Lecture 14

Introduction to Graphs

Source: Introduction to Algorithms, CLRS

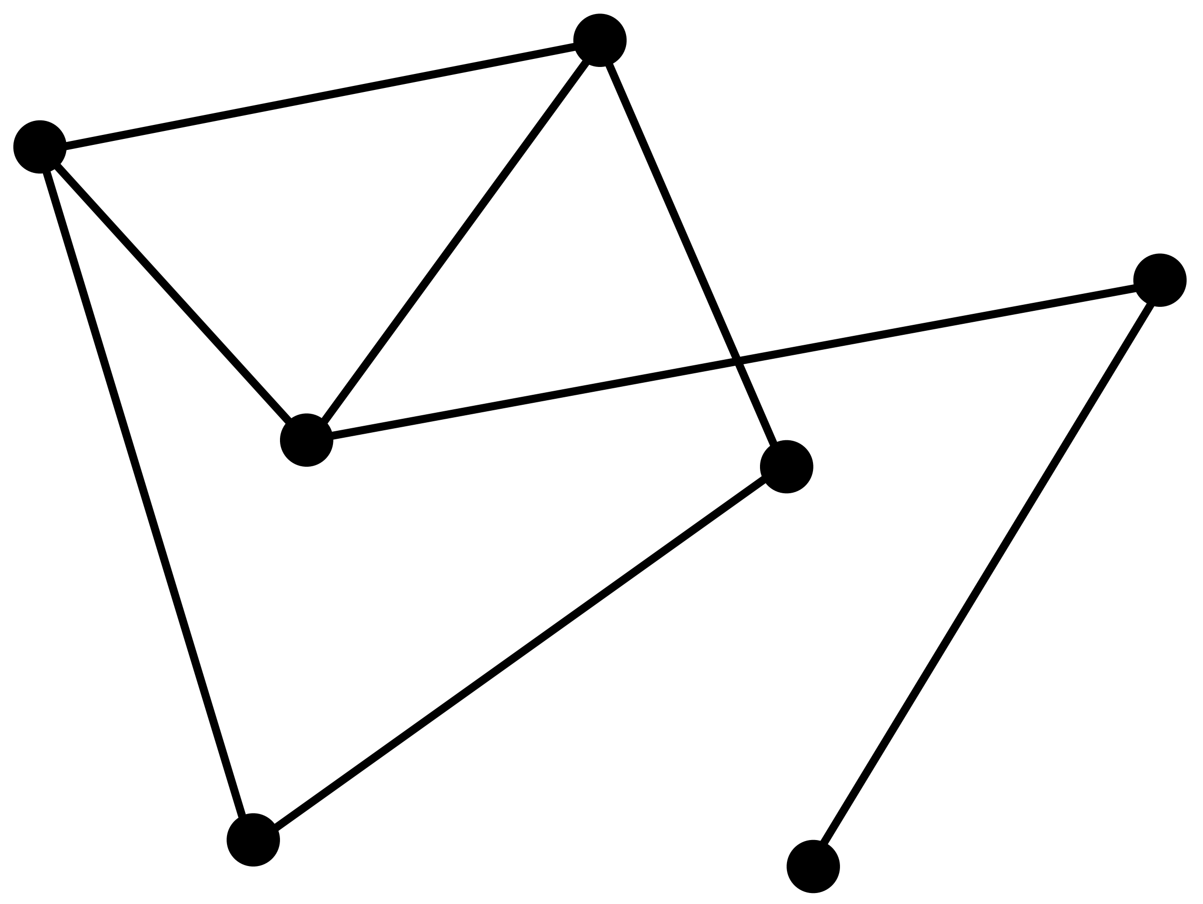
What’s a Graph?

What’s a Graph?

It’s just dots and connections.

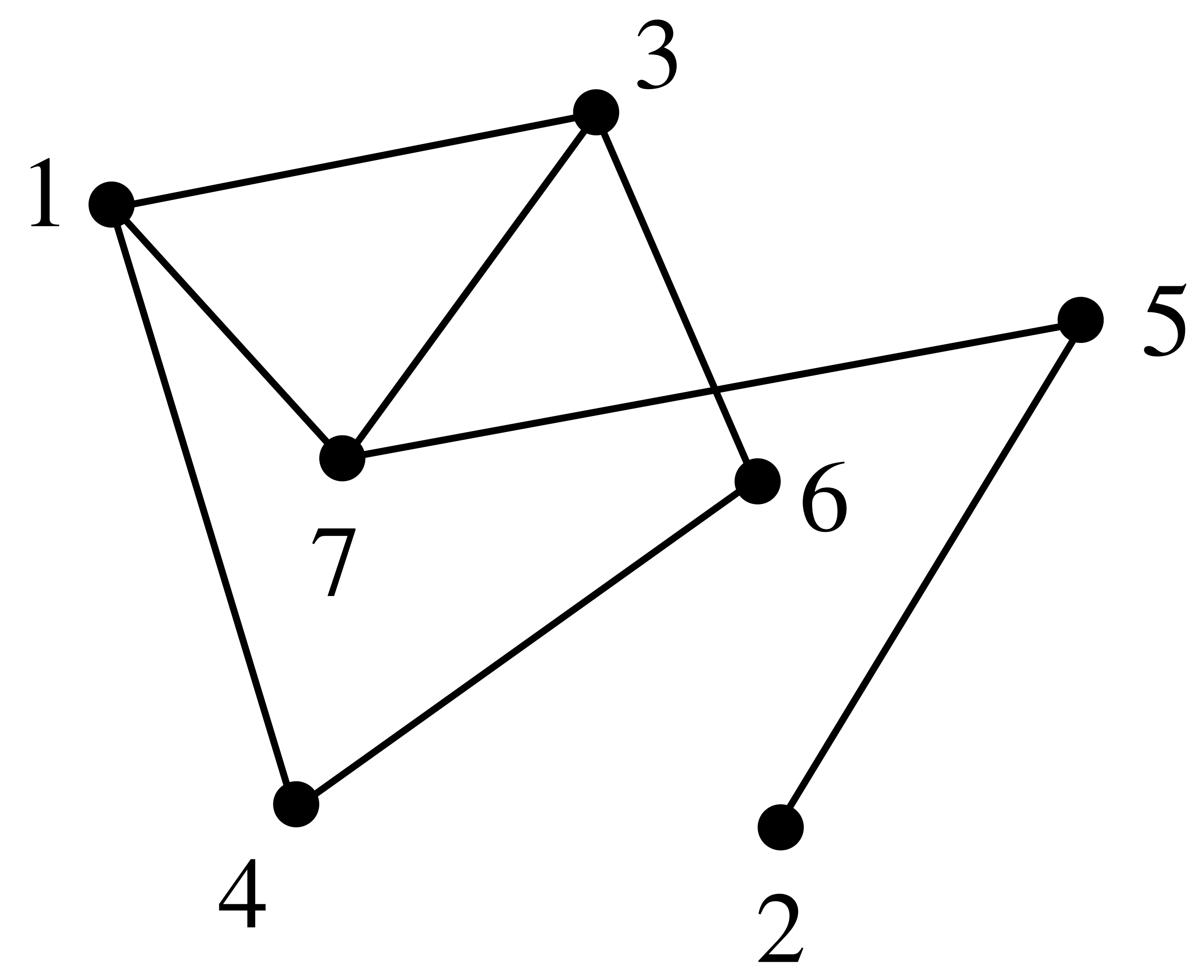
What's a Graph?

It's just dots and connections.



