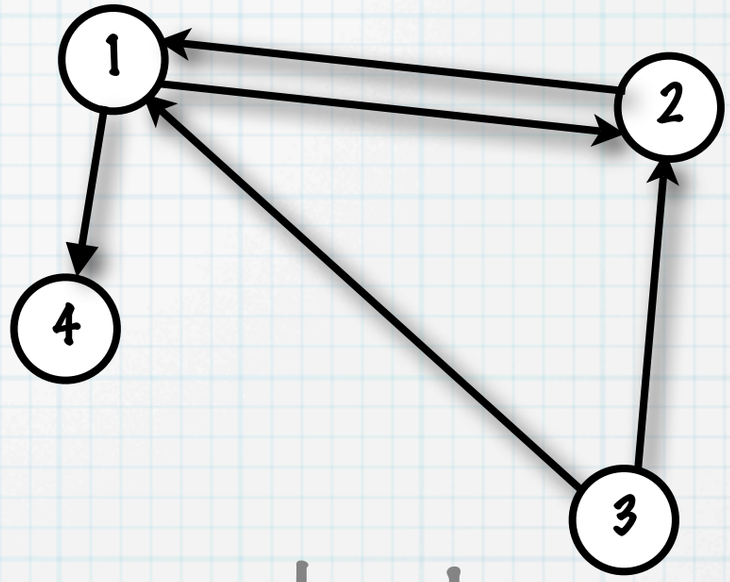
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



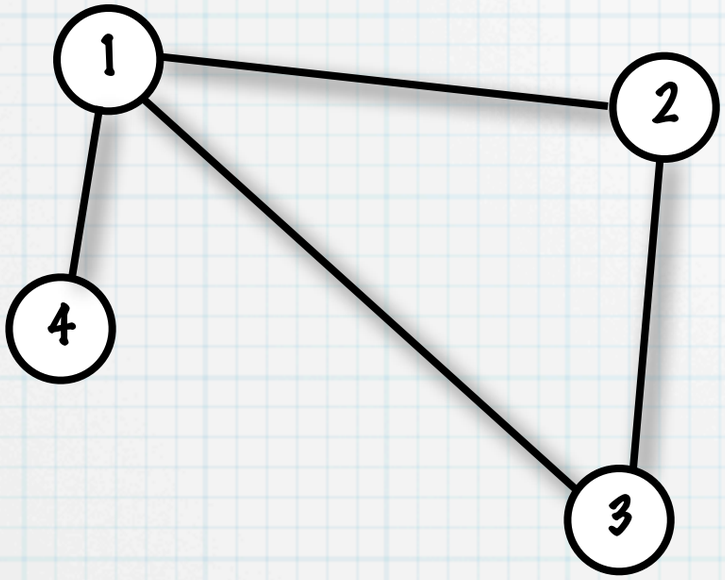
heads

tails

	1	2	3	4
1		1	0	1
2	1		0	0
3	1	1		

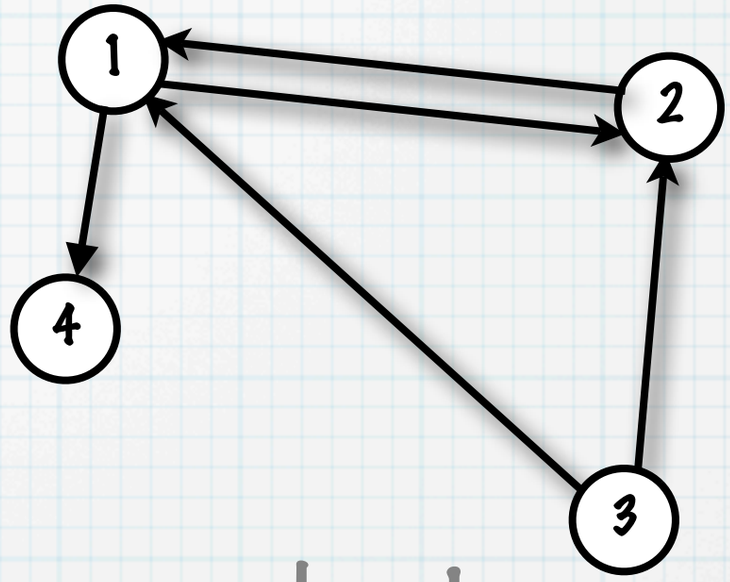
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



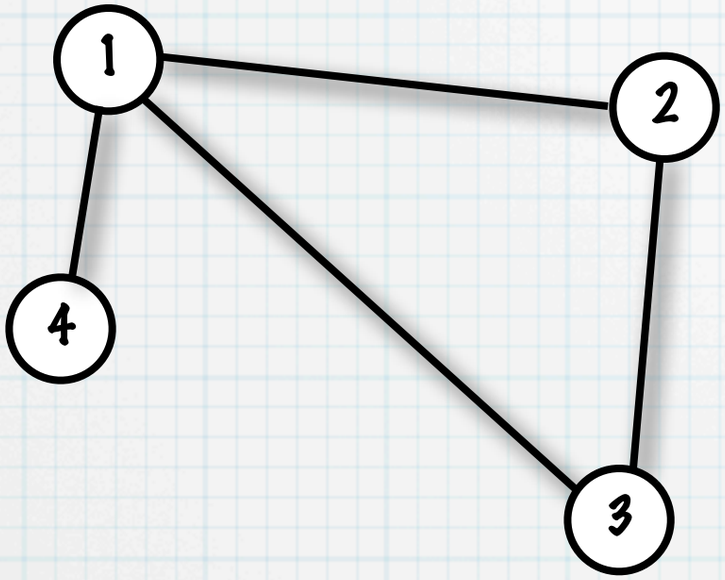
heads

tails

	1	2	3	4
1		1	0	1
2	1		0	0
3	1	1		0

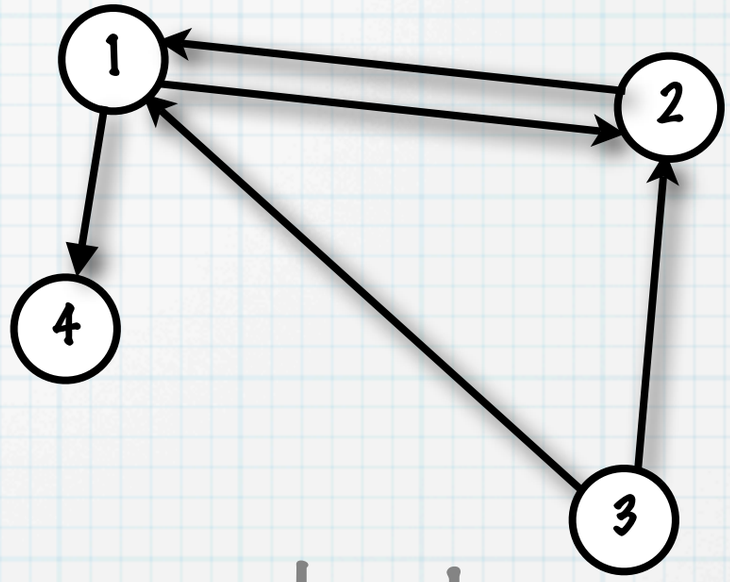
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



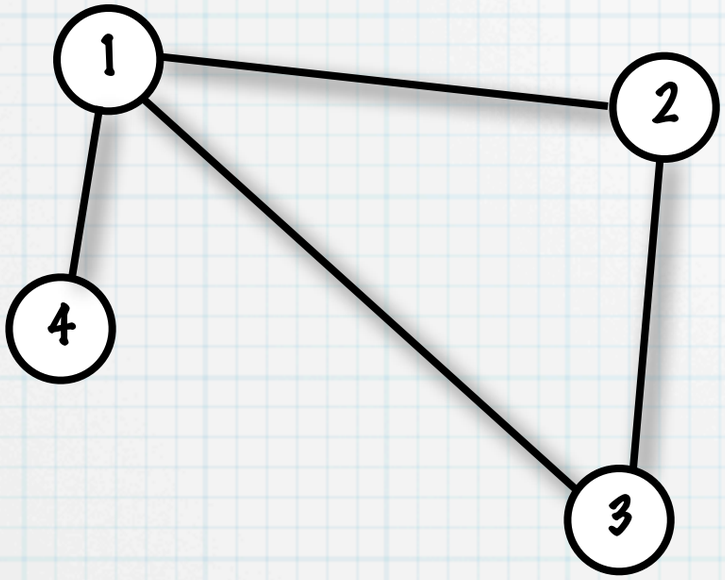
heads

tails

	1	2	3	4
1		1	0	1
2	1		0	0
3	1	1		0
4				

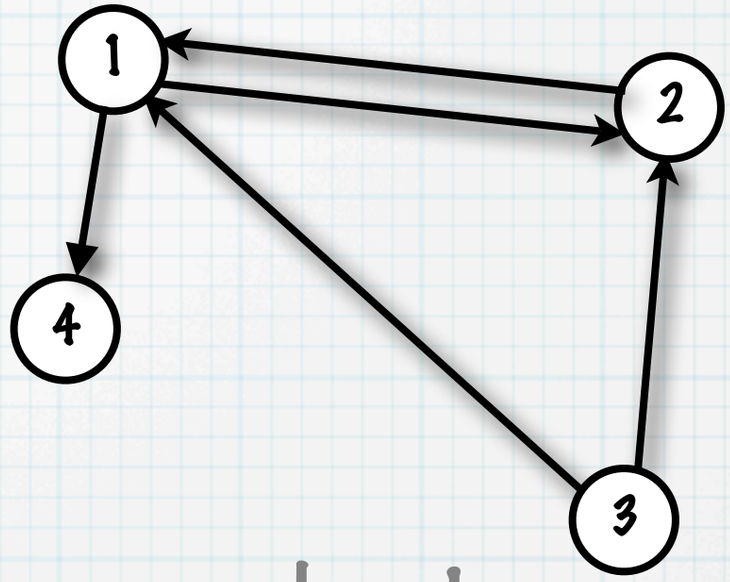
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



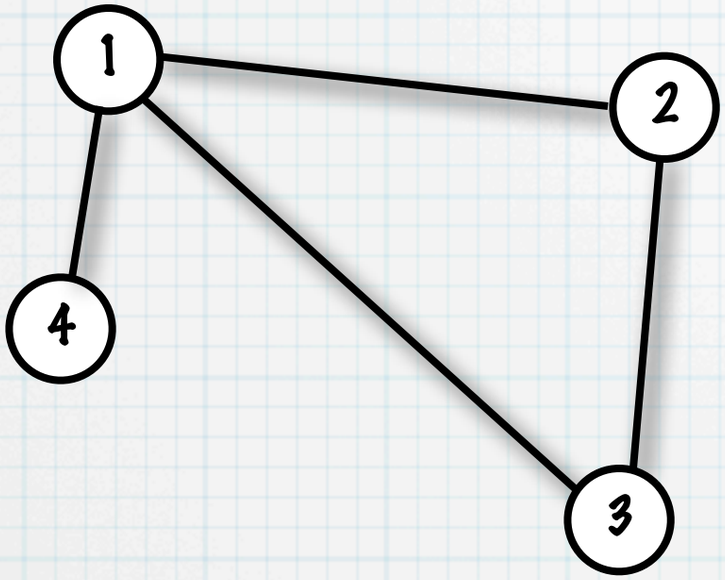
heads

tails

	1	2	3	4
1		1	0	1
2	1		0	0
3	1	1		0
4	0			

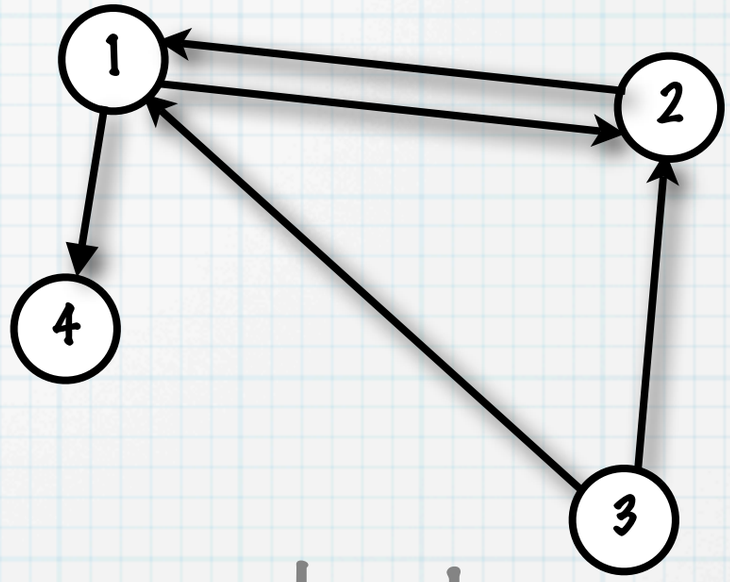
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



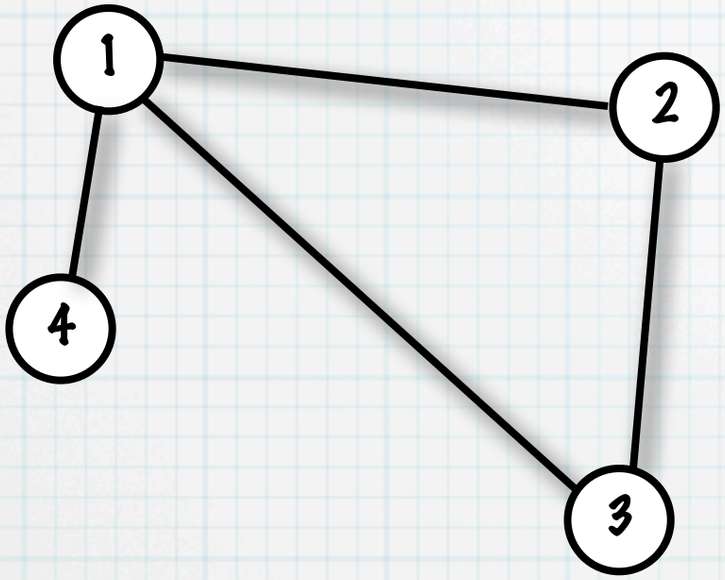
heads

tails

	1	2	3	4
1		1	0	1
2	1		0	0
3	1	1		0
4	0	0		

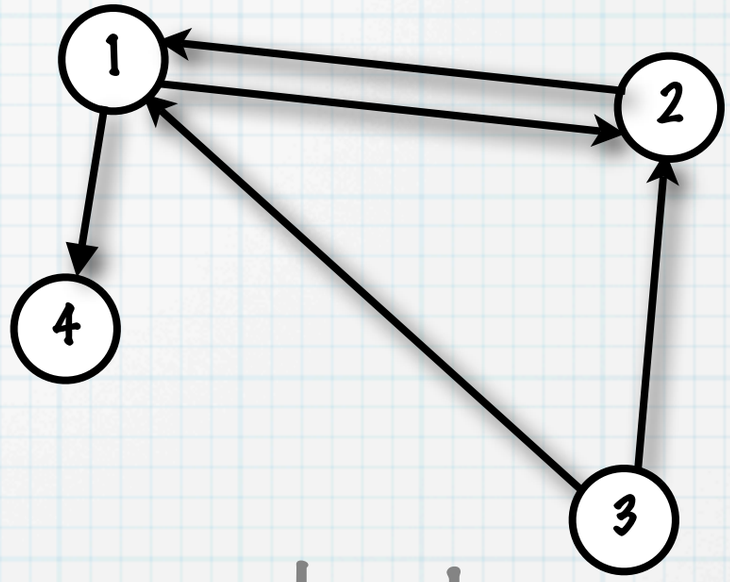
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



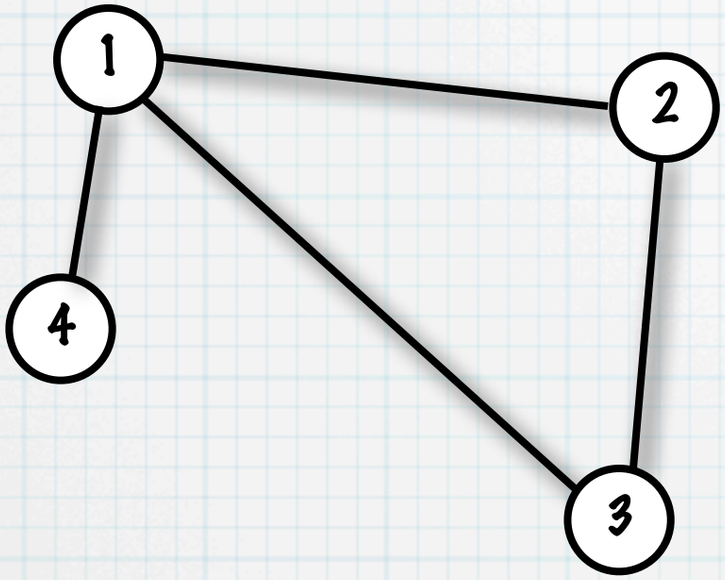
heads

tails

	1	2	3	4
1		1	0	1
2	1		0	0
3	1	1		0
4	0	0	0	

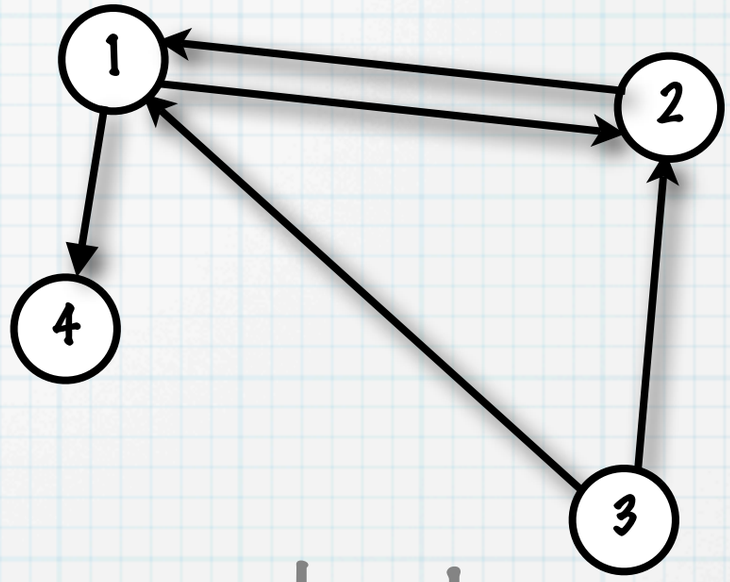
How to represent graphs (1)

Adjacency matrix A



	1	2	3	4
1		1	1	1
2	1		1	0
3	1	1		0
4	1	0	0	

only the lower (or upper) triangle can be stored



heads

tails

	1	2	3	4
1		1	0	1
2	1		0	0
3	1	1		0
4	0	0	0	

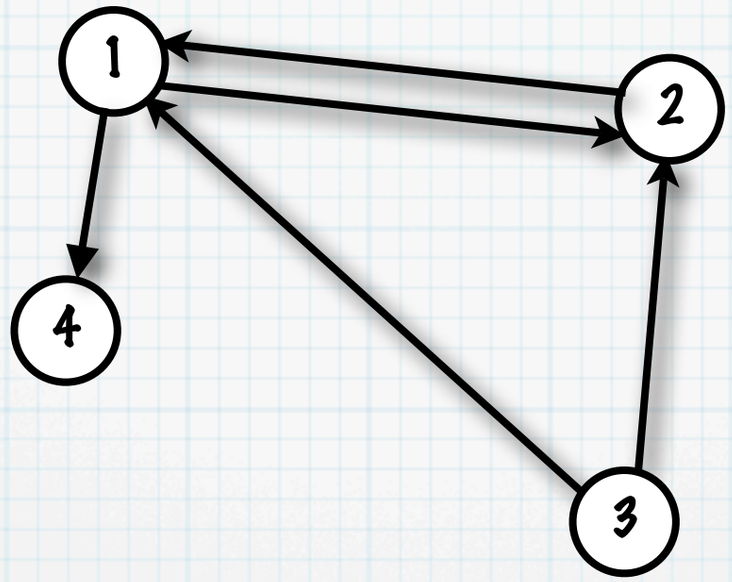
How to represent graphs (2)