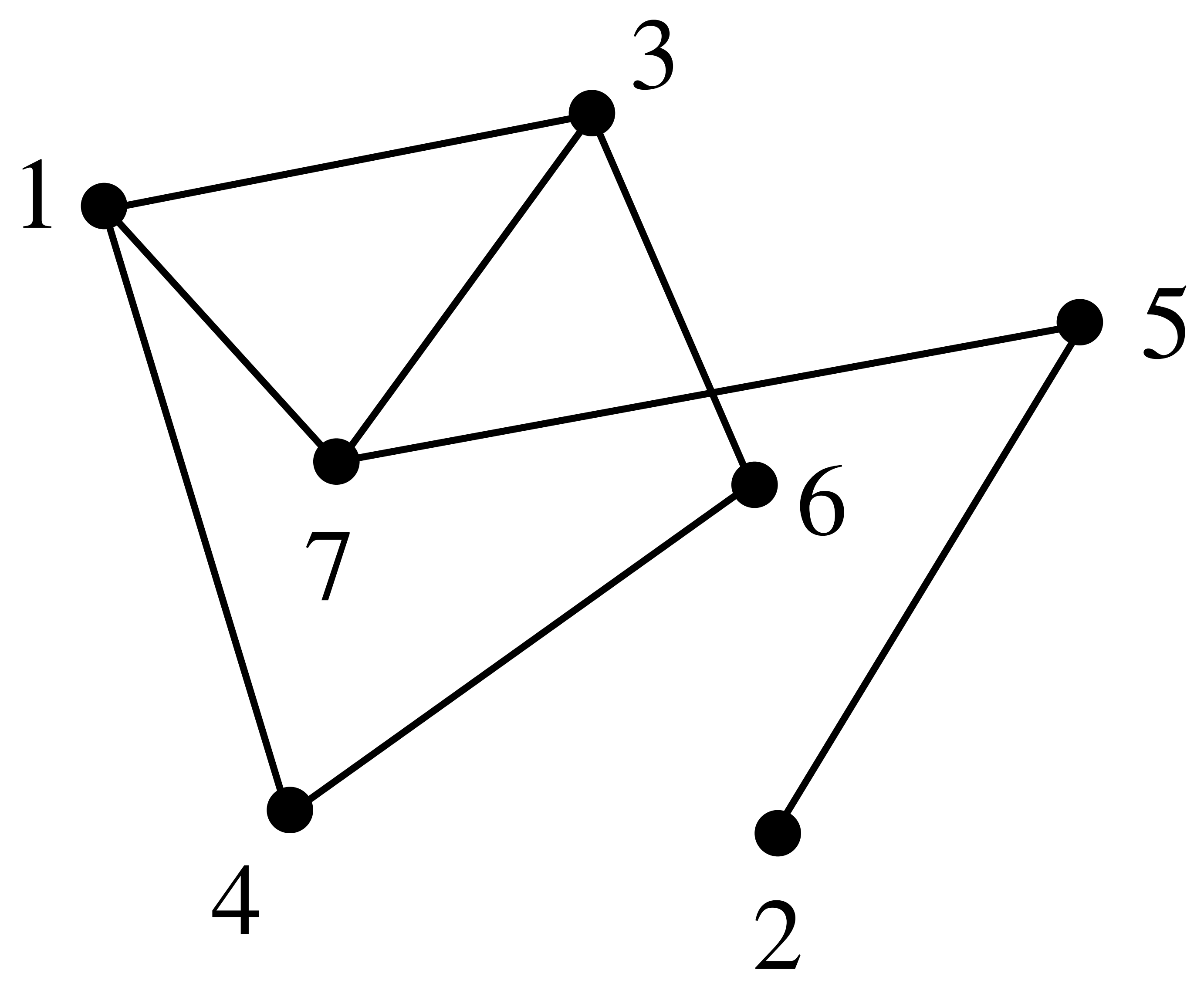
What’s a Graph?

It’s just dots and connections.



What's a Graph?

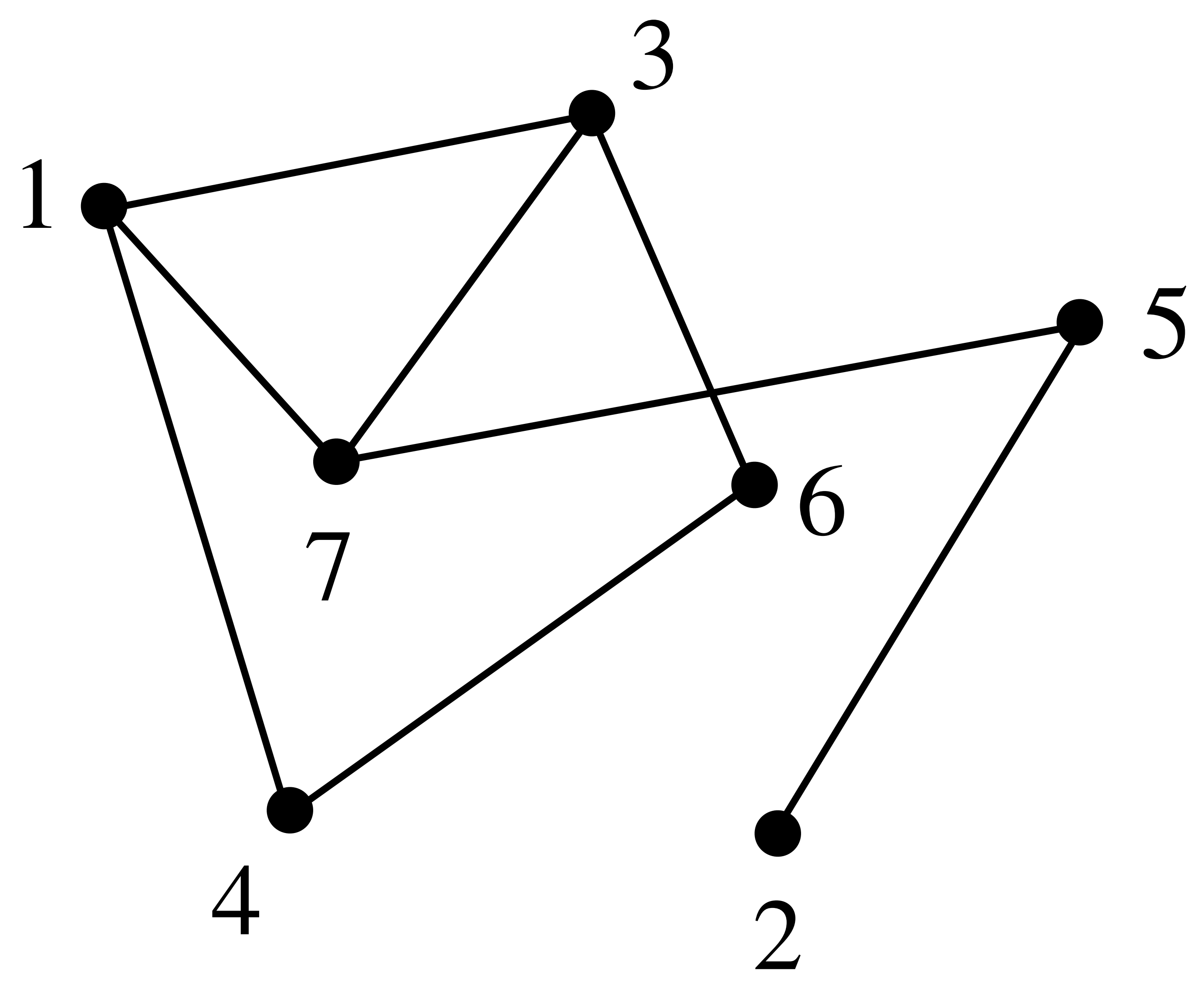
It's just dots and connections.



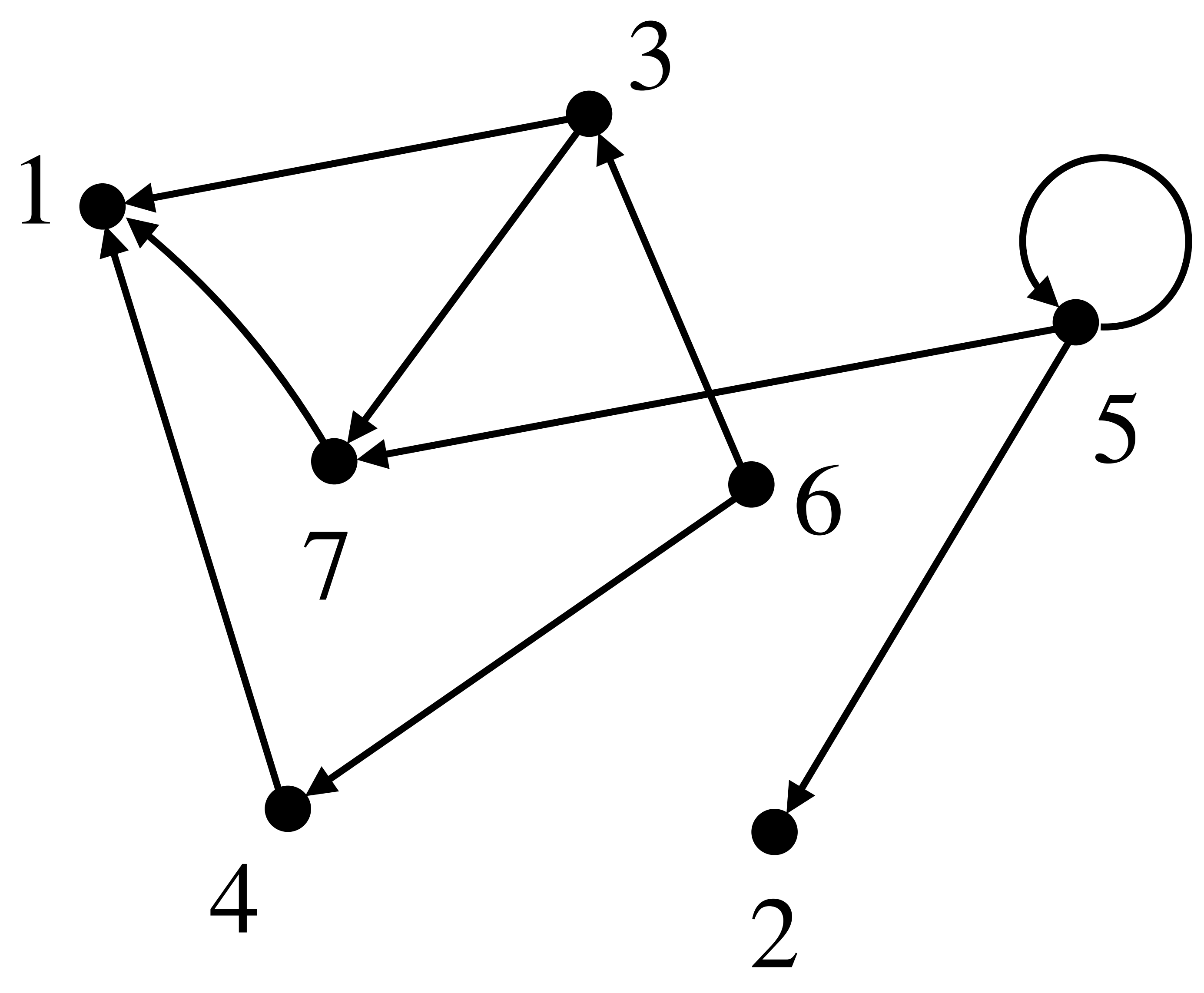
An undirected graph

What's a Graph?

It's just dots and connections.

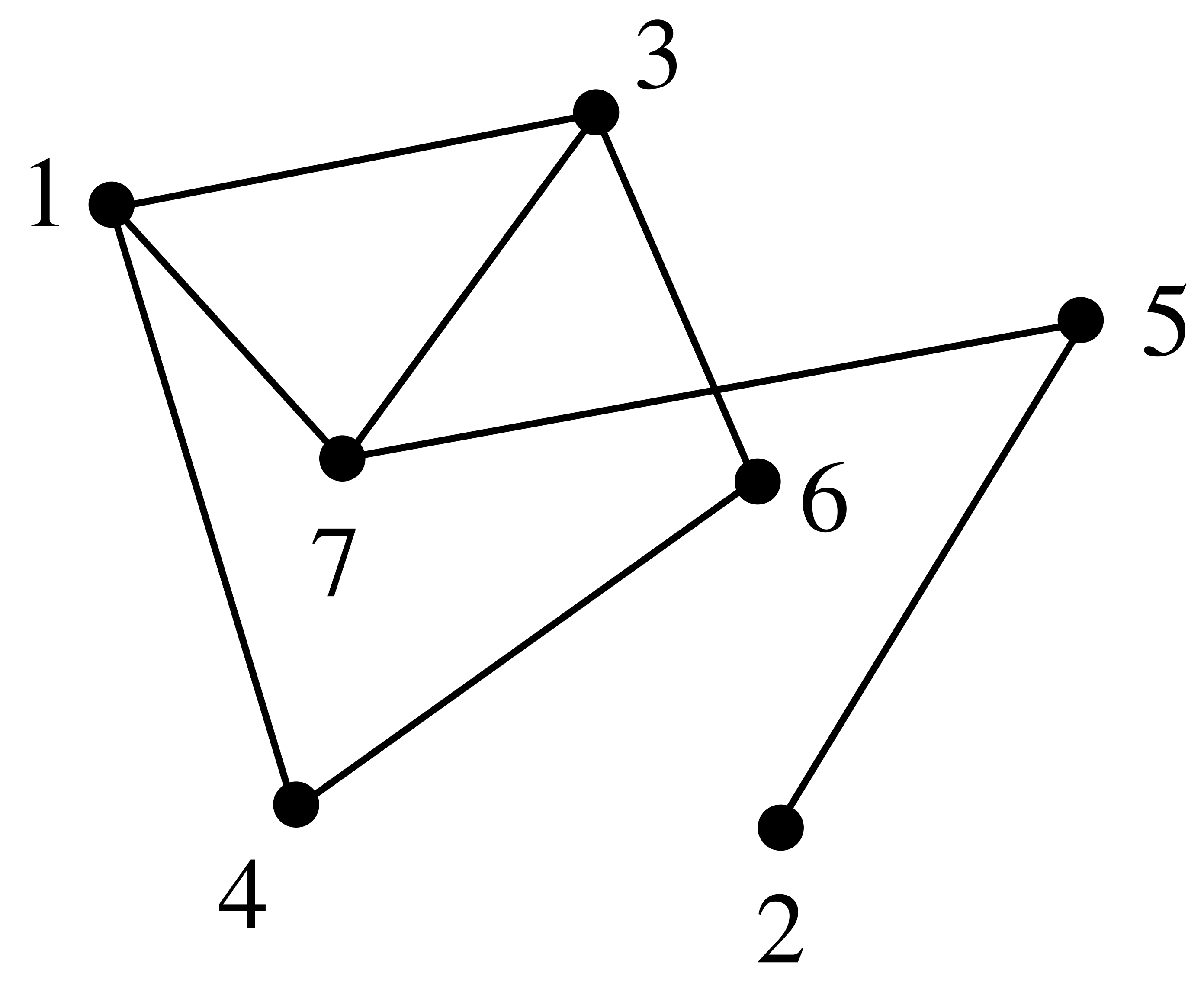


An undirected graph

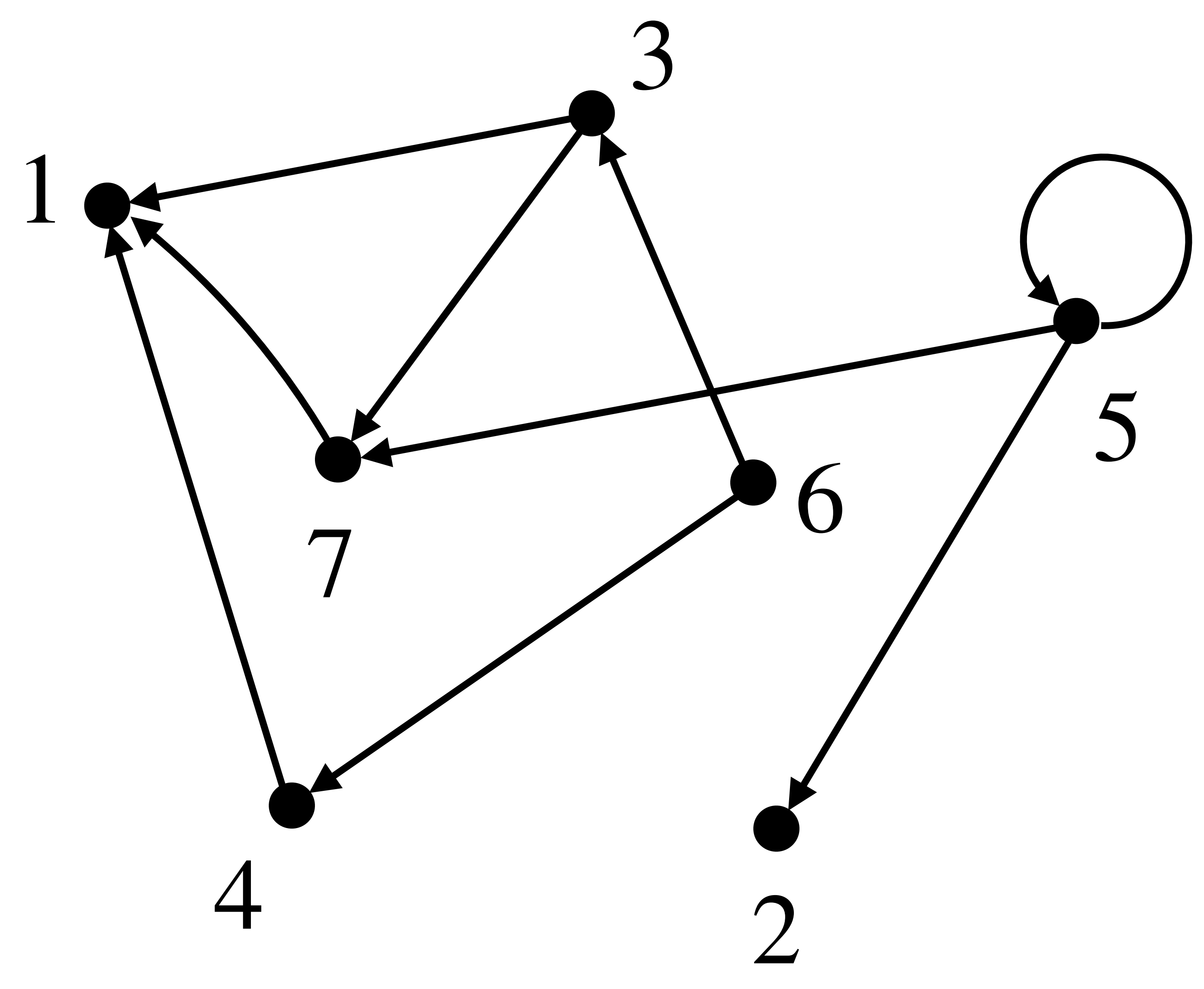


What's a Graph?