How to represent graphs (2)

Adjacency lists

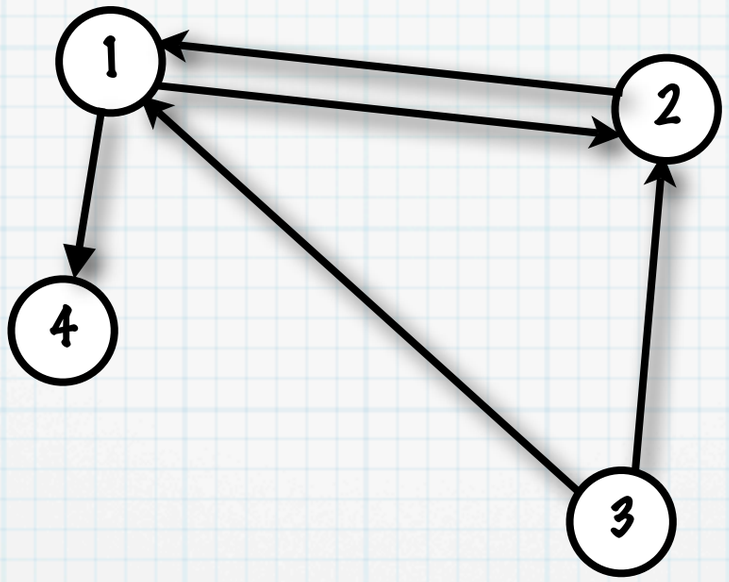
How to represent graphs (2)

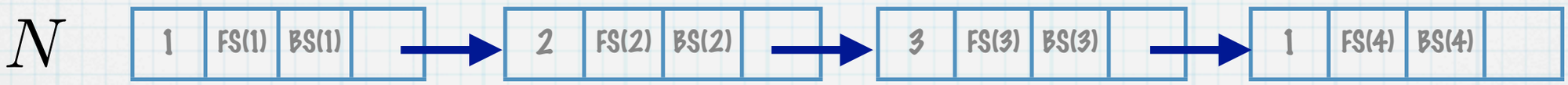
Adjacency lists



How to represent graphs (2)

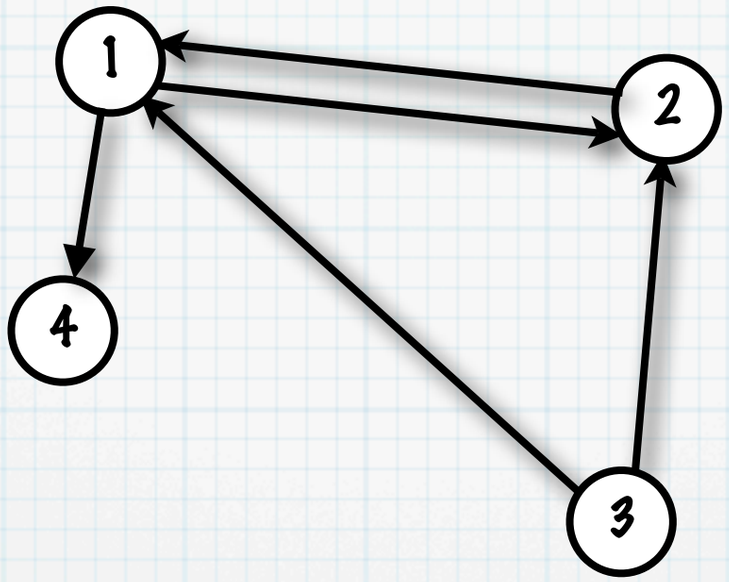
Adjacency lists

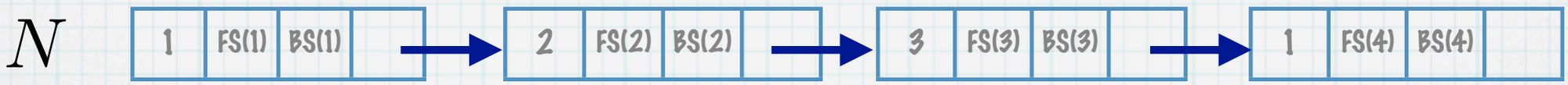




How to represent graphs (2)

Adjacency lists

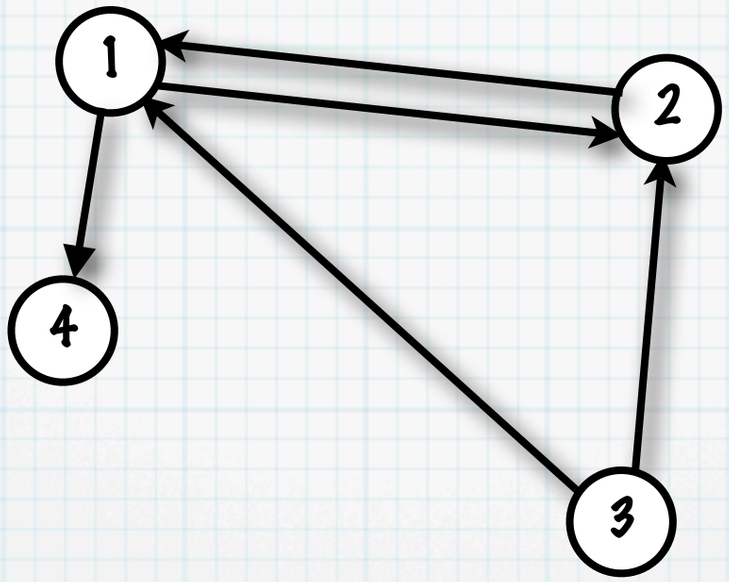


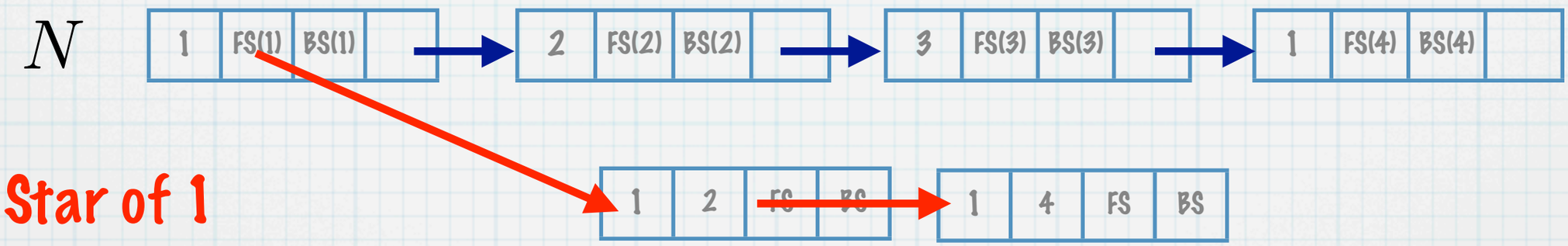


Forward Star of 1

How to represent graphs (2)

Adjacency lists

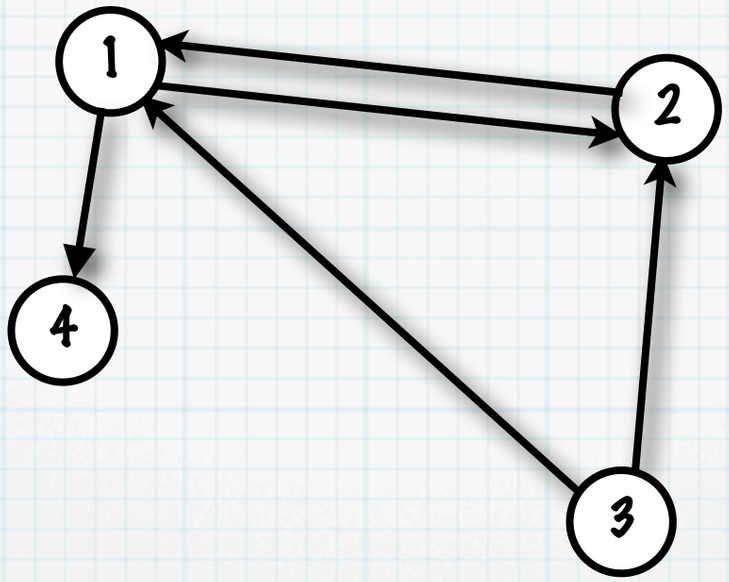


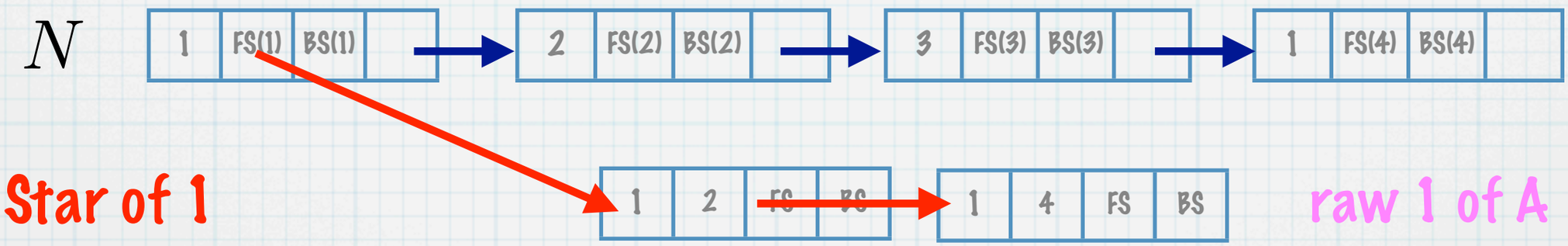


Forward Star of 1

How to represent graphs (2)

Adjacency lists



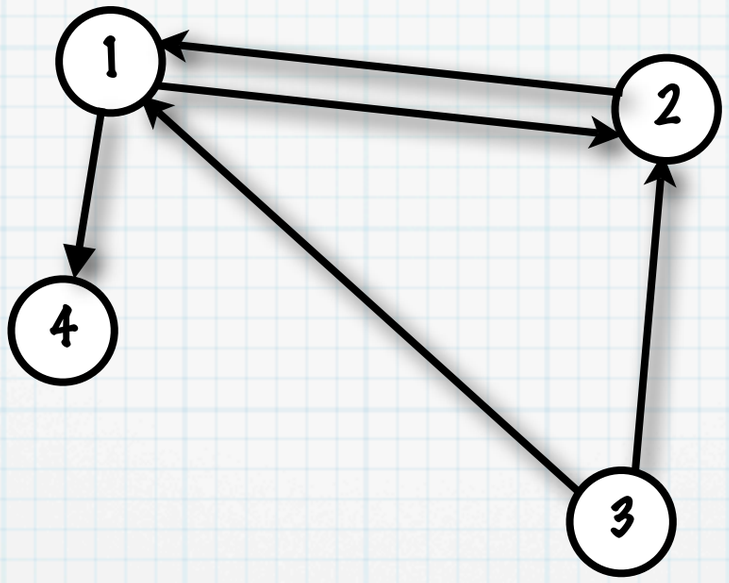


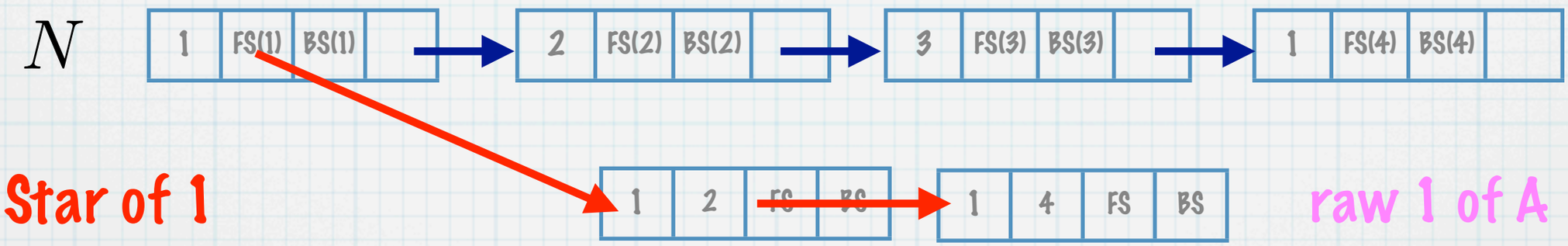
Forward Star of 1

raw 1 of A

How to represent graphs (2)

Adjacency lists





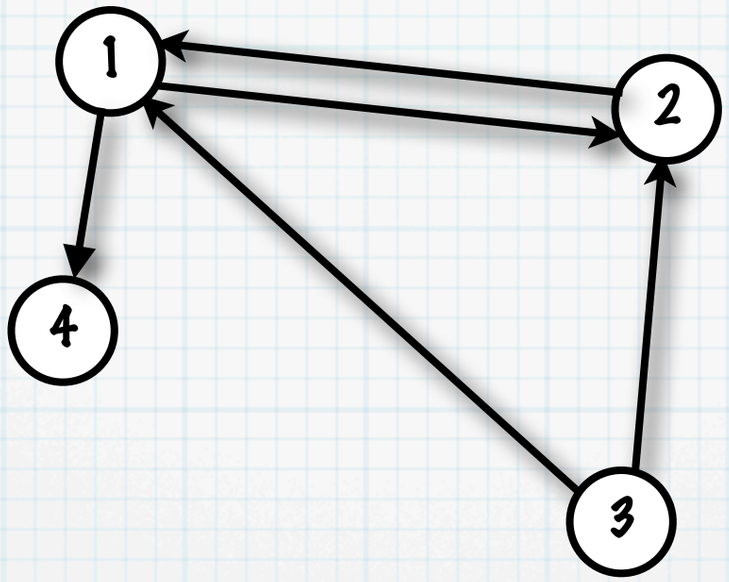
Forward Star of 1

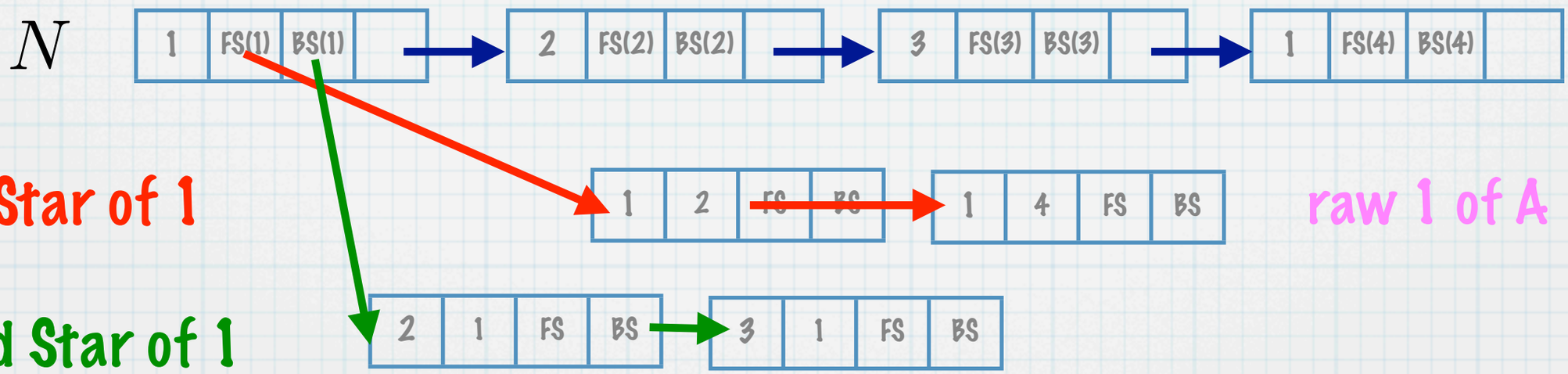
Backward Star of 1

raw 1 of A

How to represent graphs (2)

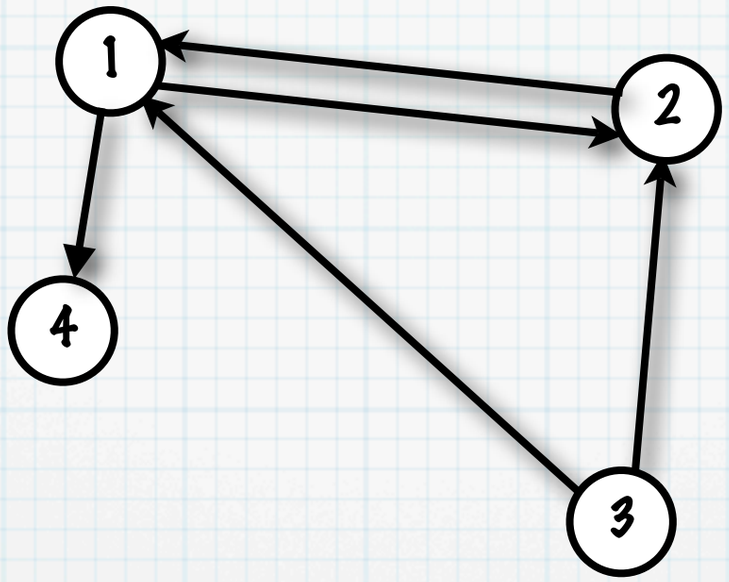
Adjacency lists

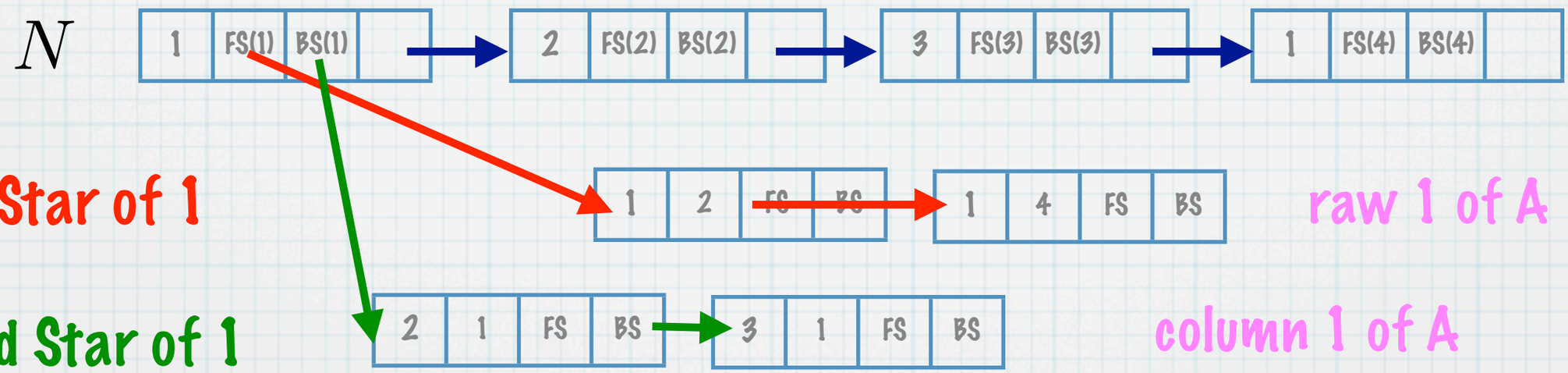




How to represent graphs (2)