It's just dots and connections.

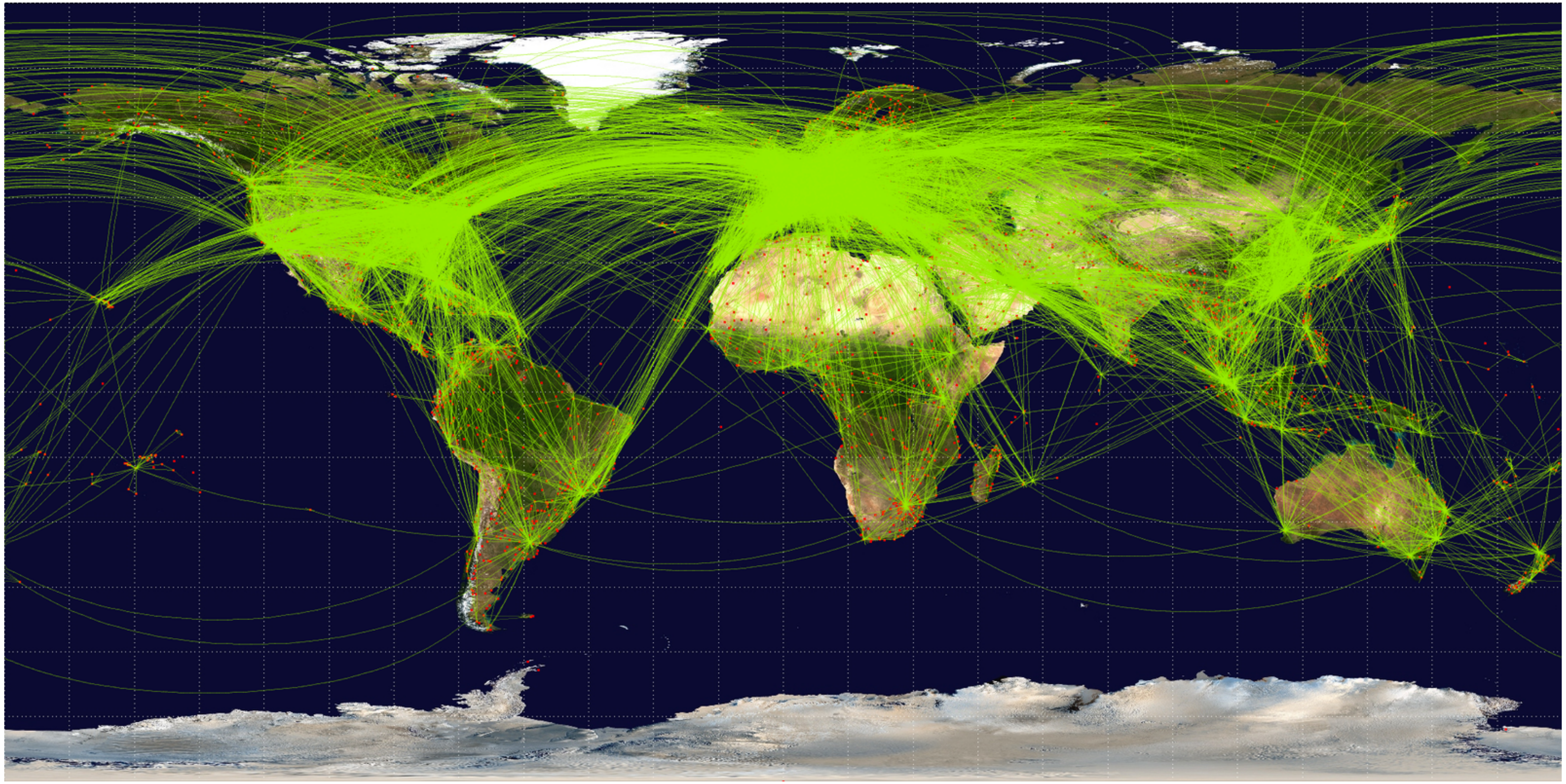


An undirected graph



A directed graph

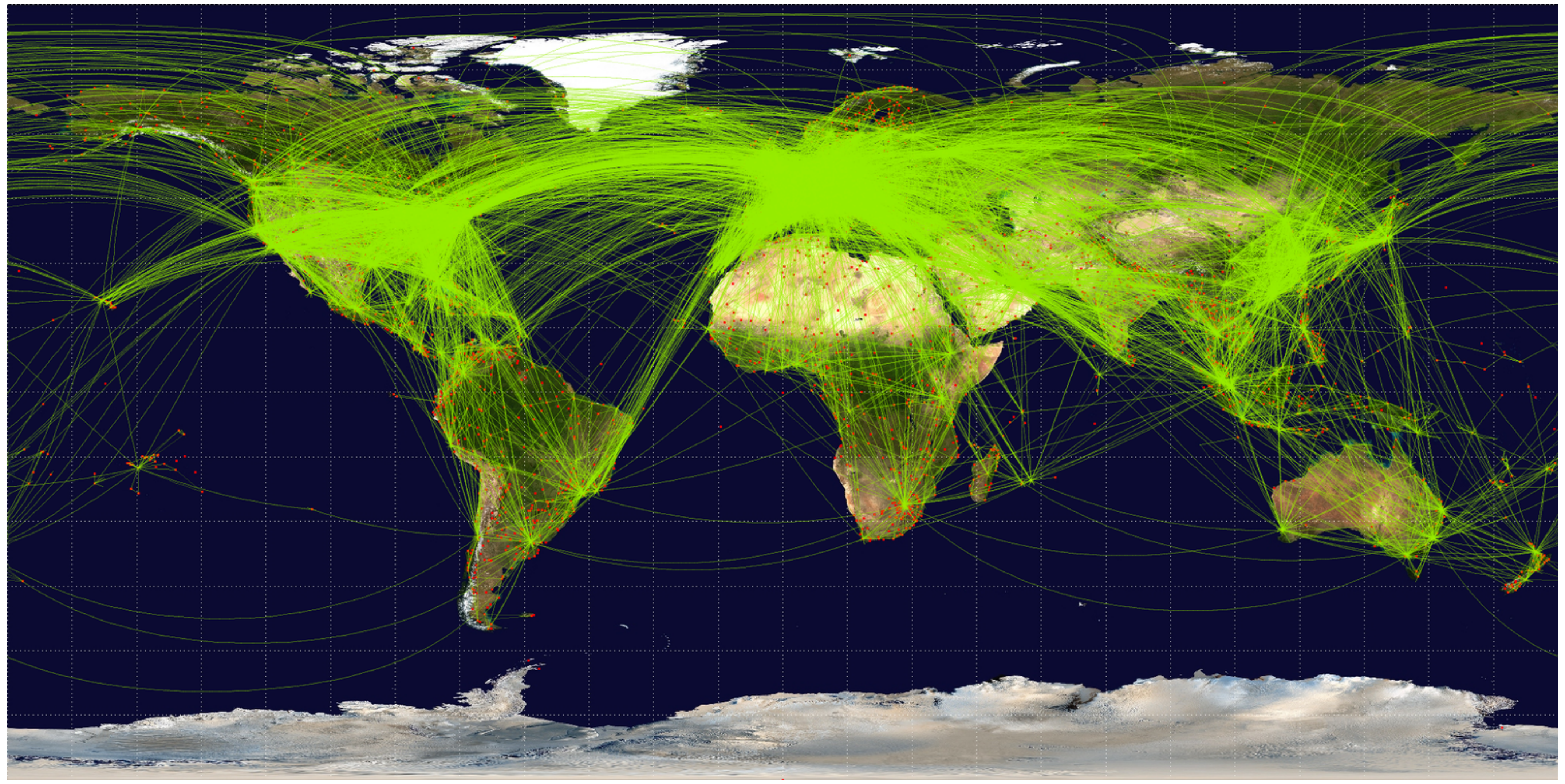
Graphs in Real Life



World airline map

Source: Wiki

Graphs in Real Life

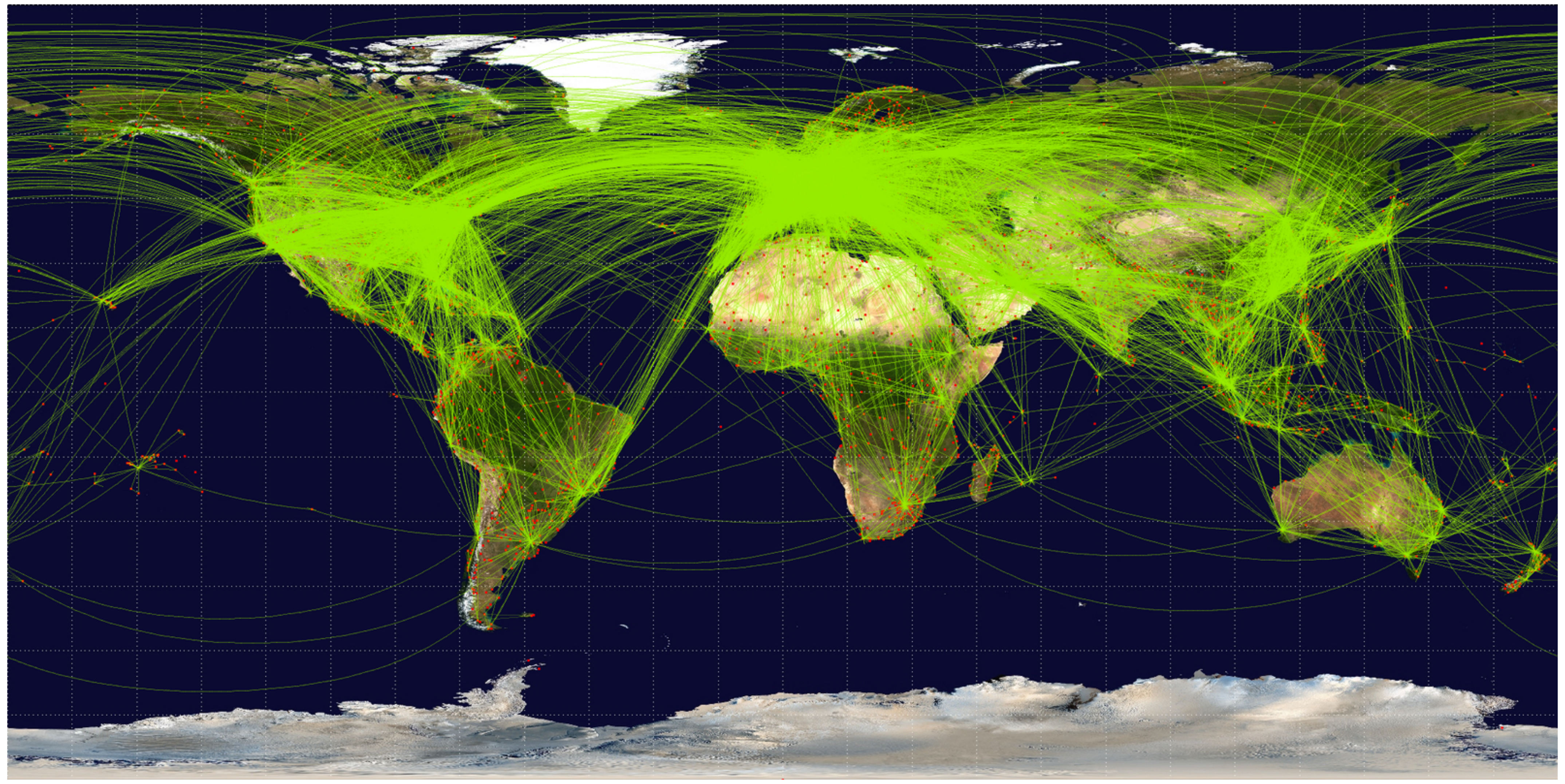


World airline map

Source: Wiki

Graphs in Real Life

Possible queries:



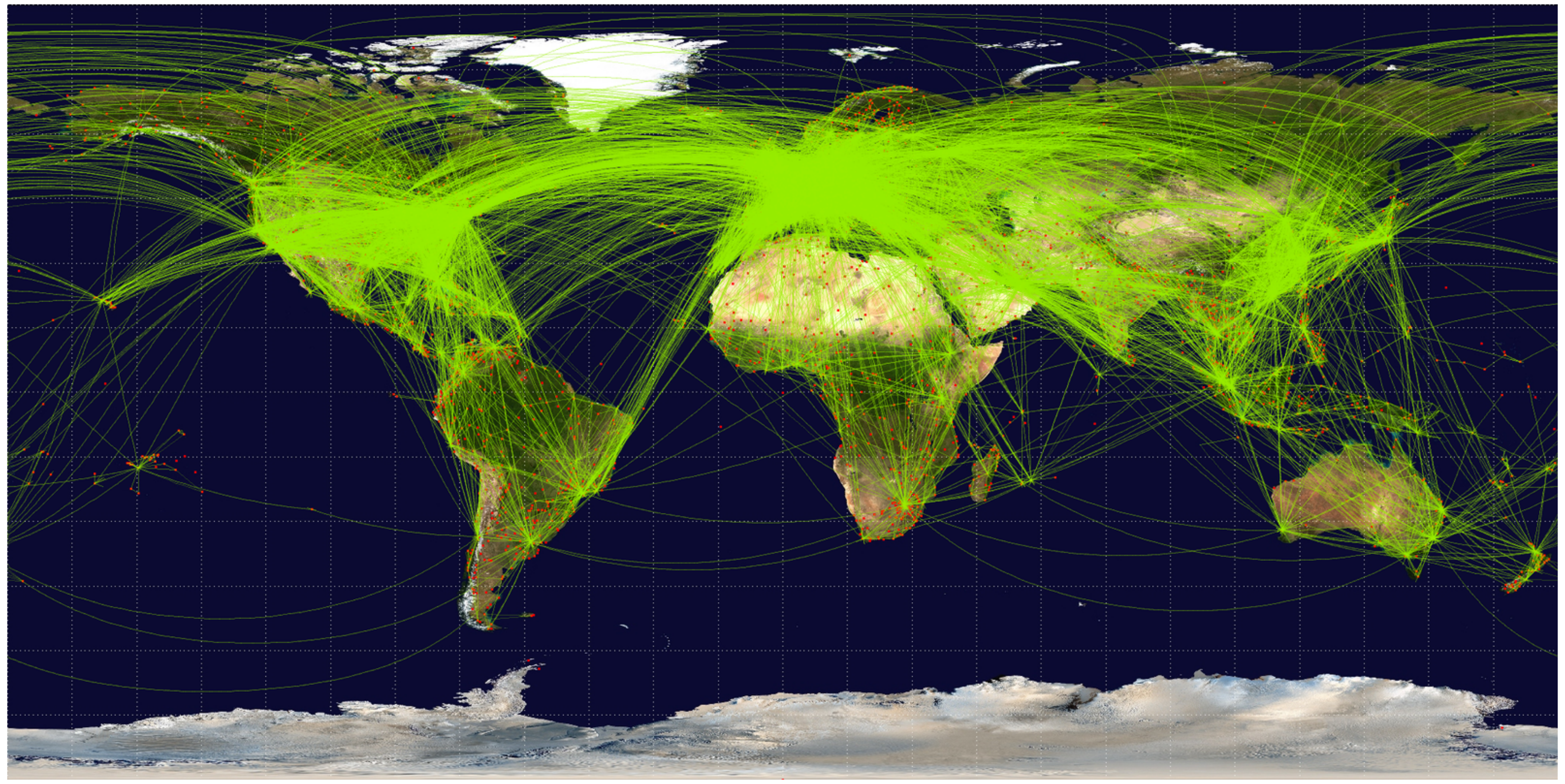
World airline map

Source: Wiki

Graphs in Real Life

Possible queries:

- What's the shortest route from **Delhi** to **Berlin**?