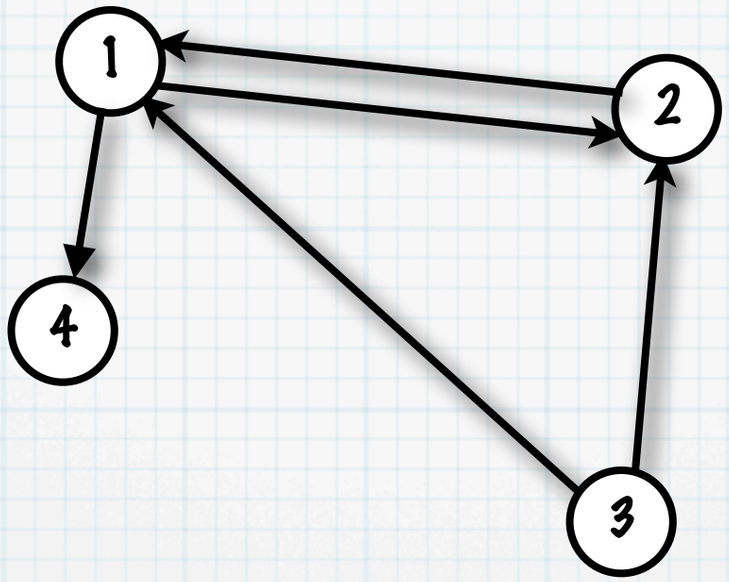
Adjacency lists

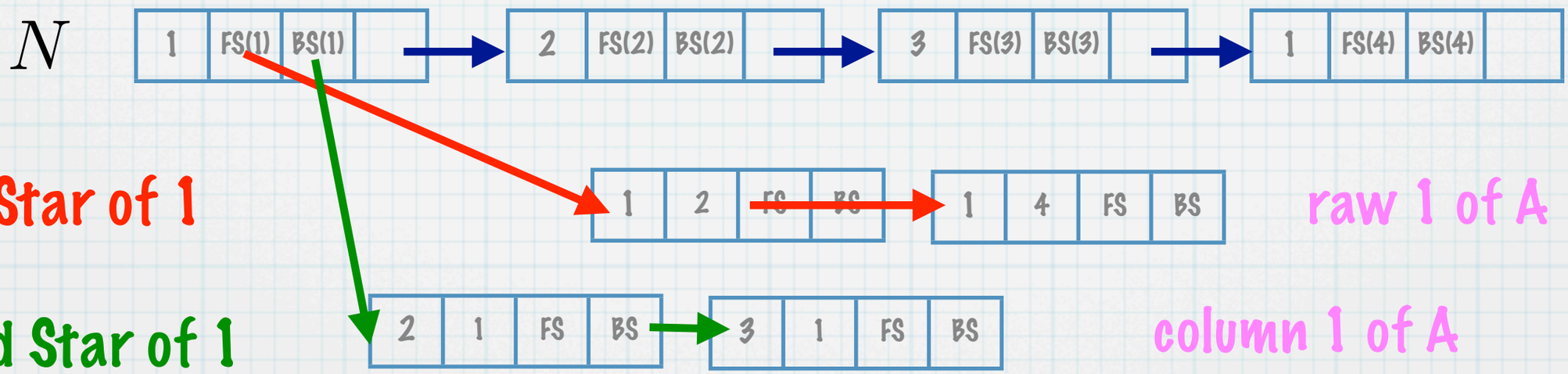




How to represent graphs (2)

Adjacency lists





Forward Star of 1

Backward Star of 1

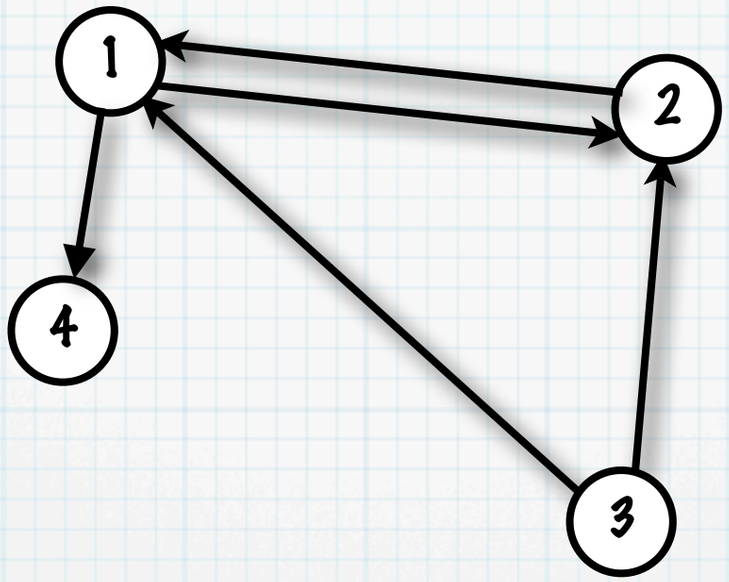
Forward Star of 2

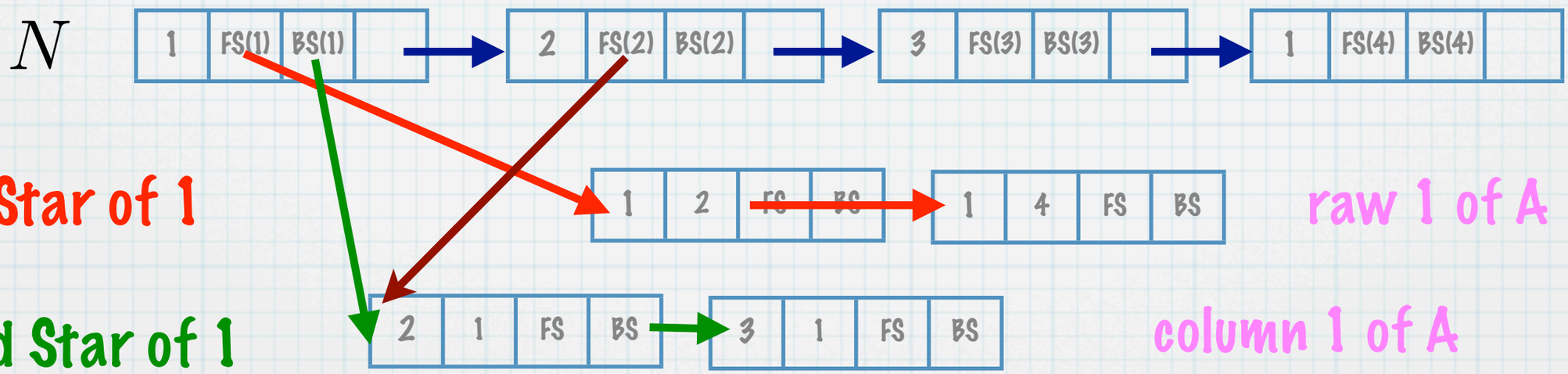
raw 1 of A

column 1 of A

How to represent graphs (2)

Adjacency lists





Forward Star of 1

Backward Star of 1

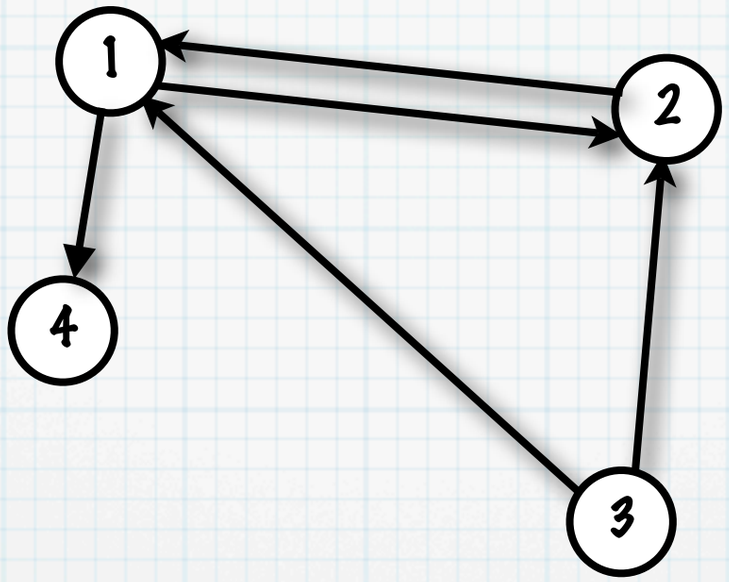
Forward Star of 2

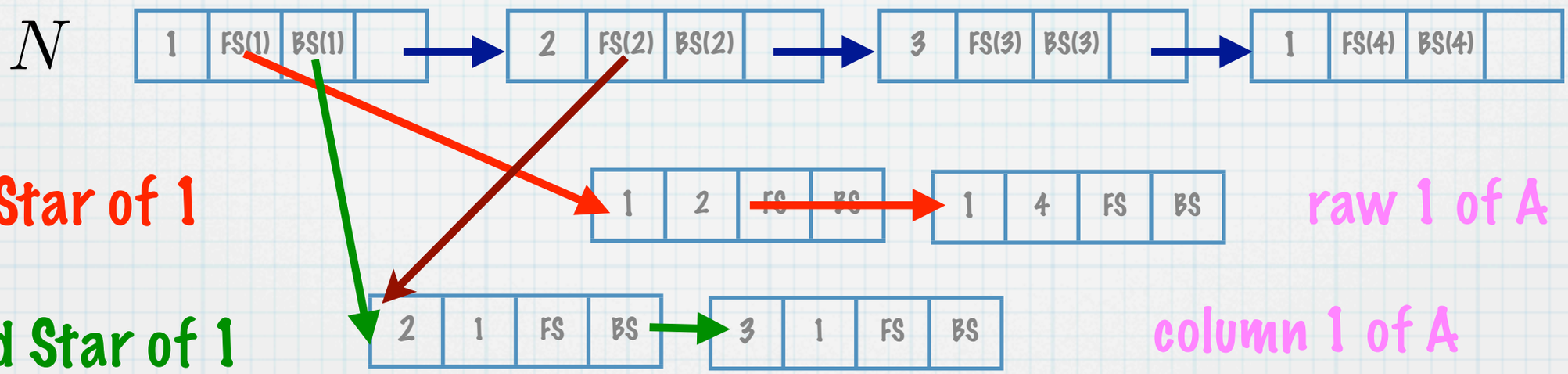
raw 1 of A

column 1 of A

How to represent graphs (2)

Adjacency lists





Forward Star of 1

Backward Star of 1

Forward Star of 2

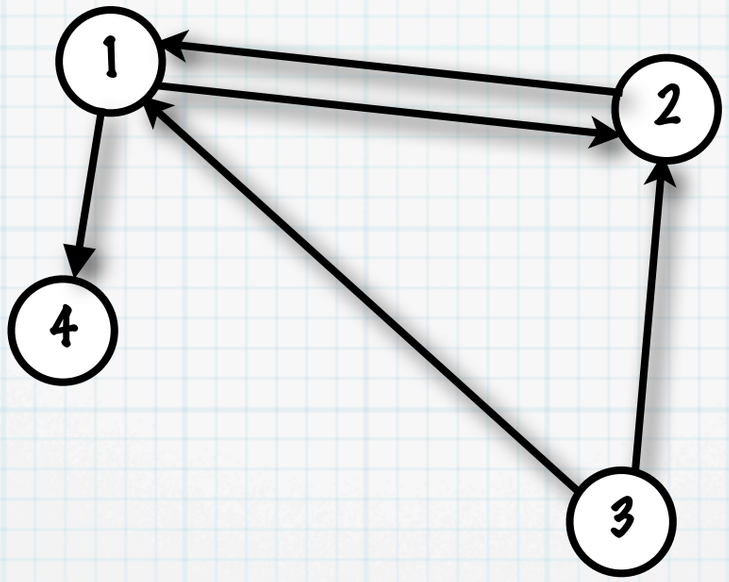
Backward Star of 2

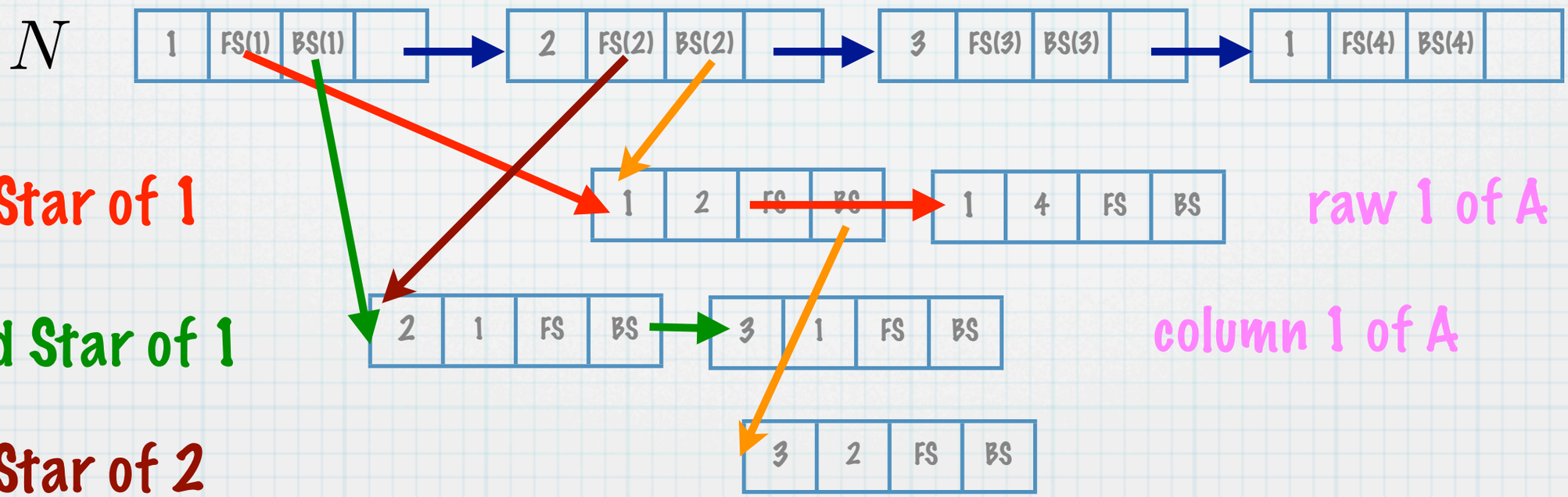
raw 1 of A

column 1 of A

How to represent graphs (2)

Adjacency lists





Forward Star of 1

Backward Star of 1

Forward Star of 2

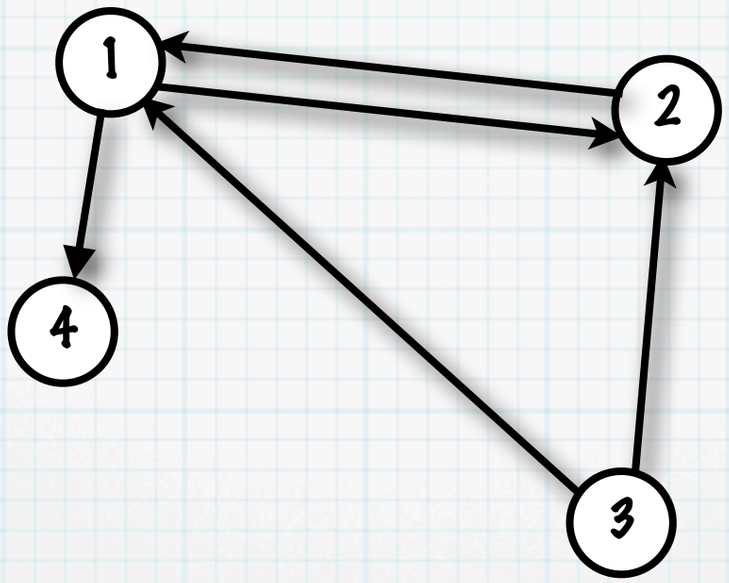
Backward Star of 2

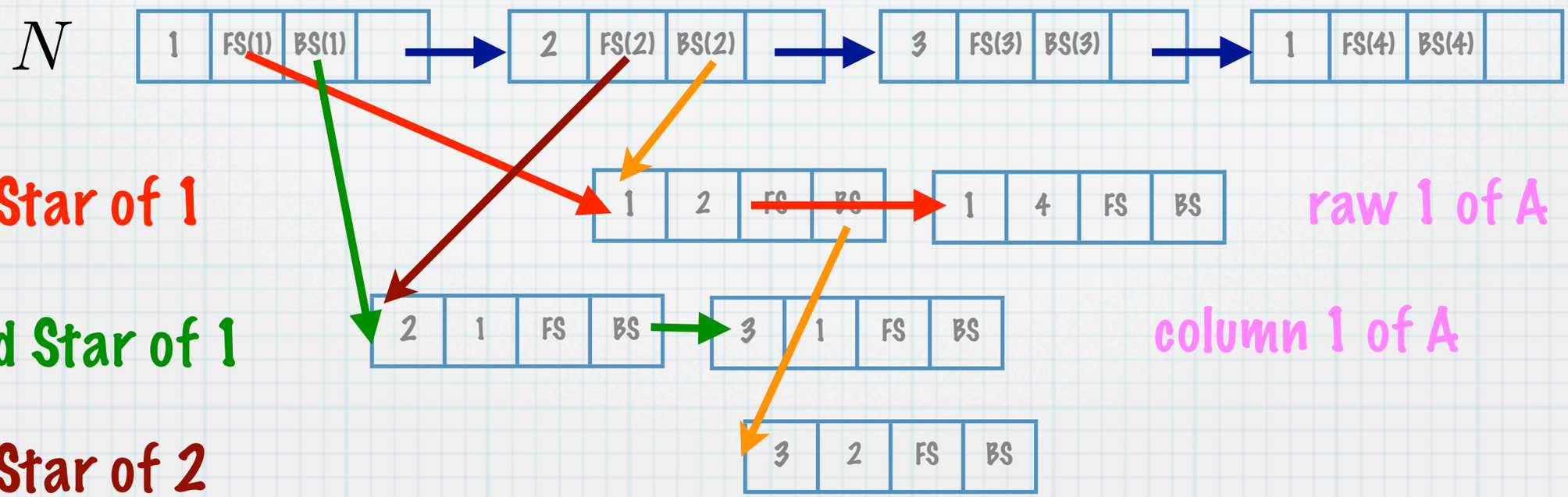
raw 1 of A

column 1 of A

How to represent graphs (2)

Adjacency lists





Forward Star of 1

Backward Star of 1

Forward Star of 2

Backward Star of 2

raw 1 of A

column 1 of A

More suitable for sparse graphs

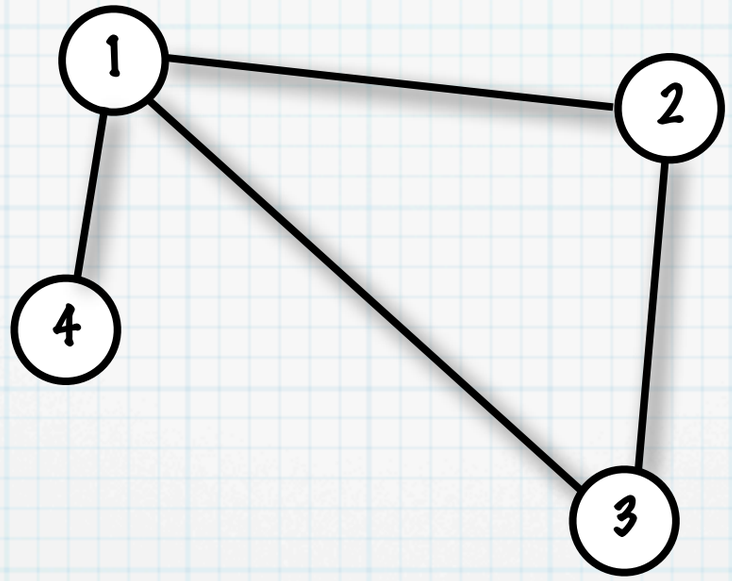
How to represent graphs (2)

How to represent graphs (2)

Adjacency lists

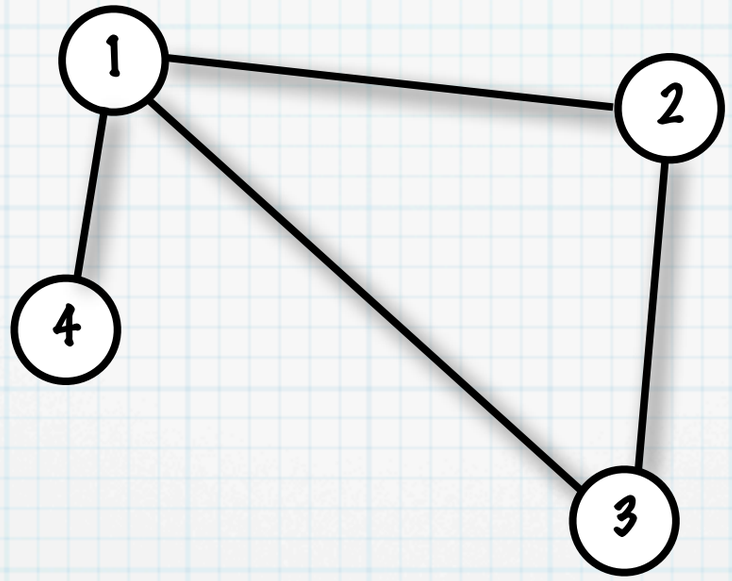
How to represent graphs (2)

Adjacency lists



How to represent graphs (2)

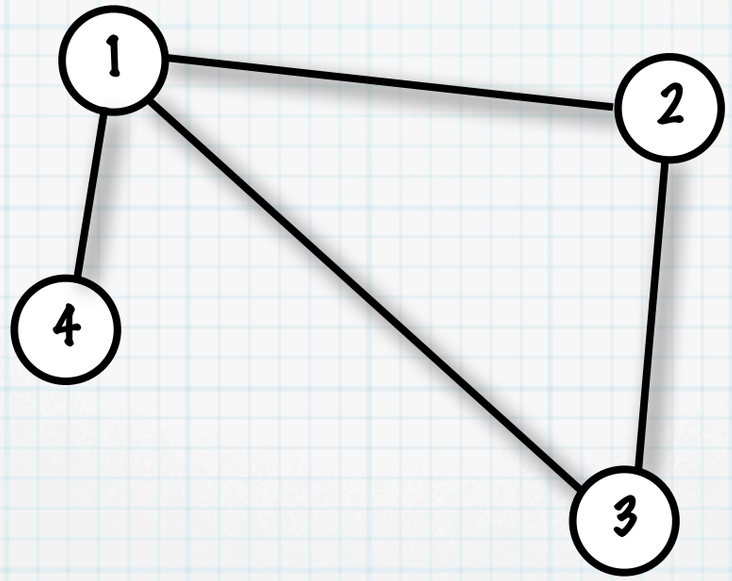
Adjacency lists



We do not distinguish between FS and BS

How to represent graphs (2)

Adjacency lists



We do not distinguish between FS and BS

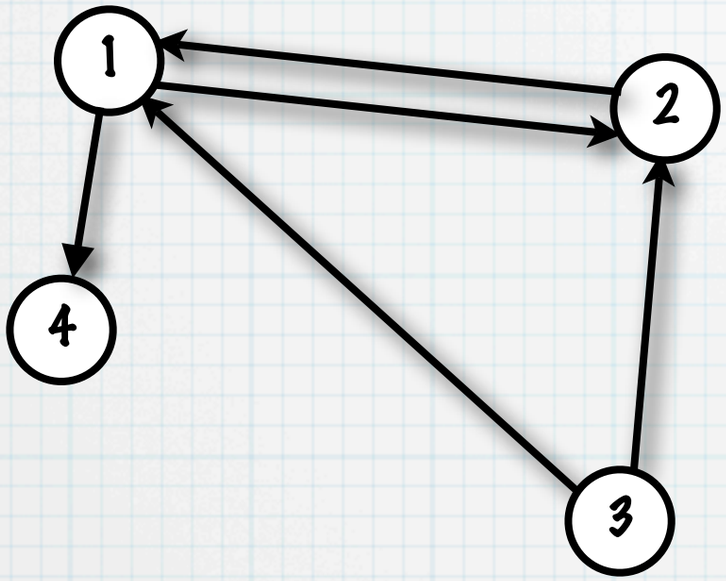
List of incident arcs: Star

How to represent graphs (3)

Node/arc incidence matrix E

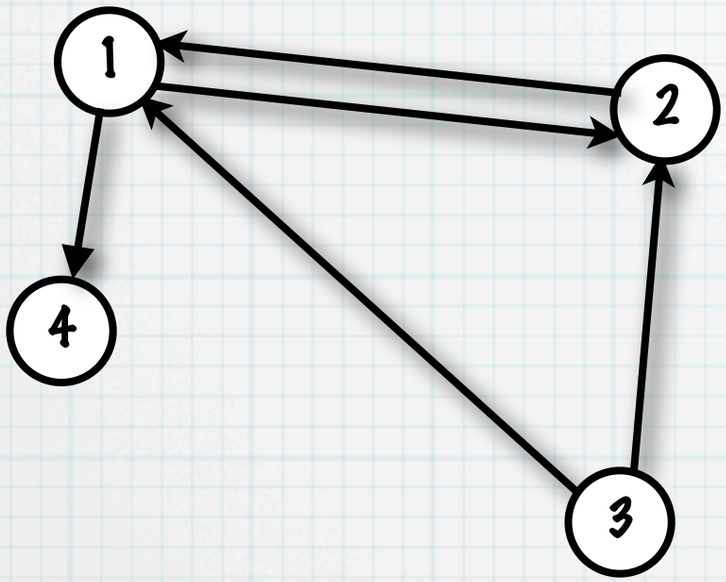
How to represent graphs (3)

Node/arc incidence matrix E



How to represent graphs (3)

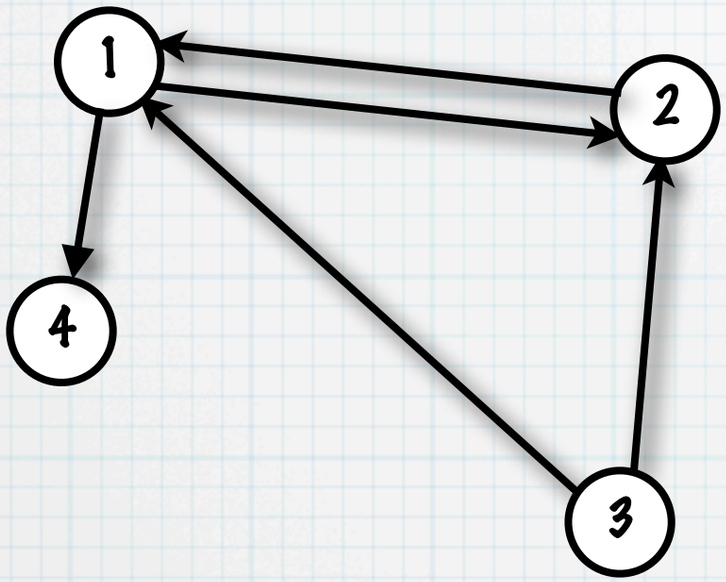
Node/arc incidence matrix E



-1: tail of the arc

How to represent graphs (3)

Node/arc incidence matrix E

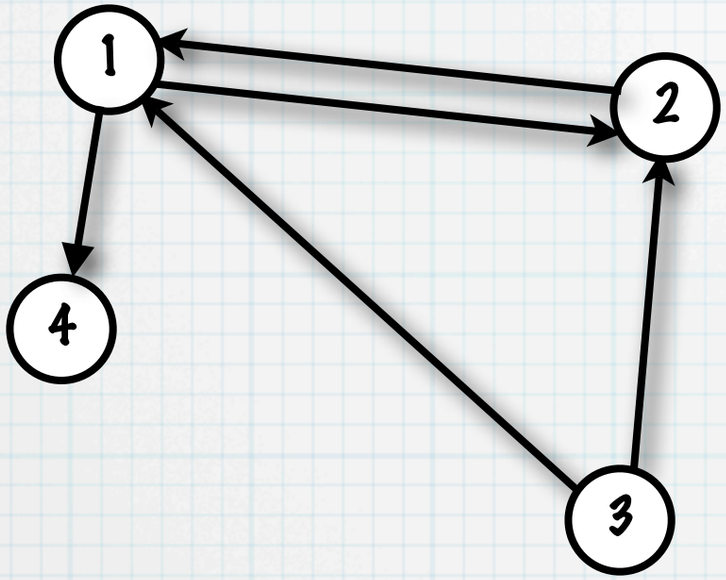


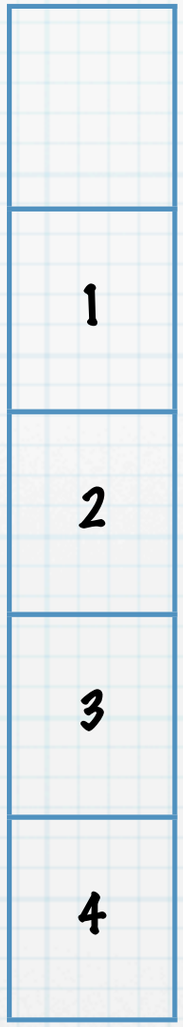
-1: tail of the arc

1: head of the arc

How to represent graphs (3)

Node/arc incidence matrix E



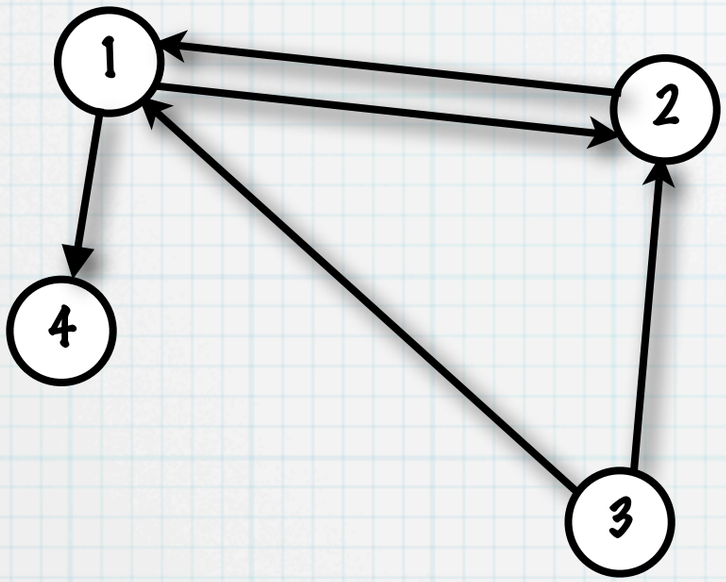


-1: tail of the arc

1: head of the arc

How to represent graphs (3)

Node/arc incidence matrix E



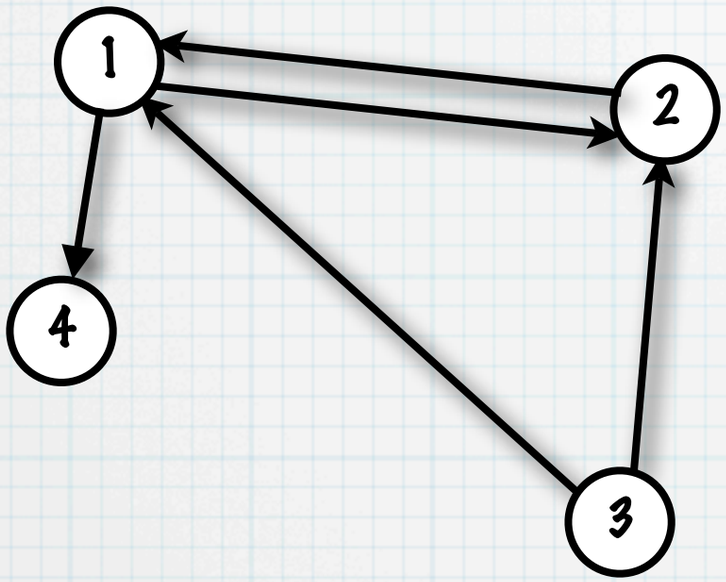
	(1, 2)
1	-1
2	1
3	0
4	0

-1: tail of the arc

1: head of the arc

How to represent graphs (3)

Node/arc incidence matrix E



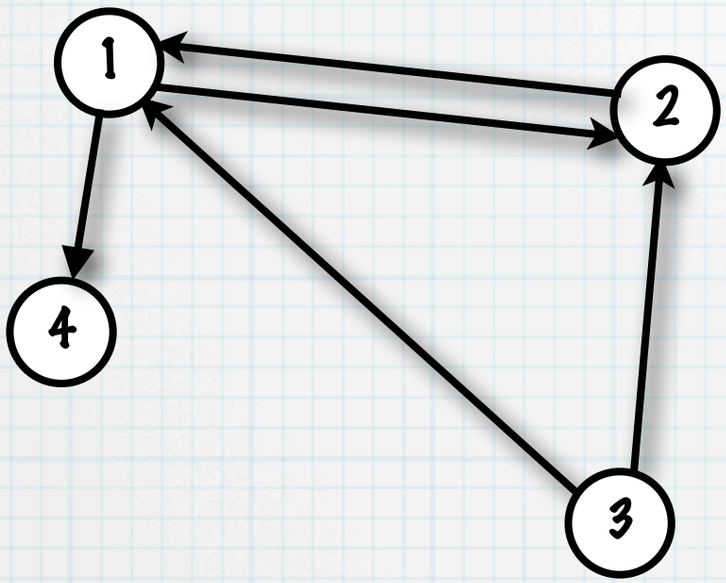
	(1, 2)	(2, 1)
1	-1	1
2	1	-1
3	0	0
4	0	0

-1: tail of the arc

1: head of the arc

How to represent graphs (3)

Node/arc incidence matrix E



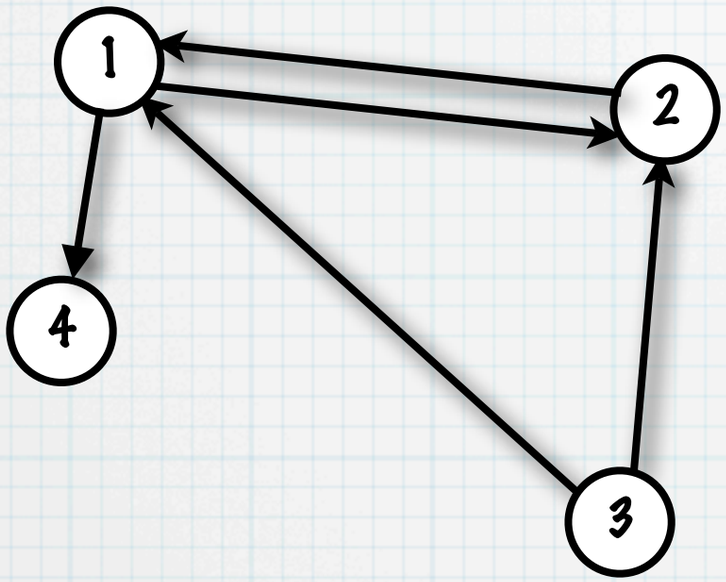
	(1, 2)	(2, 1)	(1, 4)
1	-1	1	-1
2	1	-1	0
3	0	0	0
4	0	0	1

-1: tail of the arc

1: head of the arc

How to represent graphs (3)

Node/arc incidence matrix E



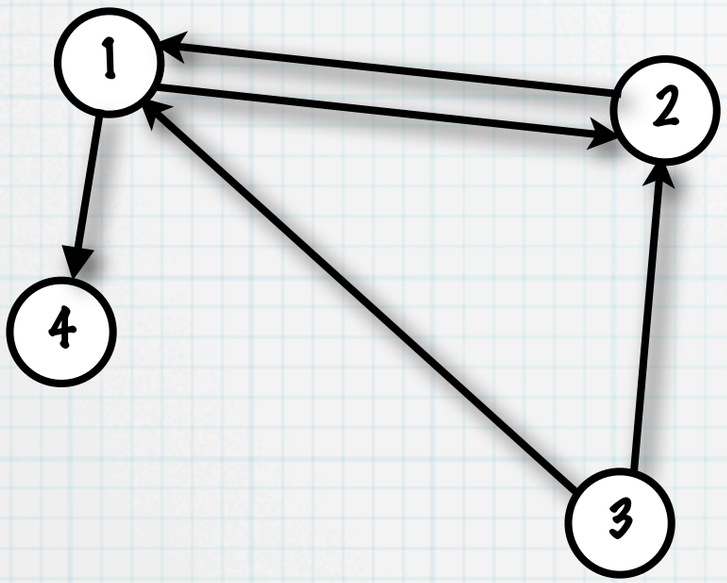
	(1, 2)	(2, 1)	(1, 4)	(3, 2)
1	-1	1	-1	0
2	1	-1	0	1
3	0	0	0	-1
4	0	0	1	0

-1: tail of the arc

1: head of the arc

How to represent graphs (3)

Node/arc incidence matrix E



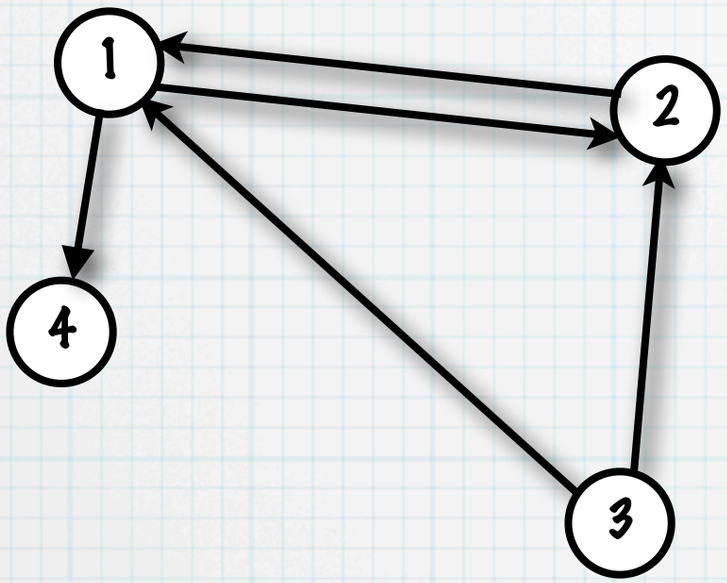
	(1, 2)	(2, 1)	(1, 4)	(3, 2)	(3, 1)
1	-1	1	-1	0	1
2	1	-1	0	1	0
3	0	0	0	-1	-1
4	0	0	1	0	0

-1: tail of the arc

1: head of the arc

How to represent graphs (3)

Node/arc incidence matrix E



	(1, 2)	(2, 1)	(1, 4)	(3, 2)	(3, 1)
1	-1	1	-1	0	1
2	1	-1	0	1	0
3	0	0	0	-1	-1
4	0	0	1	0	0

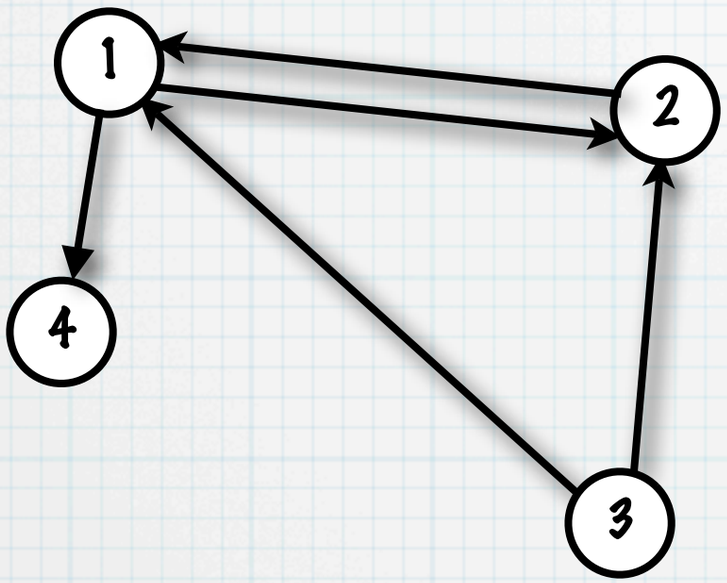
-1: tail of the arc

1: head of the arc

used in certain mathematical formulations

How to represent graphs (3)

Node/arc incidence matrix E



	(1, 2)	(2, 1)	(1, 4)	(3, 2)	(3, 1)
1	-1	1	-1	0	1
2	1	-1	0	1	0
3	0	0	0	-1	-1
4	0	0	1	0	0