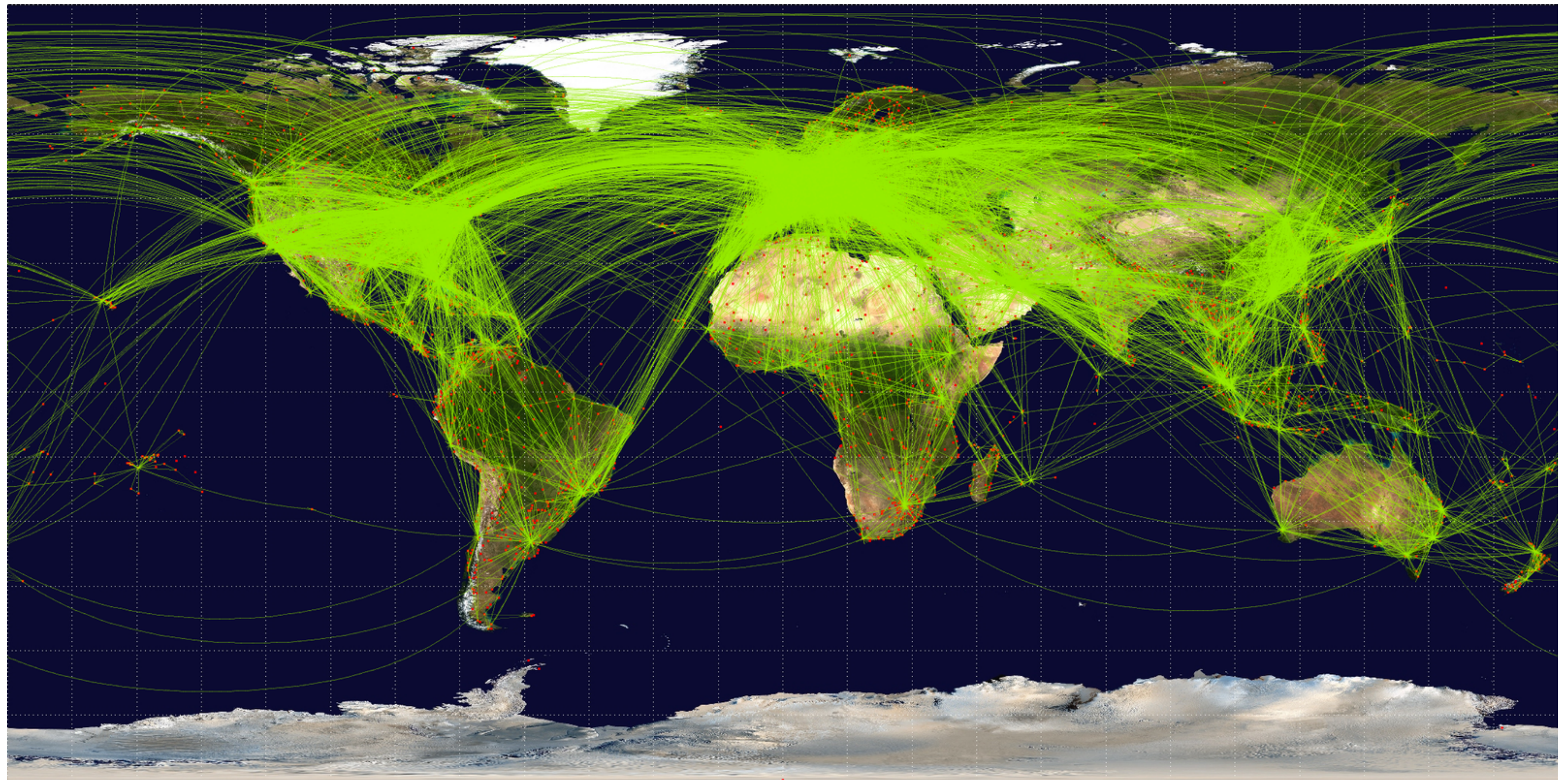
World airline map

Source: Wiki

Graphs in Real Life

Possible queries:

- What's the shortest route from **Delhi** to **Berlin**?
- Is there a route from **Lucknow** to **Helsinki**?



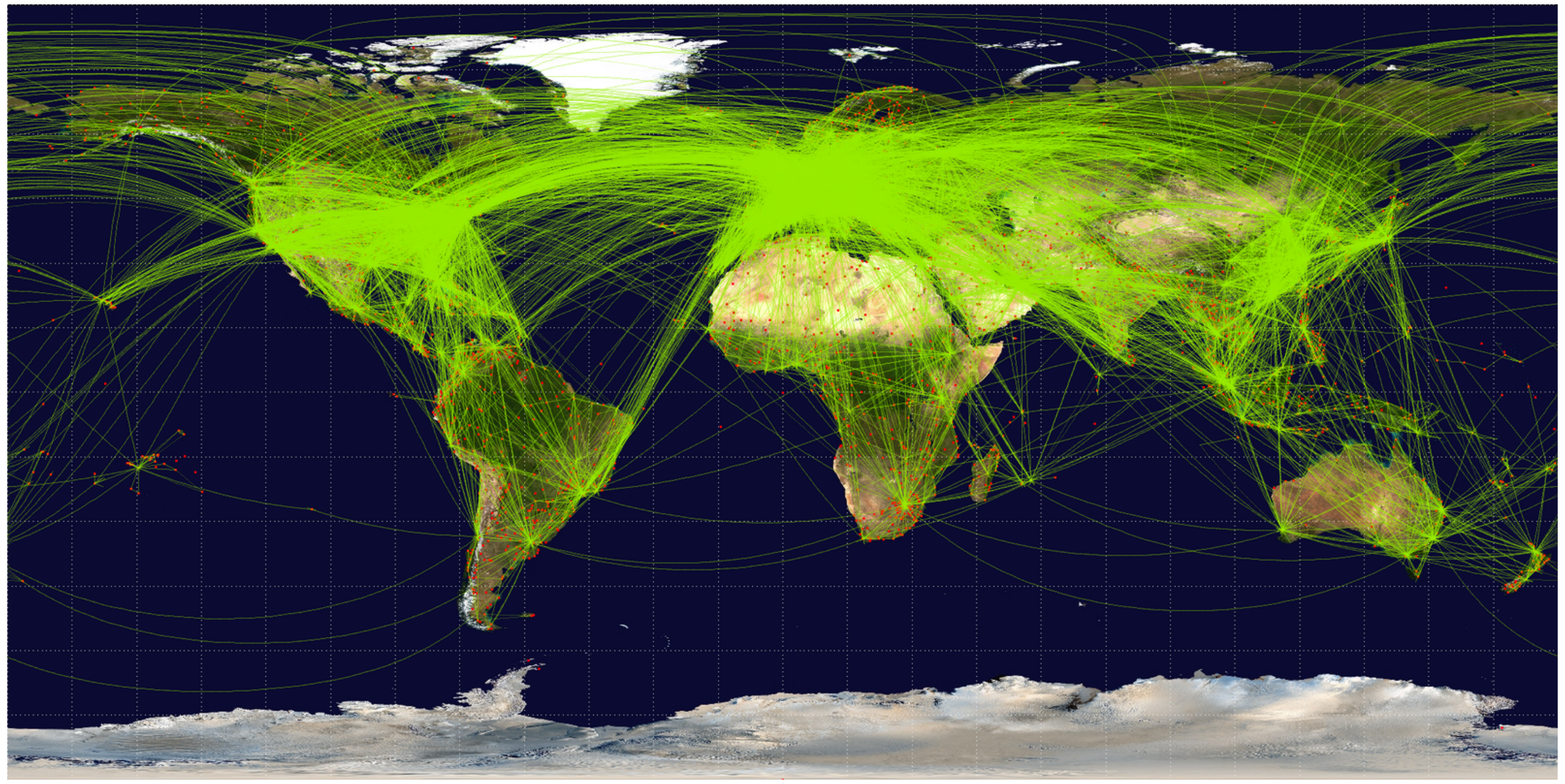
World airline map

Source: Wiki

Graphs in Real Life

Possible queries:

- What's the shortest route from Delhi to Berlin?
- Is there a route from Lucknow to Helsinki?
- How many cities have a direct flight from Mumbai?



World airline map

Source: Wiki

Graph in Real Life

Graph in Real Life

Suppose there are 5 job openings and 6 applicants.

Graph in Real Life

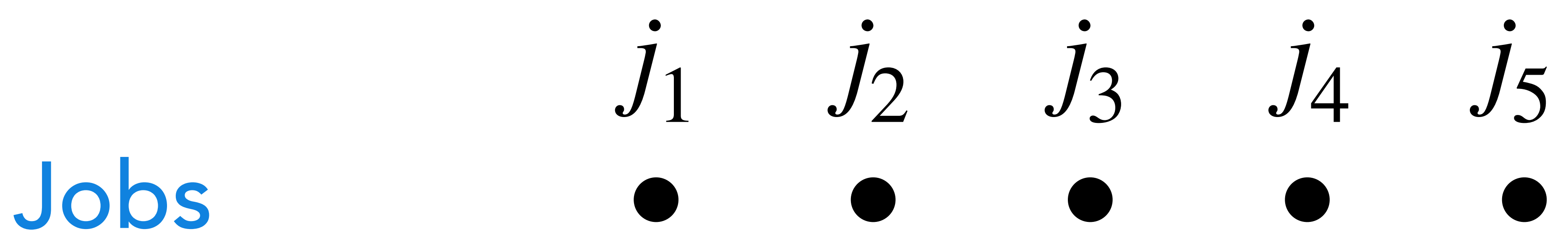
Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by

Graph in Real Life

Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by hiring exactly one applicant and one applicant can do at most one job.