-1: tail of the arc

1: head of the arc

used in certain mathematical formulations

undirected case: no distinction of sign

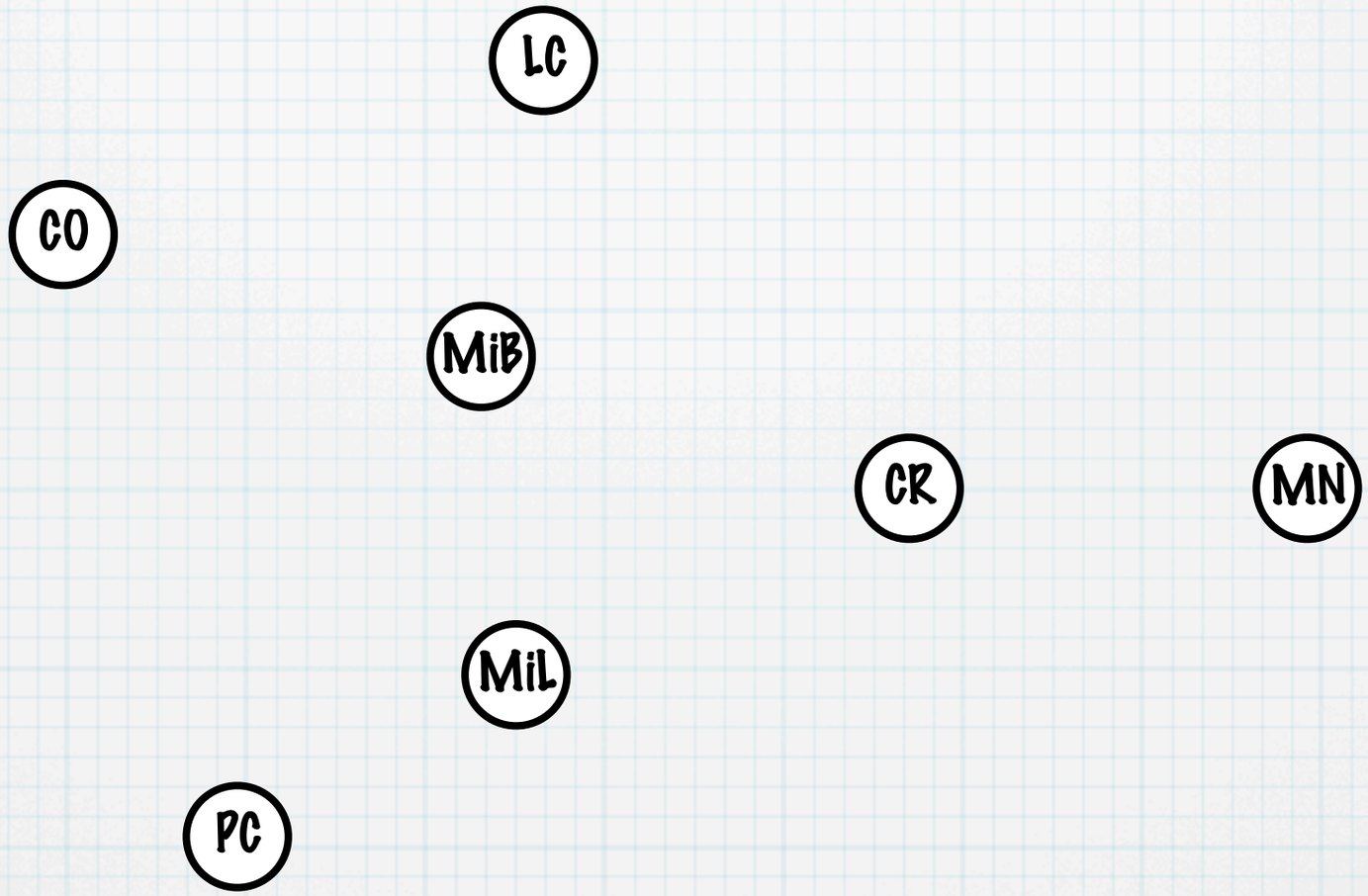
Example: TLC network design

We want to design the Intranet network of Politecnico

Example: TLC network design

We want to design the Intranet network of Politecnico

Politecnico's centers

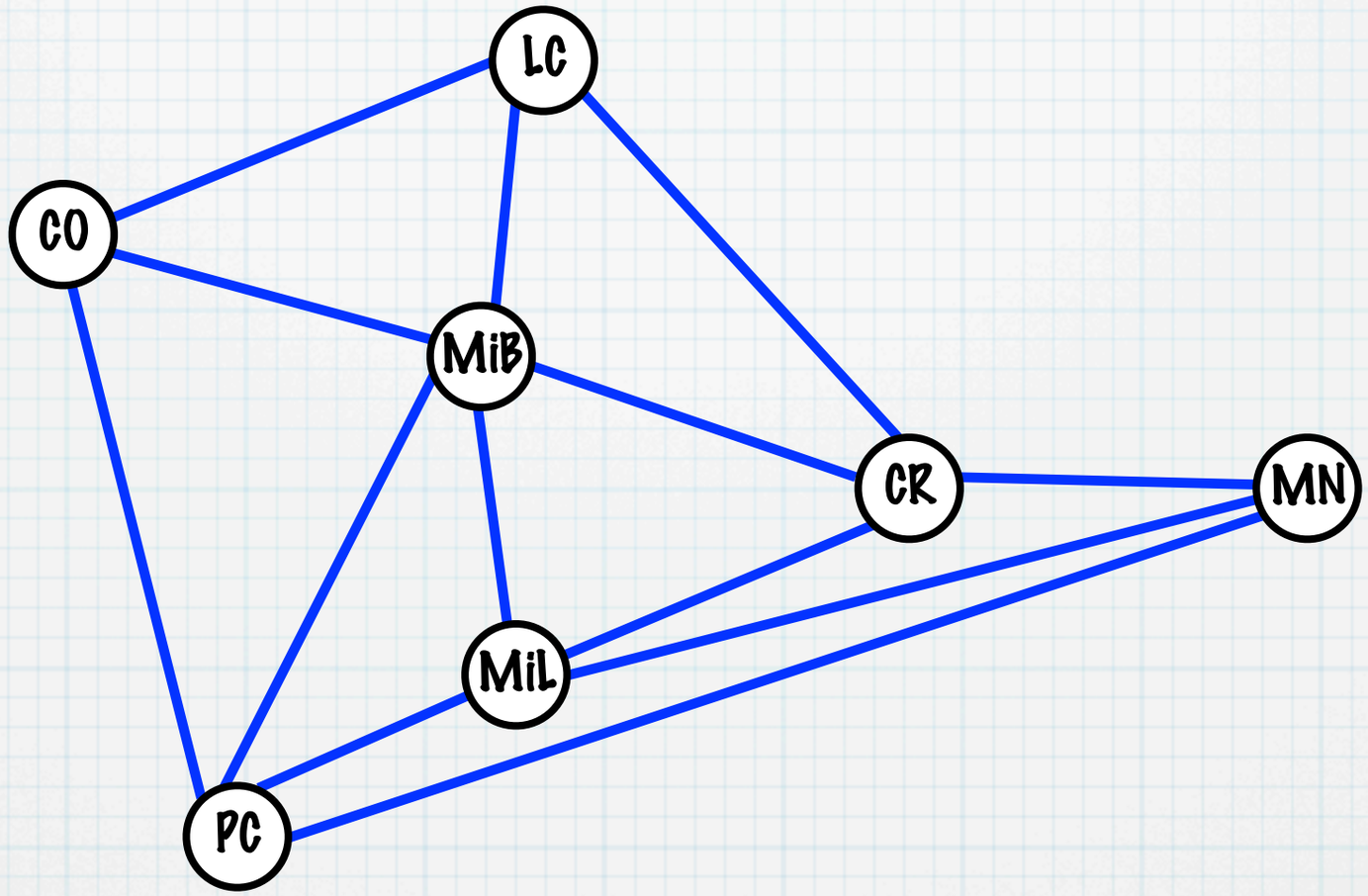


Example: TLC network design

We want to design the Intranet network of Politecnico

Politecnico's centers

Potential links



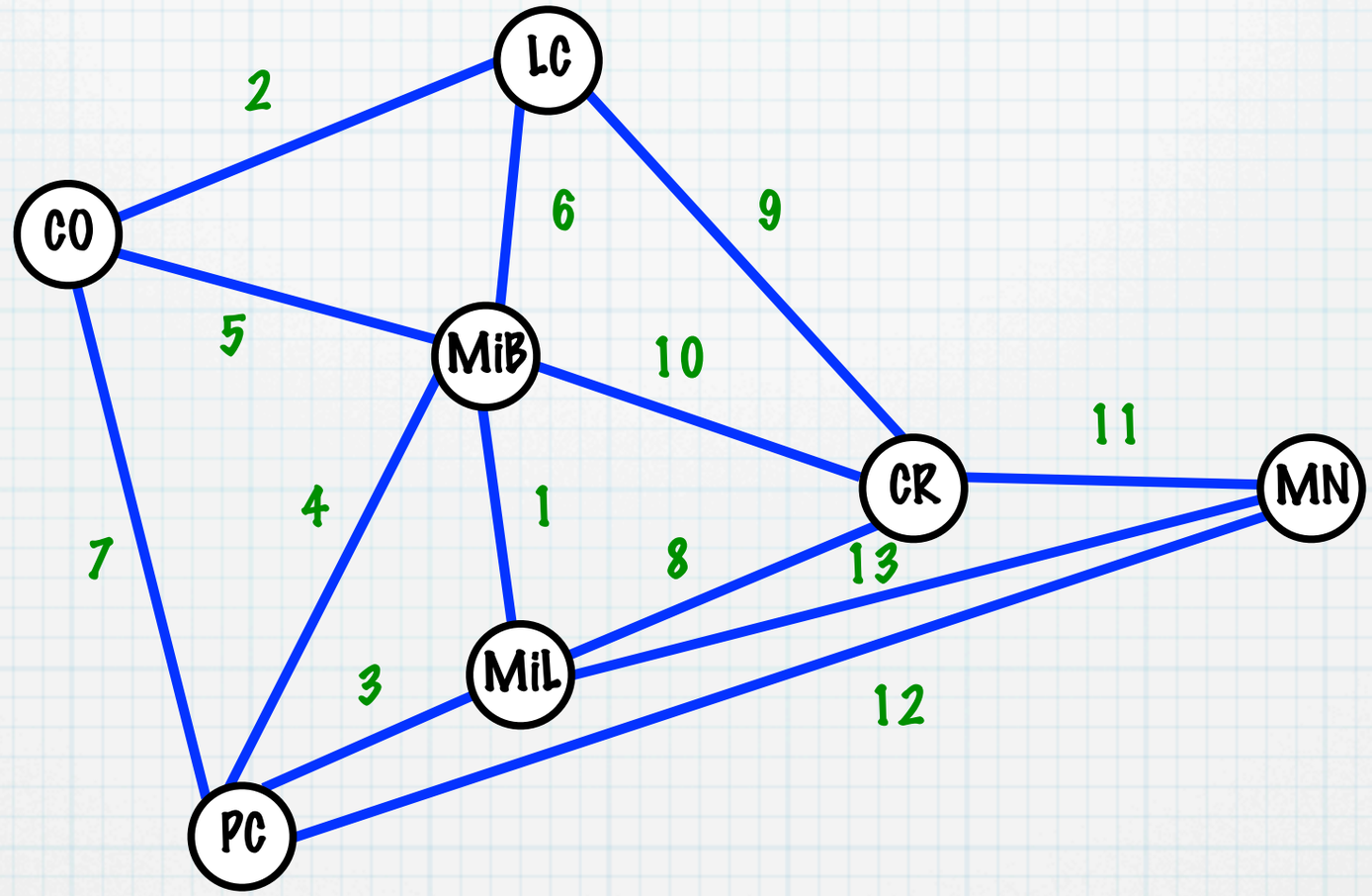
Example: TLC network design

We want to design the Intranet network of Politecnico

Politecnico's centers

Potential links

Installation costs

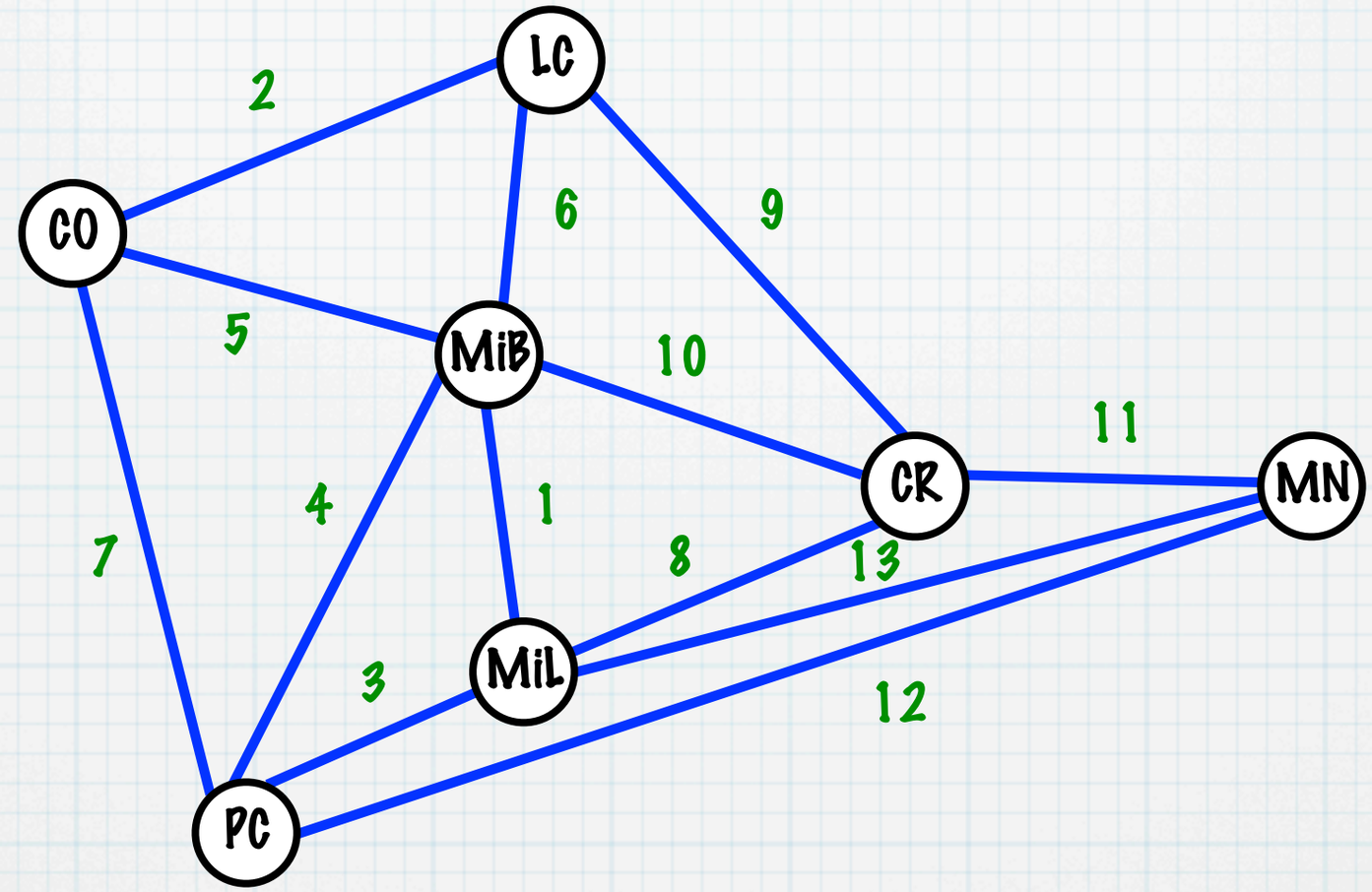


Example: TLC network design

We want to design the Intranet network of Politecnico

Politecnico's centers

Potential links



Installation costs

Problem: select a subset of arcs so that each node is **connected** to any other node by at least one path, minimizing the installation cost

Abstraction of the problem

Abstraction of the problem

Given an undirected graph $G = (N, A)$

Abstraction of the problem

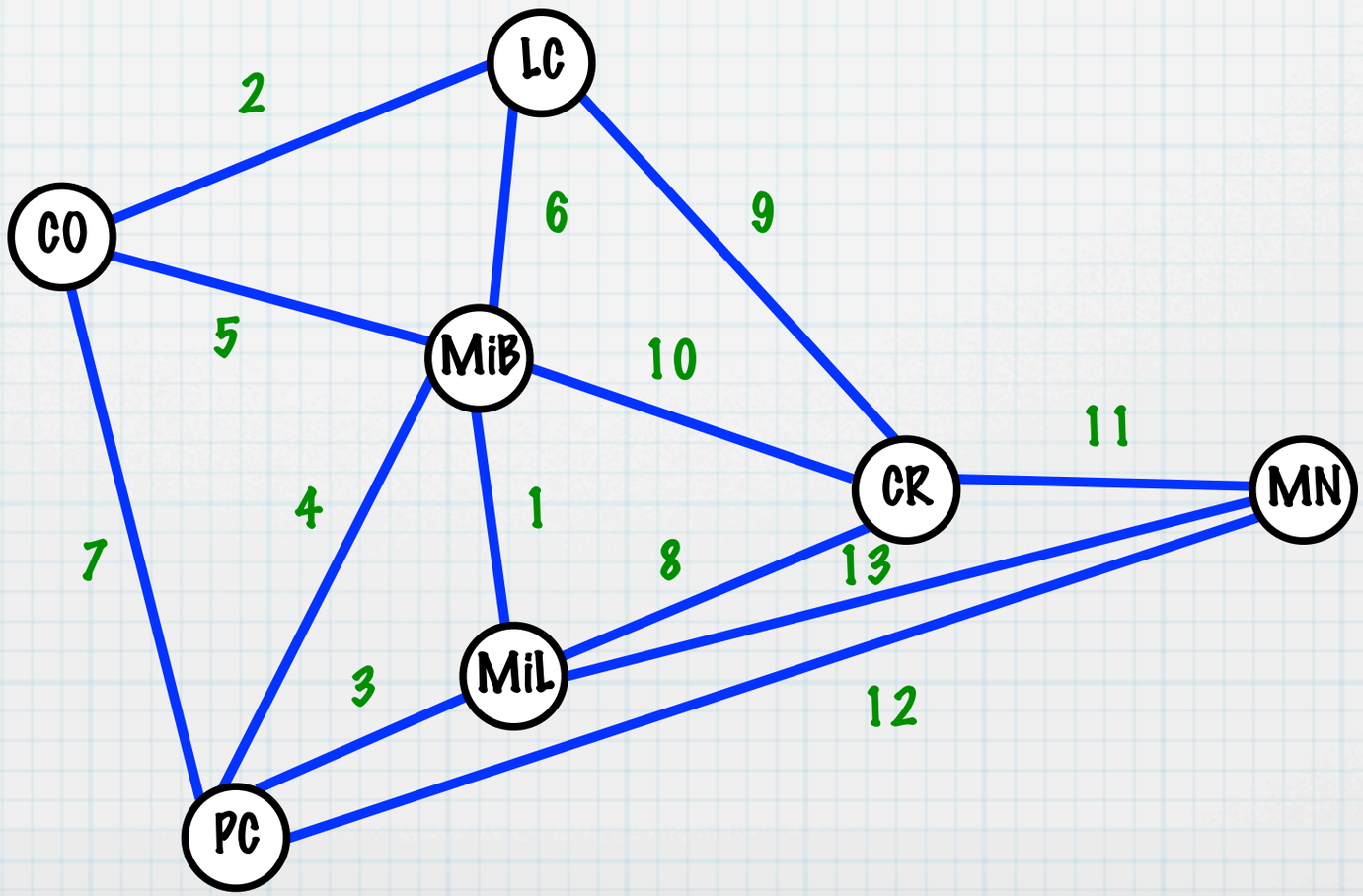
Given an undirected graph $G = (N, A)$

with weights on the arcs $w_{ij} \geq 0 \; \forall (i, j) \in A$

Abstraction of the problem

Given an undirected graph $G = (N, A)$

with weights on the arcs $w_{ij} \geq 0 \; \forall (i, j) \in A$

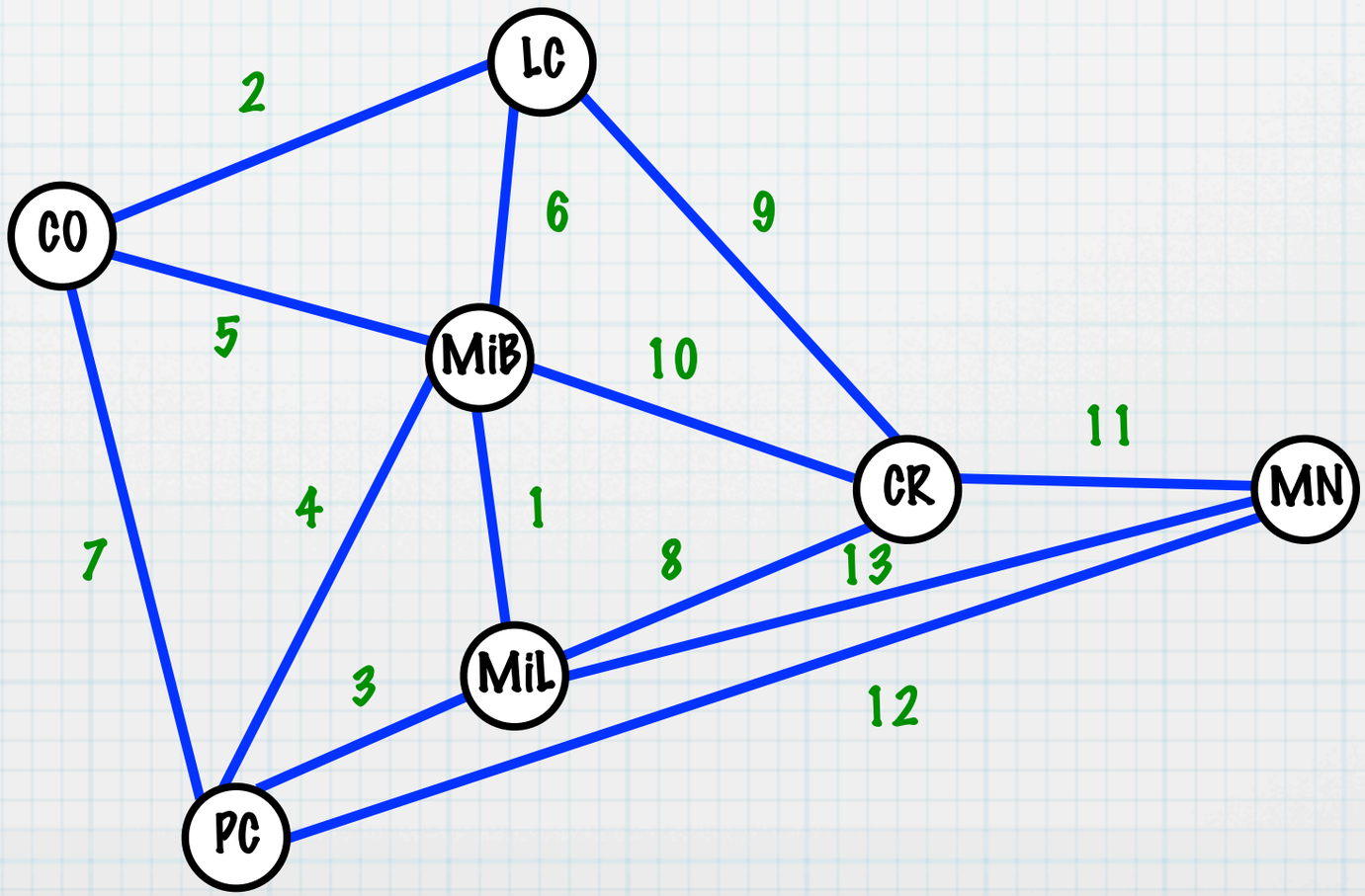


Abstraction of the problem

Given an undirected graph $G = (N, A)$

with weights on the arcs $w_{ij} \geq 0 \; \forall (i, j) \in A$

find a connected subgraph $G' = (N, A'), A' \subseteq A$

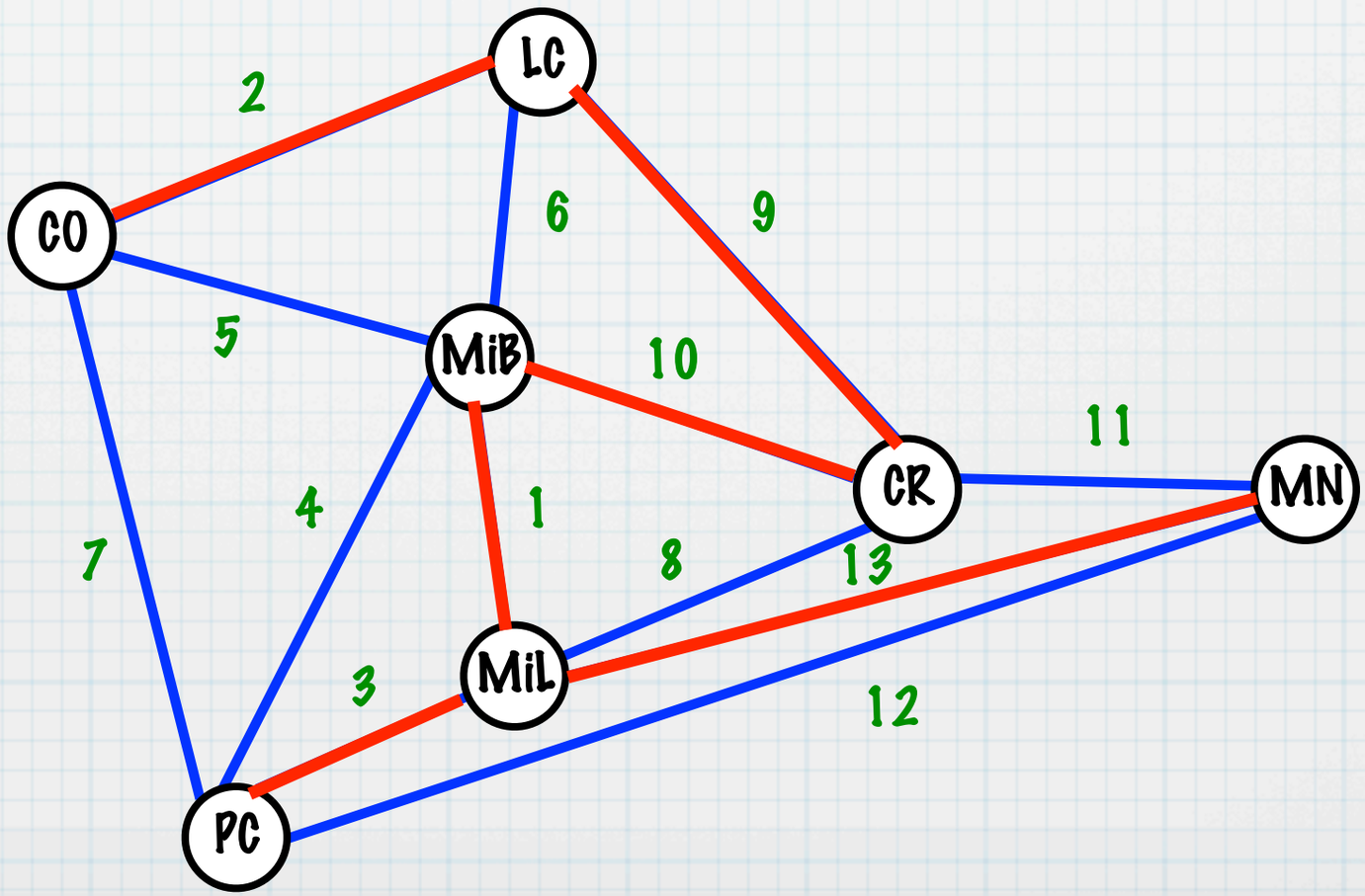


Abstraction of the problem

Given an undirected graph $G = (N, A)$

with weights on the arcs $w_{ij} \geq 0 \; \forall (i,j) \in A$

find a connected subgraph $G' = (N, A'), A' \subseteq A$



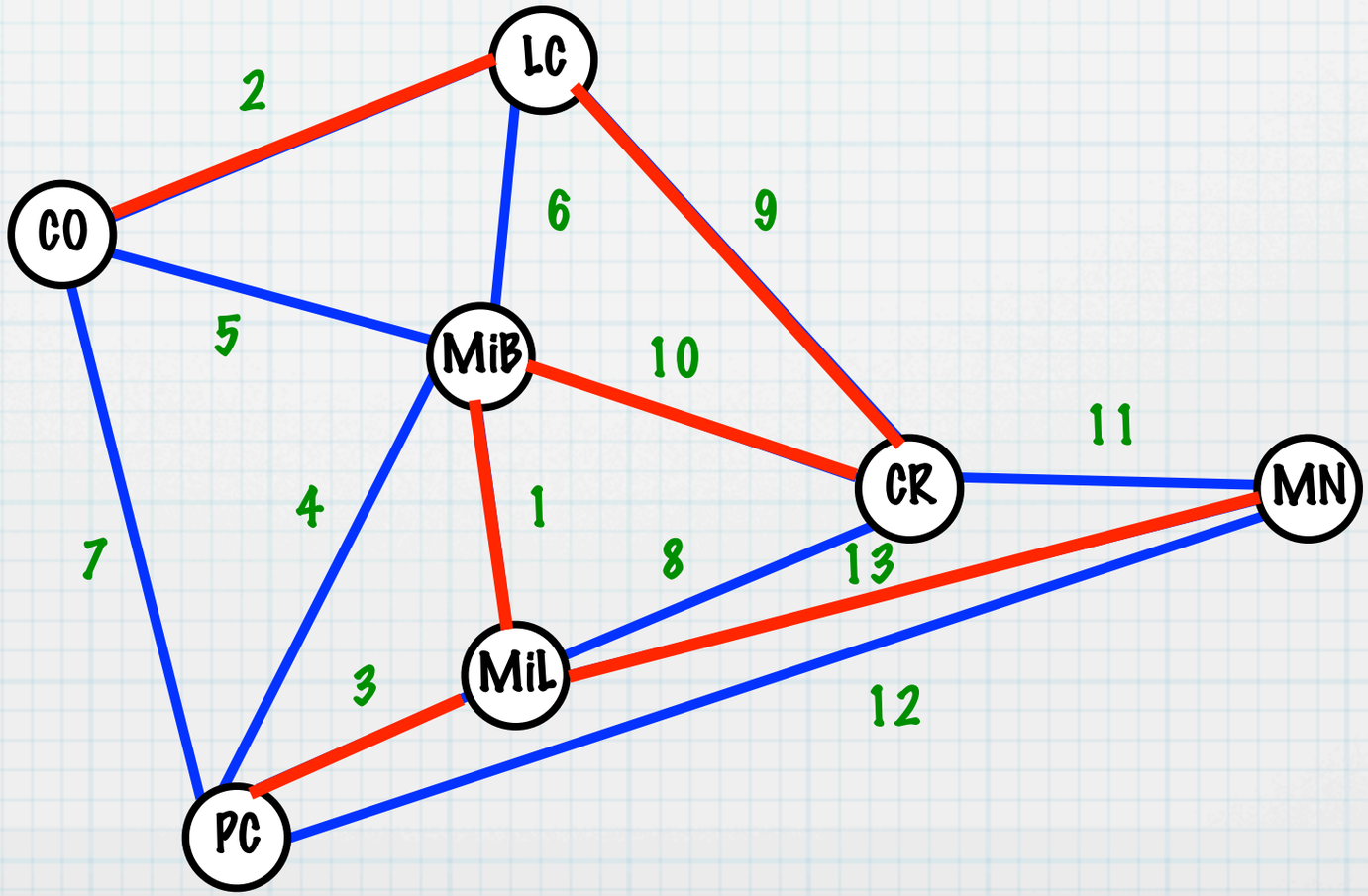
Abstraction of the problem

Given an undirected graph $G = (N, A)$

with weights on the arcs $w_{ij} \geq 0 \; \forall (i,j) \in A$

find a connected subgraph $G' = (N, A'), A' \subseteq A$

minimizing $\sum_{(i,j) \in A'} w_{ij}$



Properties of minimal feasible solutions

Properties of minimal feasible solutions

* No cycles: otherwise we could remove some arcs

Properties of minimal feasible solutions

* No cycles: otherwise we could remove some arcs
* G' is connected

Properties of minimal feasible solutions

* No cycles: otherwise we could remove some arcs
* G' is connected
* If |N| = n then |A'| = n-1

Properties of minimal feasible solutions

* No cycles: otherwise we could remove some arcs
* G' is connected
* If |N| = n then |A'| = n-1
* G' is a spanning tree

Properties of minimal feasible solutions

* No cycles: otherwise we could remove some arcs
* G' is connected
* If |N| = n then |A'| = n-1
* G' is a spanning tree
* if we add one arc to G' we obtain a (unique) cycle

Properties of minimal feasible solutions

* No cycles: otherwise we could remove some arcs
* G' is connected
* If |N| = n then |A'| = n-1
* G' is a spanning tree
* if we add one arc to G' we obtain a (unique) cycle
* if we remove one arc from G' we obtain two connected components

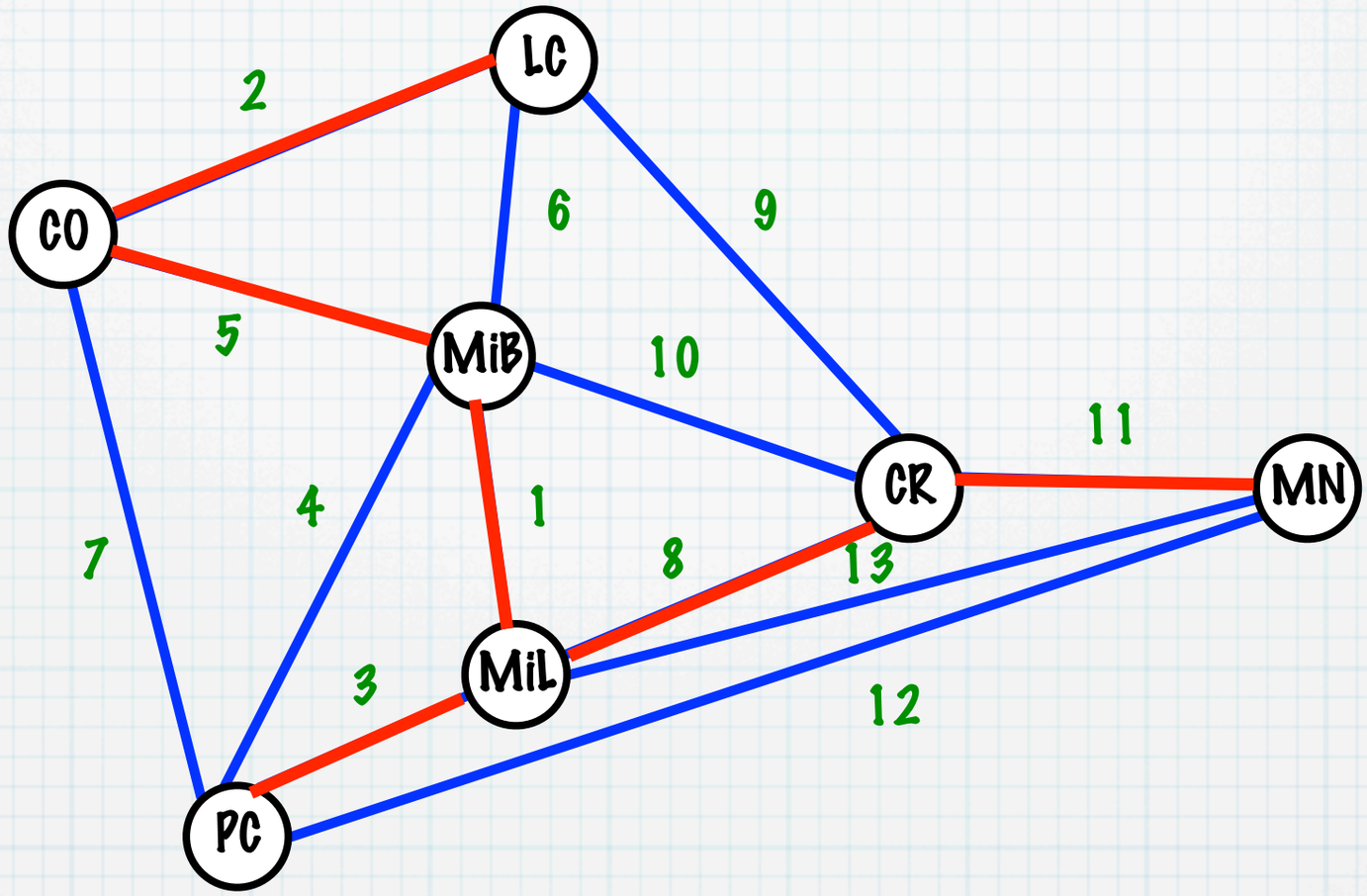
Properties of minimal feasible solutions

* No cycles: otherwise we could remove some arcs
* G' is connected
* If |N| = n then |A'| = n-1
* G' is a spanning tree
* if we add one arc to G' we obtain a (unique) cycle
* if we remove one arc from G' we obtain two connected components

Minimum Spanning Tree problem (MST)

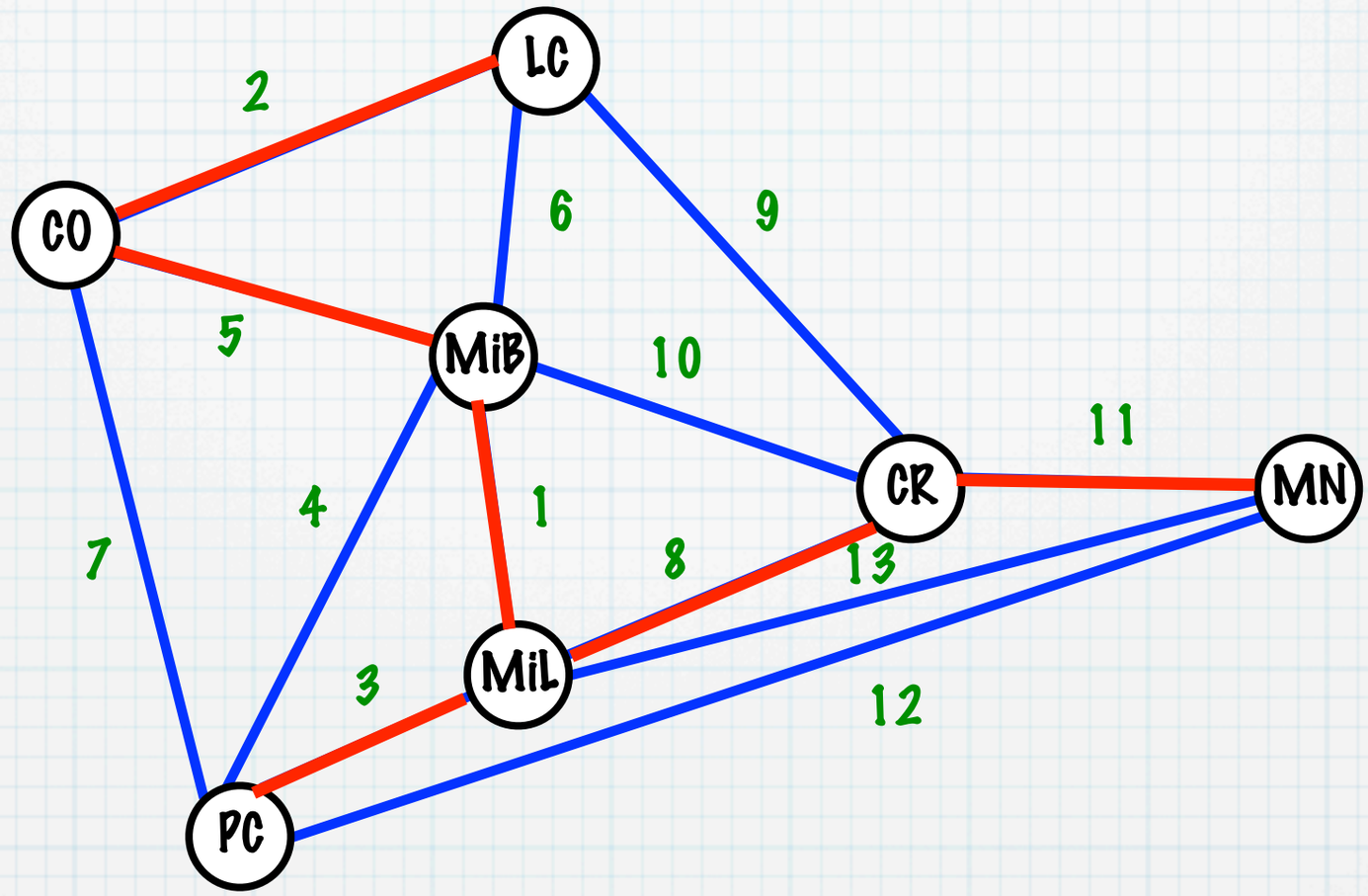
Property suggesting a solution algorithm