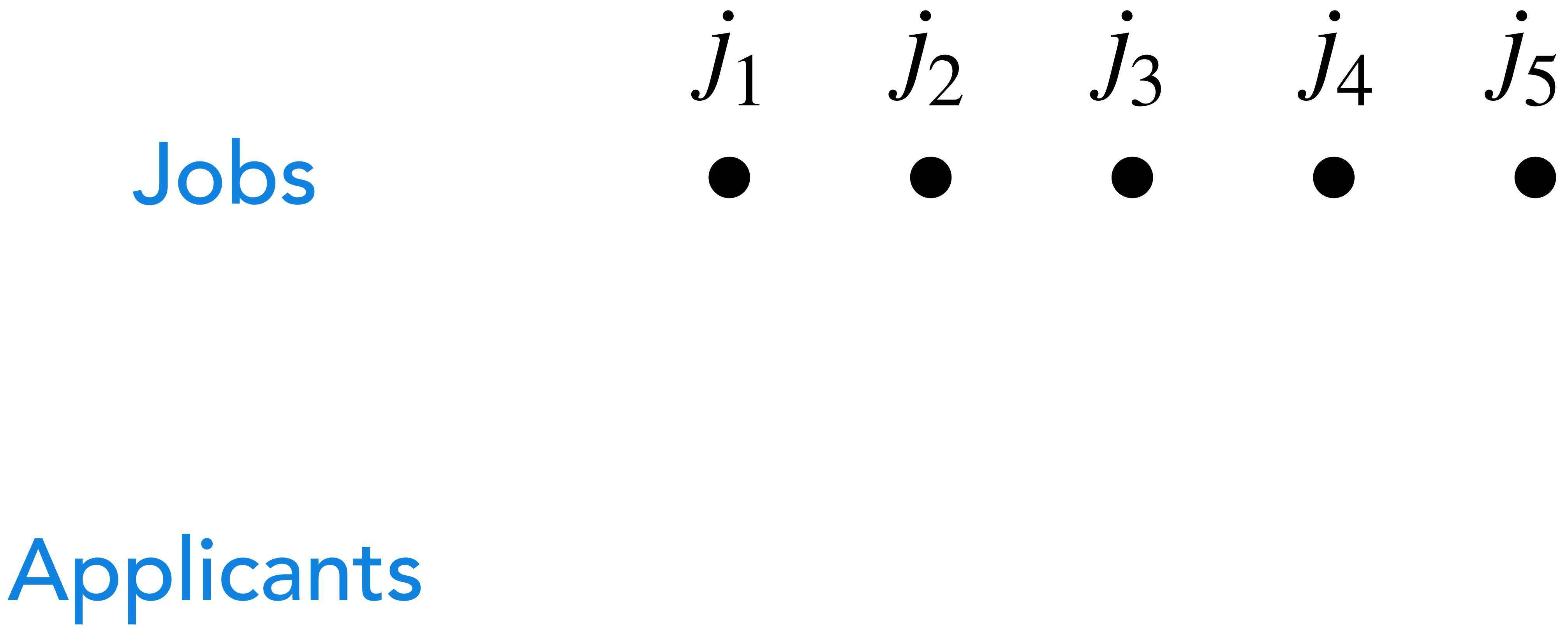
Graph in Real Life

Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by hiring exactly one applicant and one applicant can do at most one job.



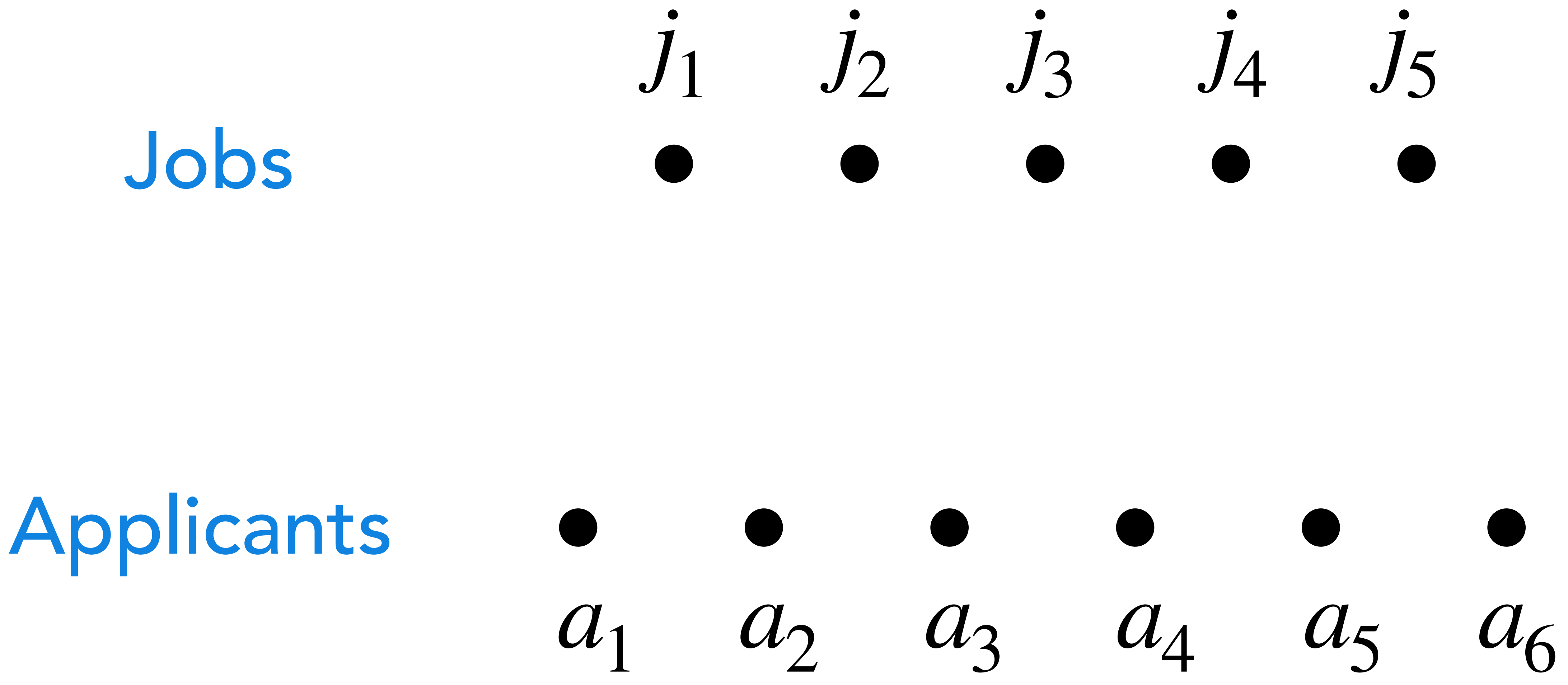
Graph in Real Life

Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by hiring exactly one applicant and one applicant can do at most one job.