MST



Property suggesting a solution algorithm

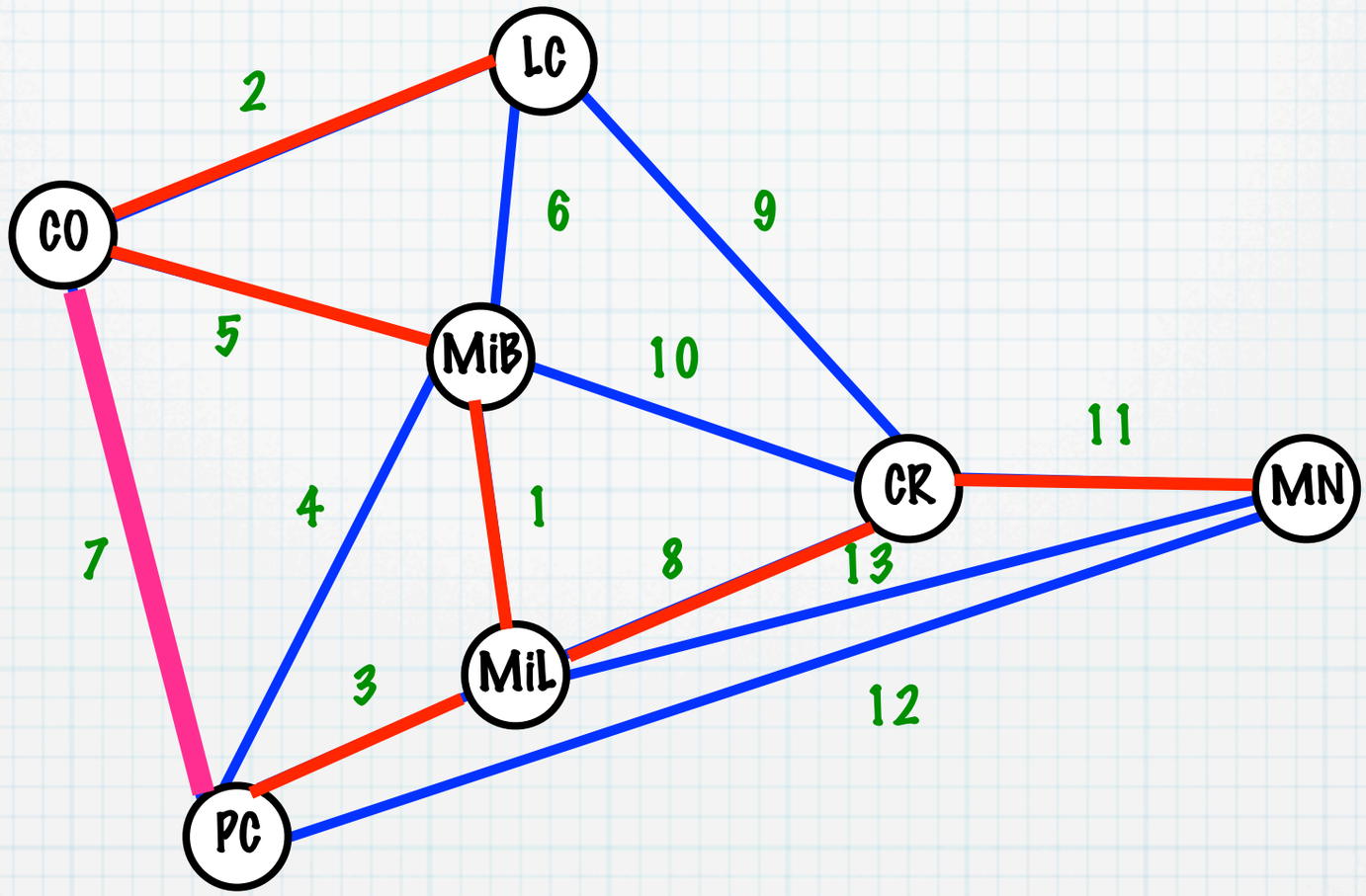
MST



Consider $A' \cup \{i, j\} : \{i, j\} \in A \setminus A'$

Property suggesting a solution algorithm

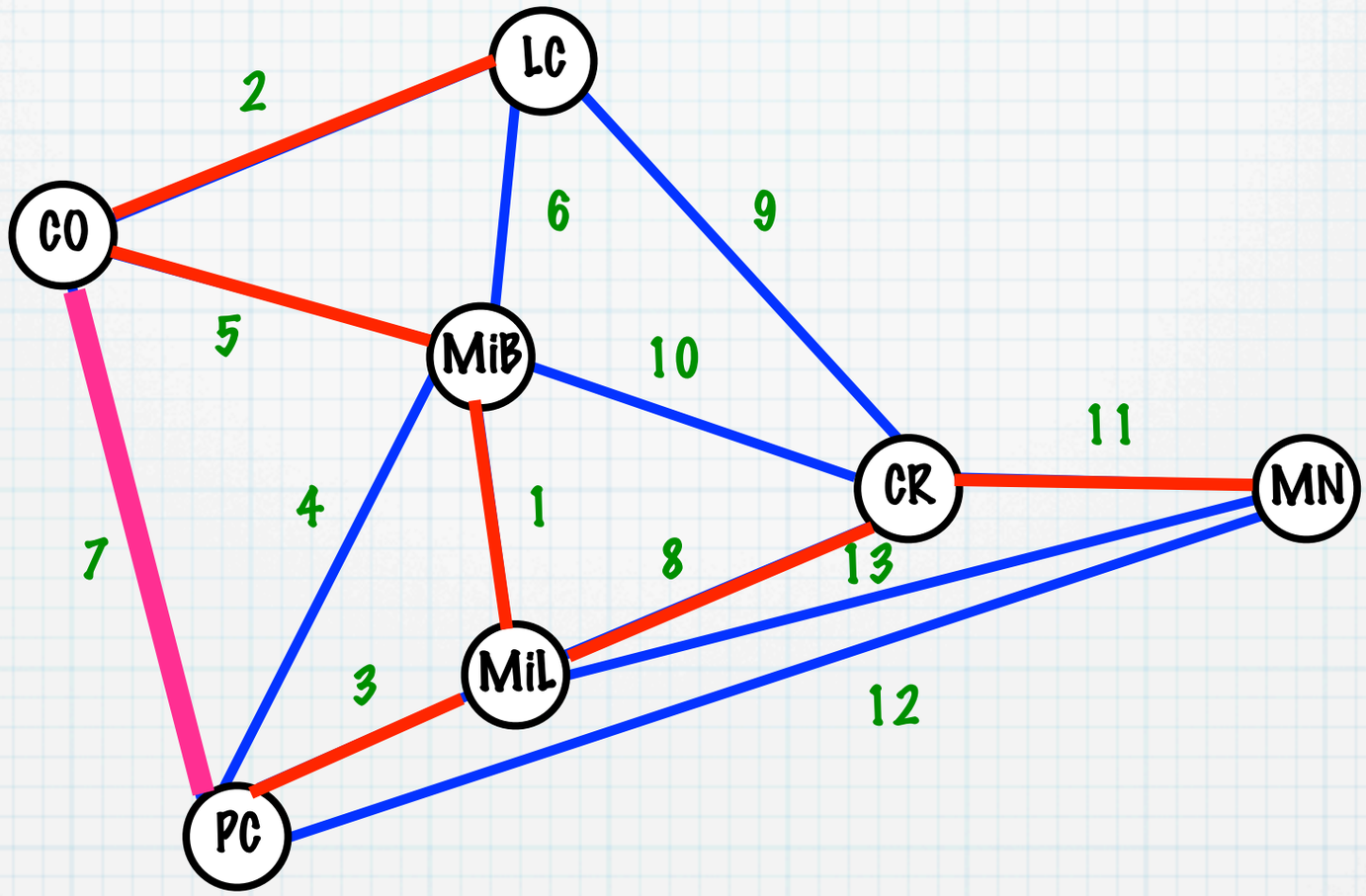
MST



Consider $A' \cup \{i, j\} : \{i, j\} \in A \setminus A'$

Property suggesting a solution algorithm

MST

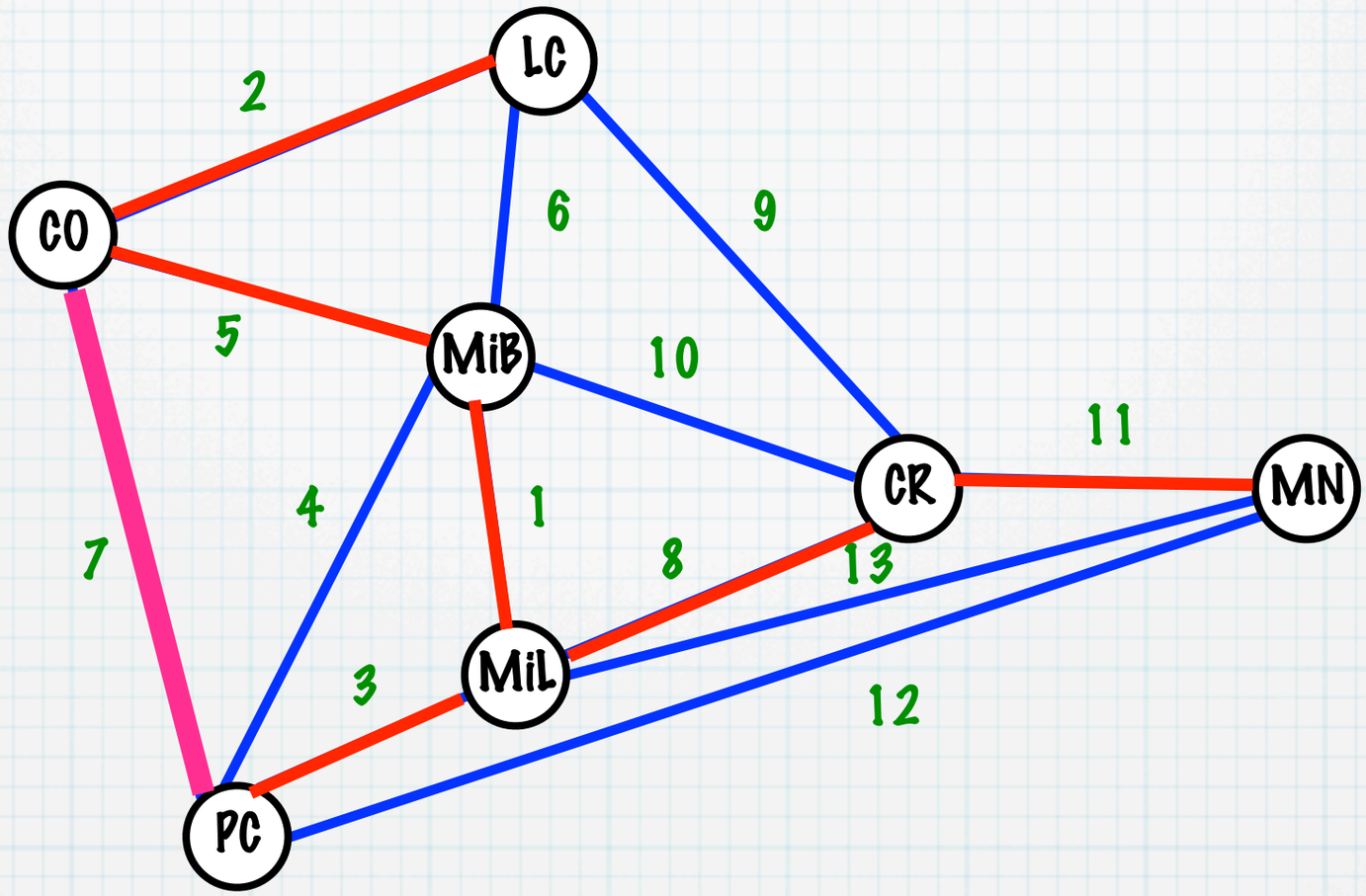


Consider $A' \cup \{i, j\} : \{i, j\} \in A \setminus A'$

the added arc induces a unique cycle C

Property suggesting a solution algorithm

MST



Consider $A' \cup \{i,j\} : \{i,j\} \in A \setminus A'$

the added arc induces a unique cycle C

then $w_{ij} \geq w_{hk} \qquad \forall \{h,k\} \in C$

Algorithm [Kruskal 1956]

Algorithm [Kruskal 1956]

* Sort arcs in increasing order of w_{ij}

Algorithm [Kruskal 1956]

* Sort arcs in increasing order of w_{ij}
* $A' = \emptyset$

Algorithm [Kruskal 1956]

* Sort arcs in increasing order of w_{ij}
* $A' = \emptyset$
* repeat

Algorithm [Kruskal 1956]

* Sort arcs in increasing order of w_{ij}
* $A' = \emptyset$
* repeat
  * consider the next arc {i,j} in the order

Algorithm [Kruskal 1956]

* Sort arcs in increasing order of w_{ij}
* $A' = \emptyset$
* repeat
  * consider the next arc $\{i,j\}$ in the order
  * **if $A' \cup \{i,j\}$ does not induce a cycle**

    then $A' \leftarrow A' \cup \{i,j\}$

Algorithm [Kruskal 1956]

* Sort arcs in increasing order of w_{ij}
* $A' = \emptyset$
* repeat
  * consider the next arc $\{i,j\}$ in the order
  * if $A' \cup \{i,j\}$ does not induce a cycle
    then $A' \leftarrow A' \cup \{i,j\}$
* until $|A'| = n-1$

Implementation

Critical point: discover the cycle

...

Implementation

Critical point: discover the cycle

...

If you want to learn more on algorithmic aspects

Implementation

Critical point: discover the cycle

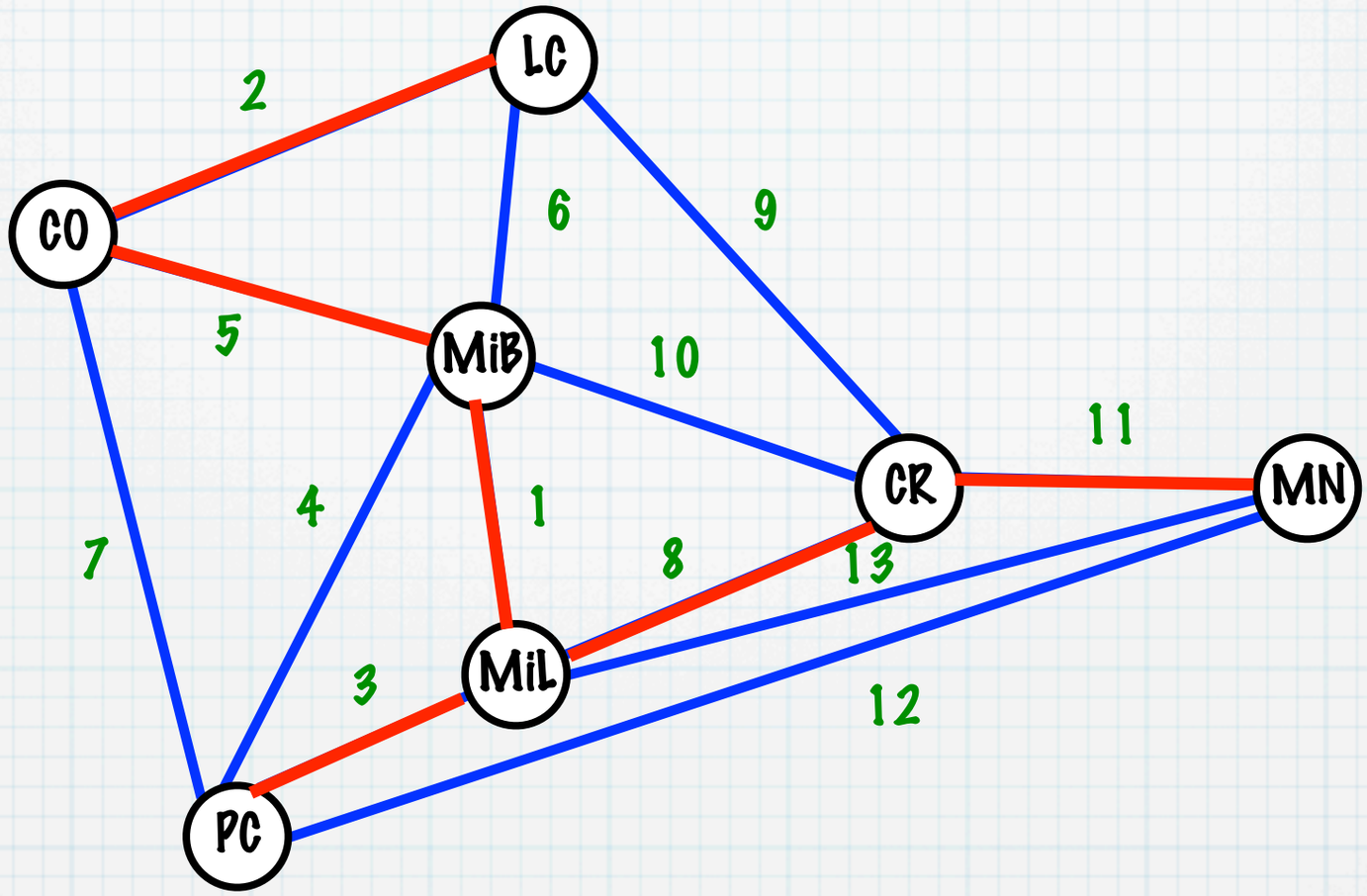
...

If you want to learn more on algorithmic aspects

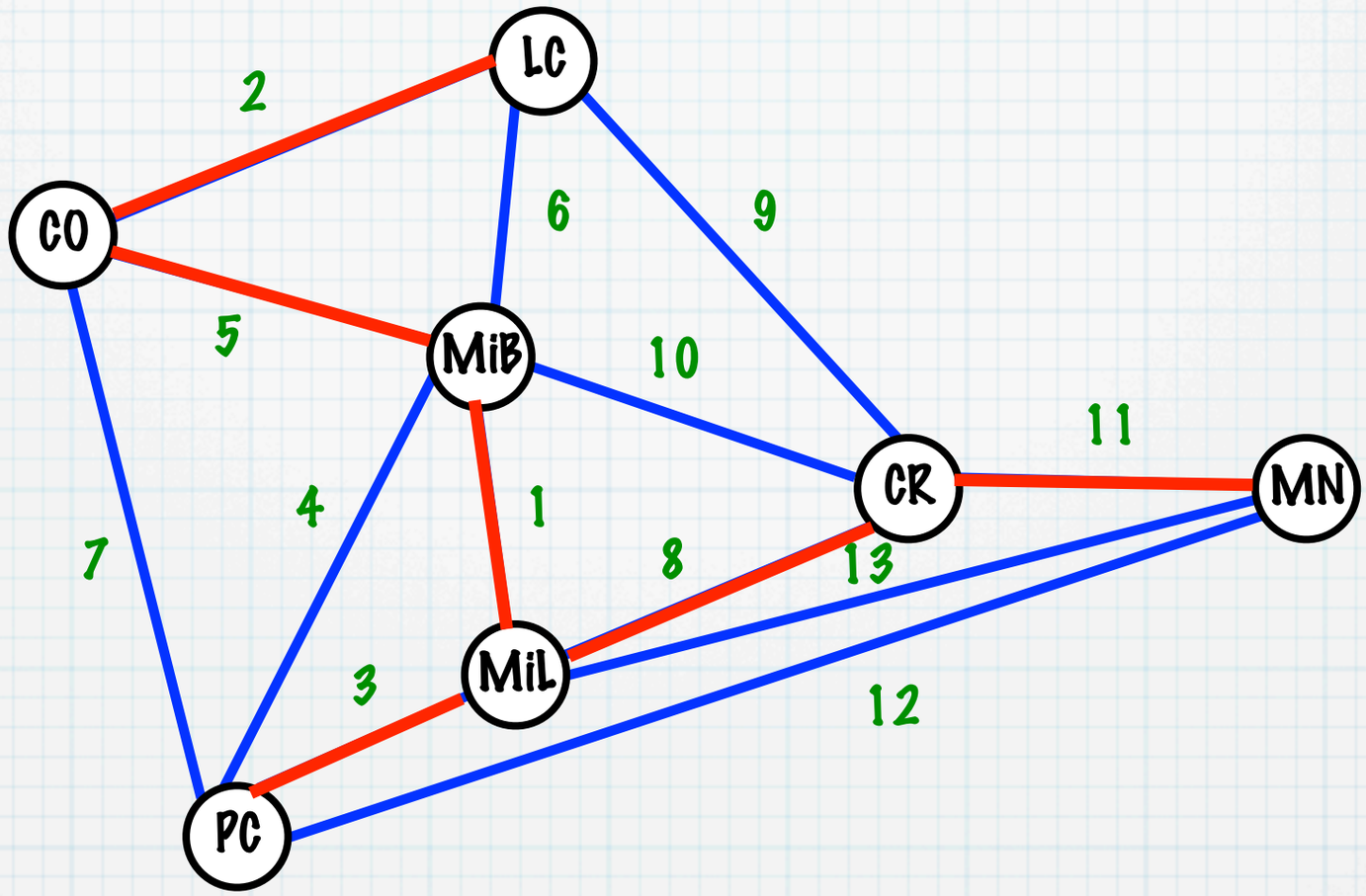
Design and Analysis of Algorithms

second semester

Property suggesting a solution algorithm

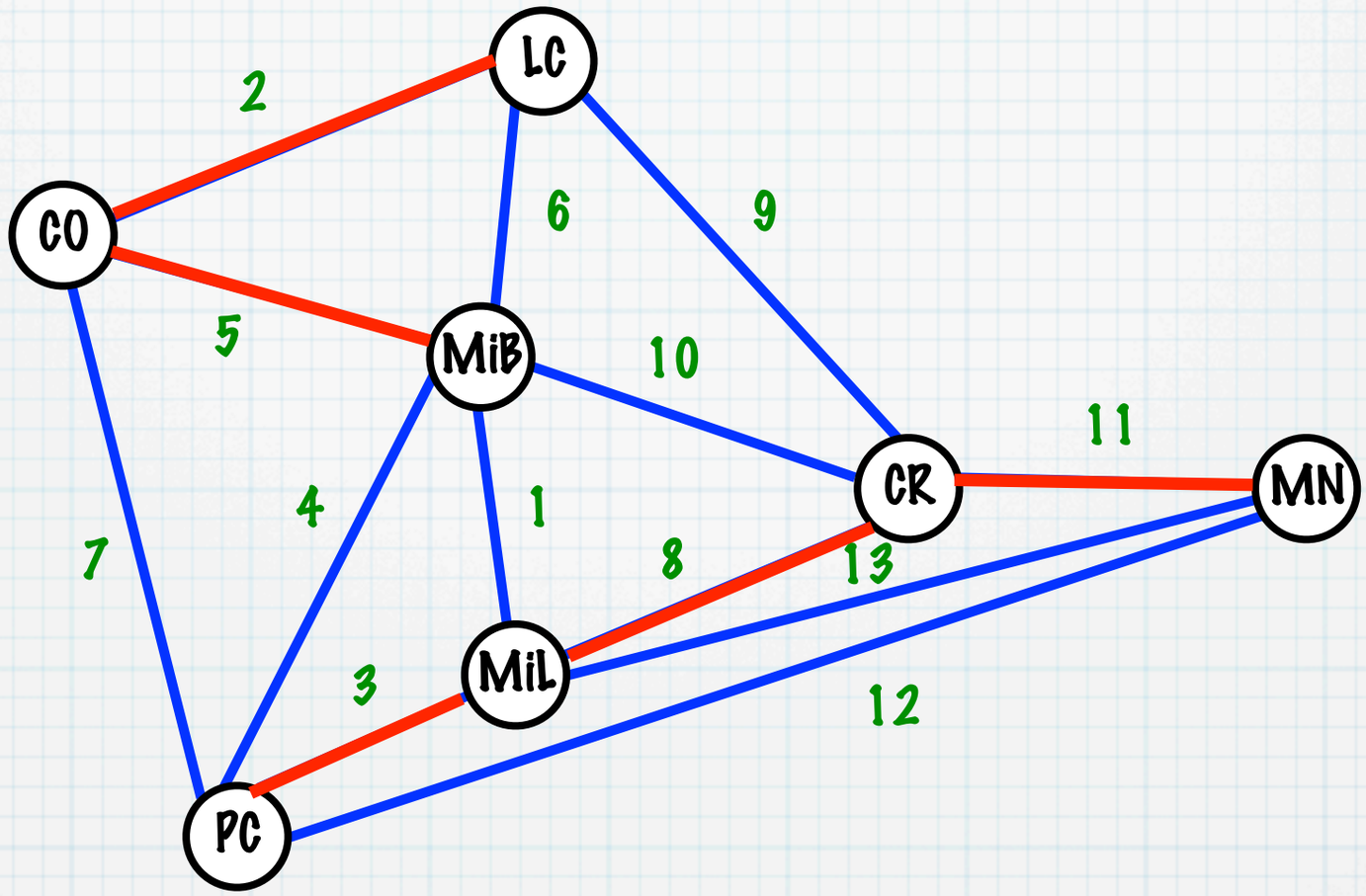


Property suggesting a solution algorithm



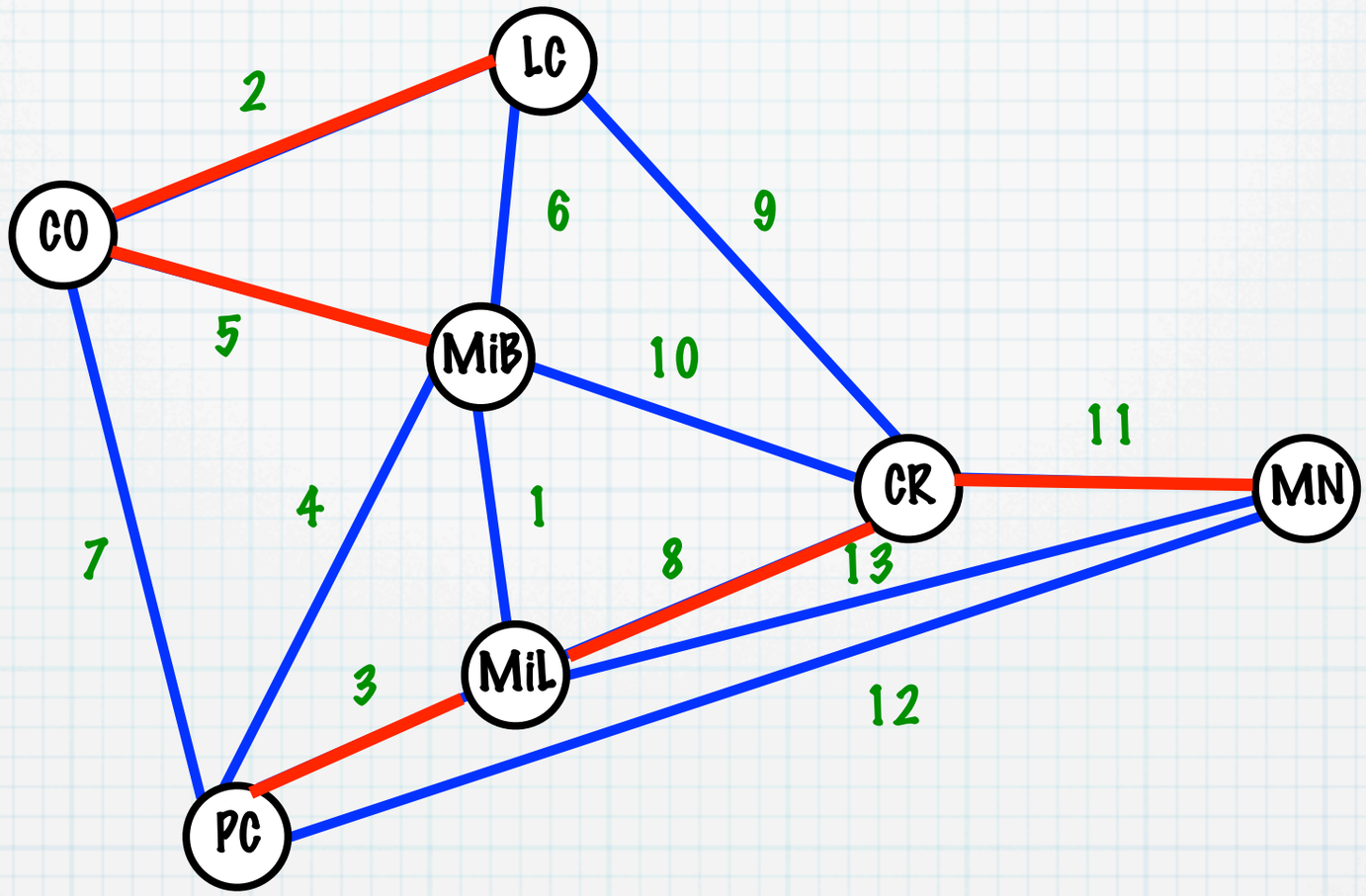
Consider $A' \setminus \{i, j\} : \{i, j\} \in A'$

Property suggesting a solution algorithm



Consider $A' \setminus \{i, j\} : \{i, j\} \in A'$

Property suggesting a solution algorithm

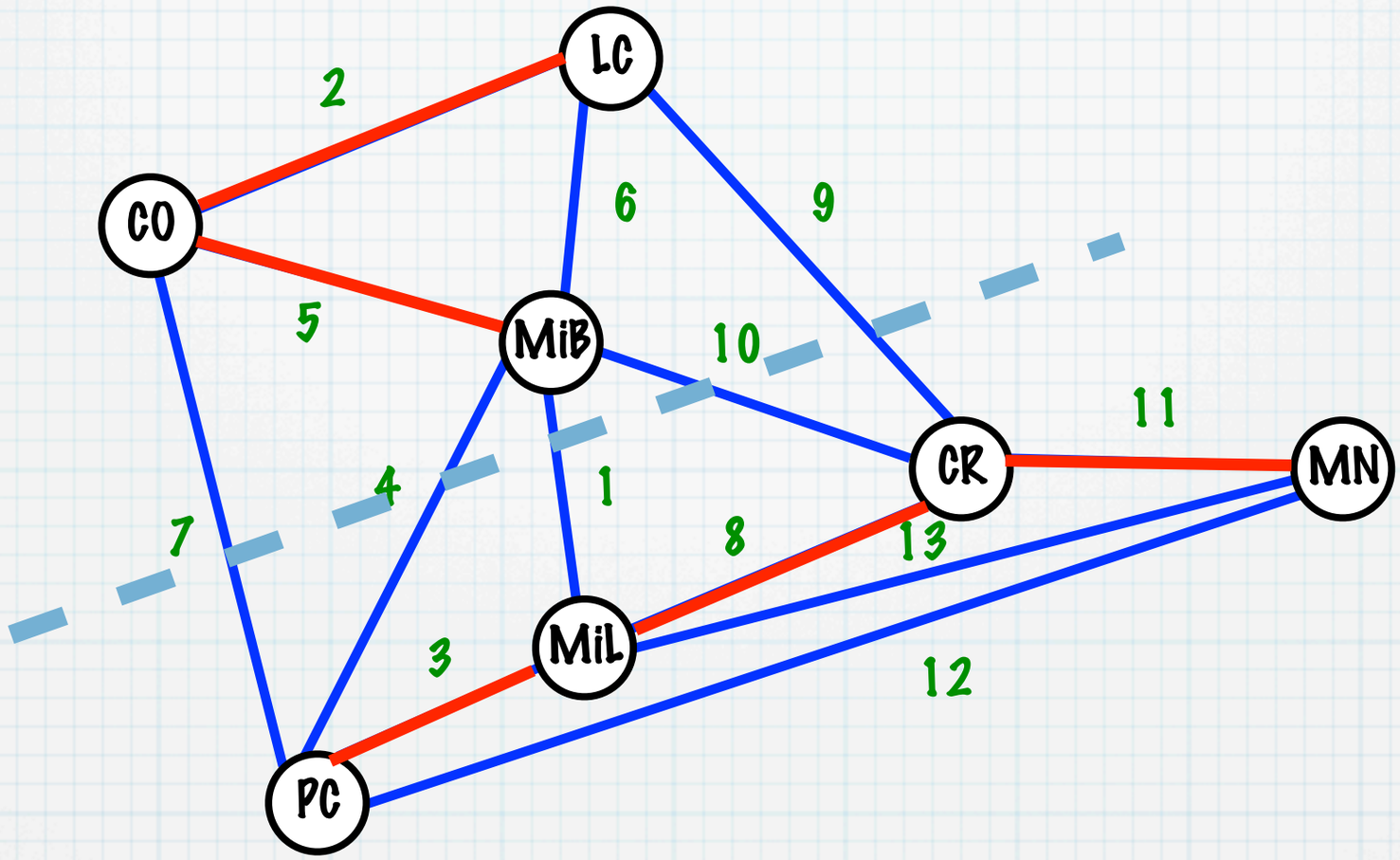


Consider $A' \setminus \{i, j\} : \{i, j\} \in A'$

the removed arc induces two connected components

(H, H')

Property suggesting a solution algorithm

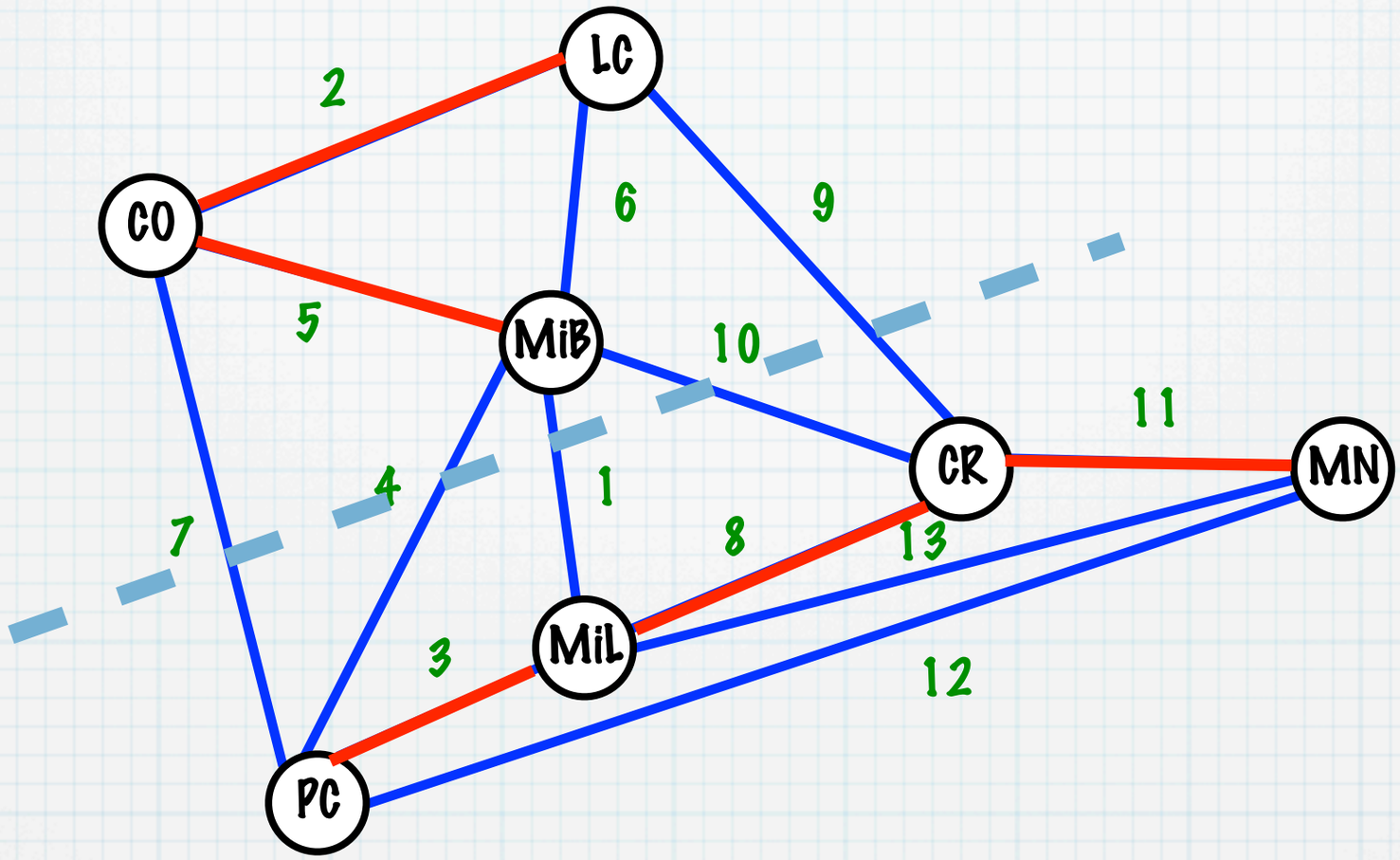


Consider $A' \setminus \{i, j\} : \{i, j\} \in A'$

the removed arc induces two connected components

(H, H')

Property suggesting a solution algorithm



Consider $A' \setminus \{i,j\} : \{i,j\} \in A'$

the removed arc induces two connected components

(H, H')

then $w_{i,j} \leq w_{hk} \quad \forall \{h,k\} : h \in H, k \in H'$

Algorithms

Other algorithms can be derived from this property

...

Algorithms

Other algorithms can be derived from this property

...

If you want to learn more

Algorithms

Other algorithms can be derived from this property

...

If you want to learn more

Design and Analysis of Algorithms

second semester

Design a reliable network

* What happens to the Spanning Tree if a link breaks down?

Design a reliable network

* What happens to the Spanning Tree if a link breaks down?
* How can we make the network reliable in case of a single failure?

Design a reliable network

* What happens to the Spanning Tree if a link breaks down?
* How can we make the network reliable in case of a single failure?

* The cheapest solution is the minimum cost Hamiltonian cycle

Curiosity: Hamiltonian cycle

In 1800s lord Hamilton studied the problem arising from the "Icosian game"

Curiosity: Hamiltonian cycle

In 1800s lord Hamilton studied the problem arising from the "Icosian game"

This gave the start to the literature on the Traveling Salesperson Problem (TSP)

Curiosity: Hamiltonian cycle

In 1800s lord Hamilton studied the problem arising from the "Icosian game"

This gave the start to the literature on the Traveling Salesperson Problem (TSP)

www.tsp.gatech.edu

Curiosity: Hamiltonian cycle

In 1800s lord Hamilton studied the problem arising from the "Icosian game"

This gave the start to the literature on the Traveling Salesperson Problem (TSP)

www.tsp.gatech.edu

There are no necessary and sufficient conditions to say if a graph has a Hamiltonian Cycle