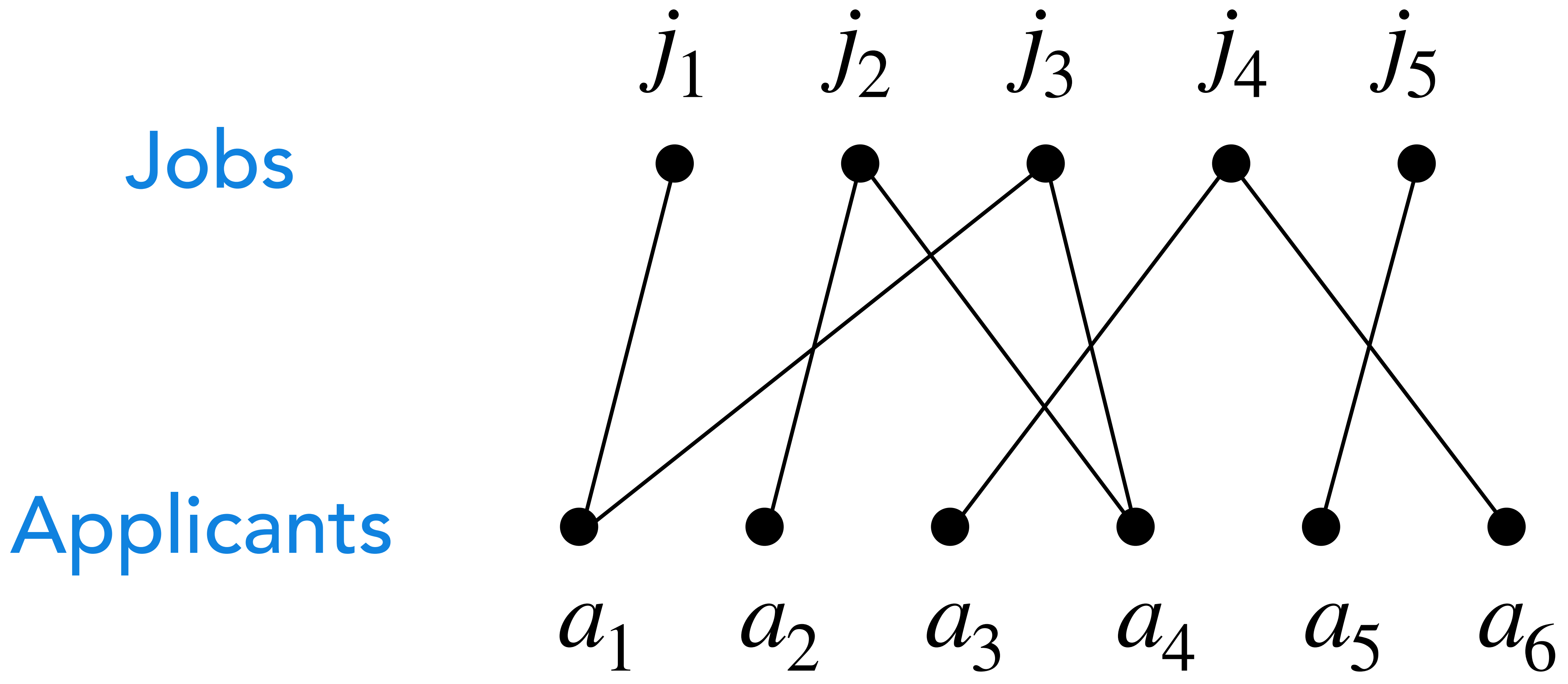
Graph in Real Life

Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by hiring exactly one applicant and one applicant can do at most one job.



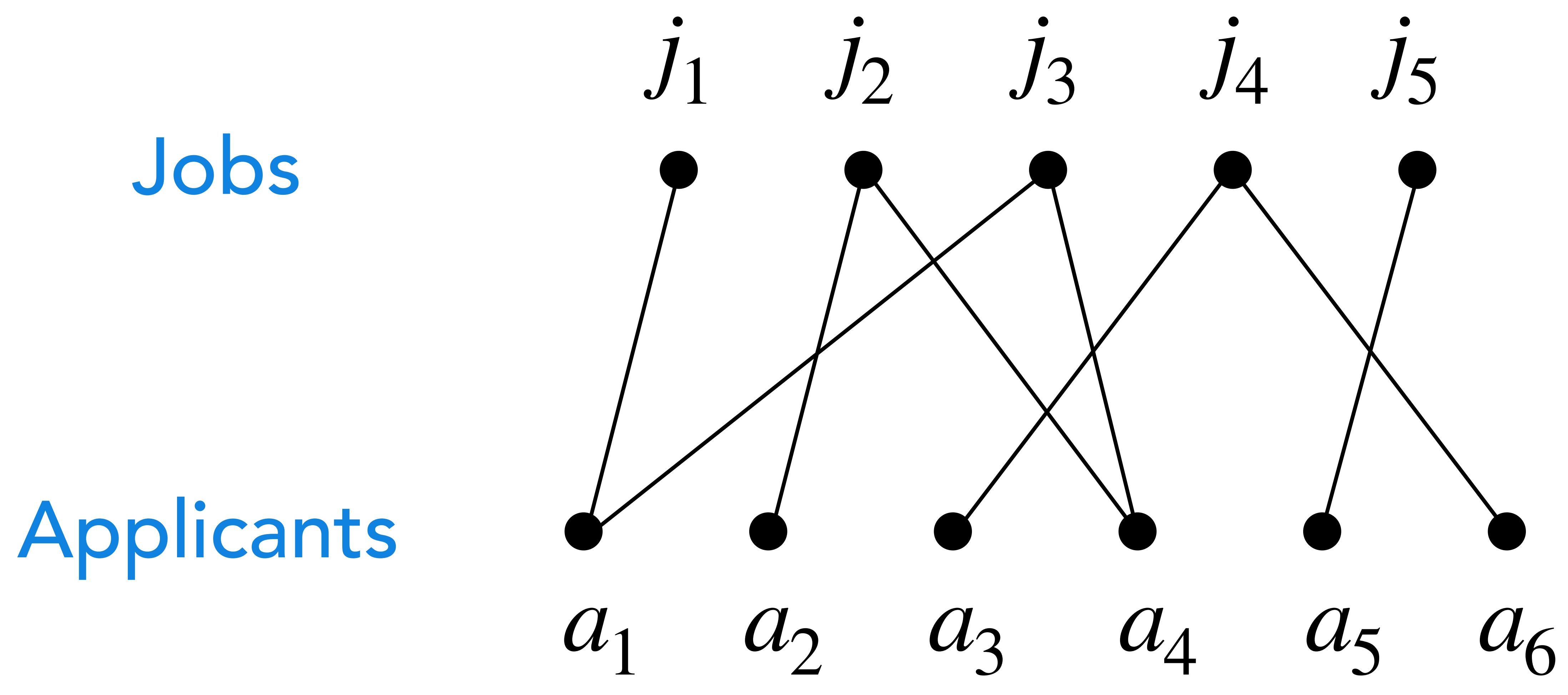
Graph in Real Life

Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by hiring exactly one applicant and one applicant can do at most one job.



Graph in Real Life

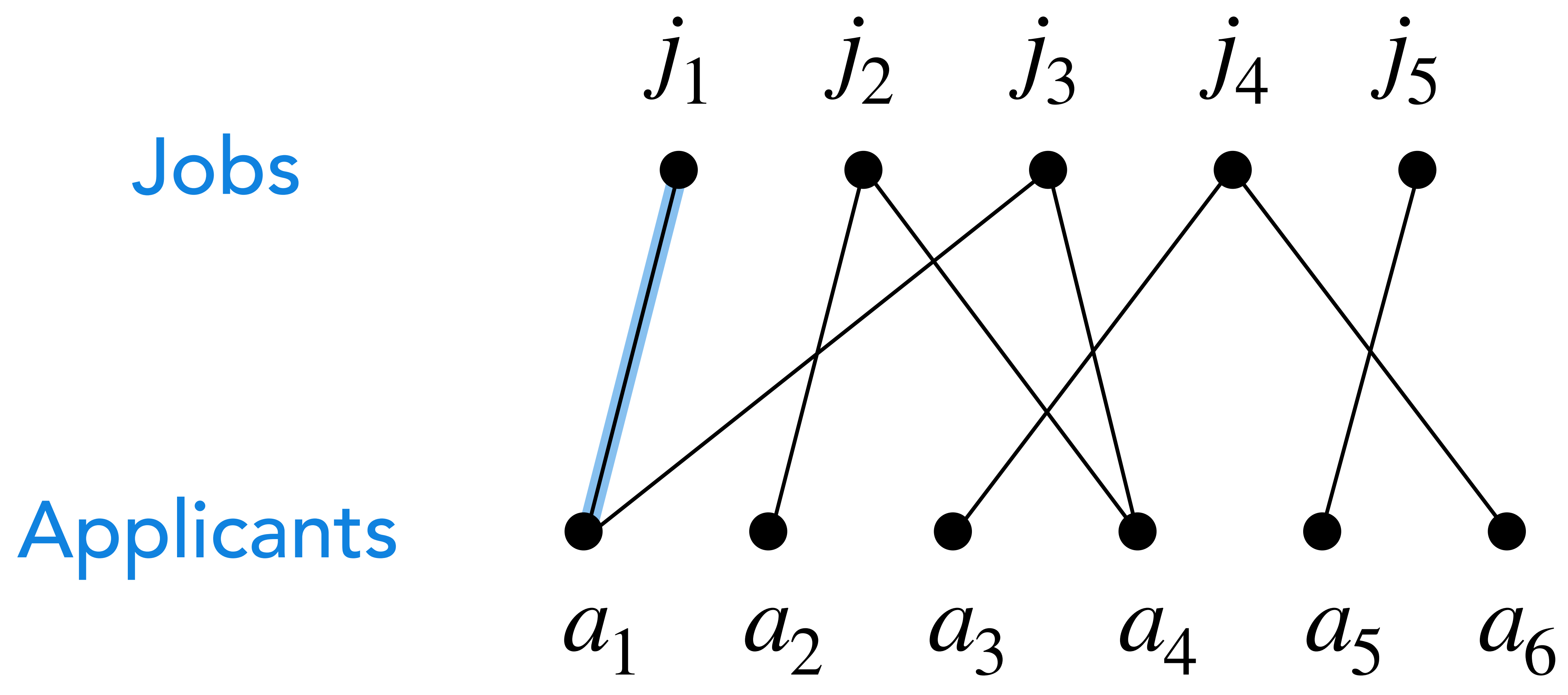
Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by hiring exactly one applicant and one applicant can do at most one job.



Is it possible to fill all the job openings?

Graph in Real Life

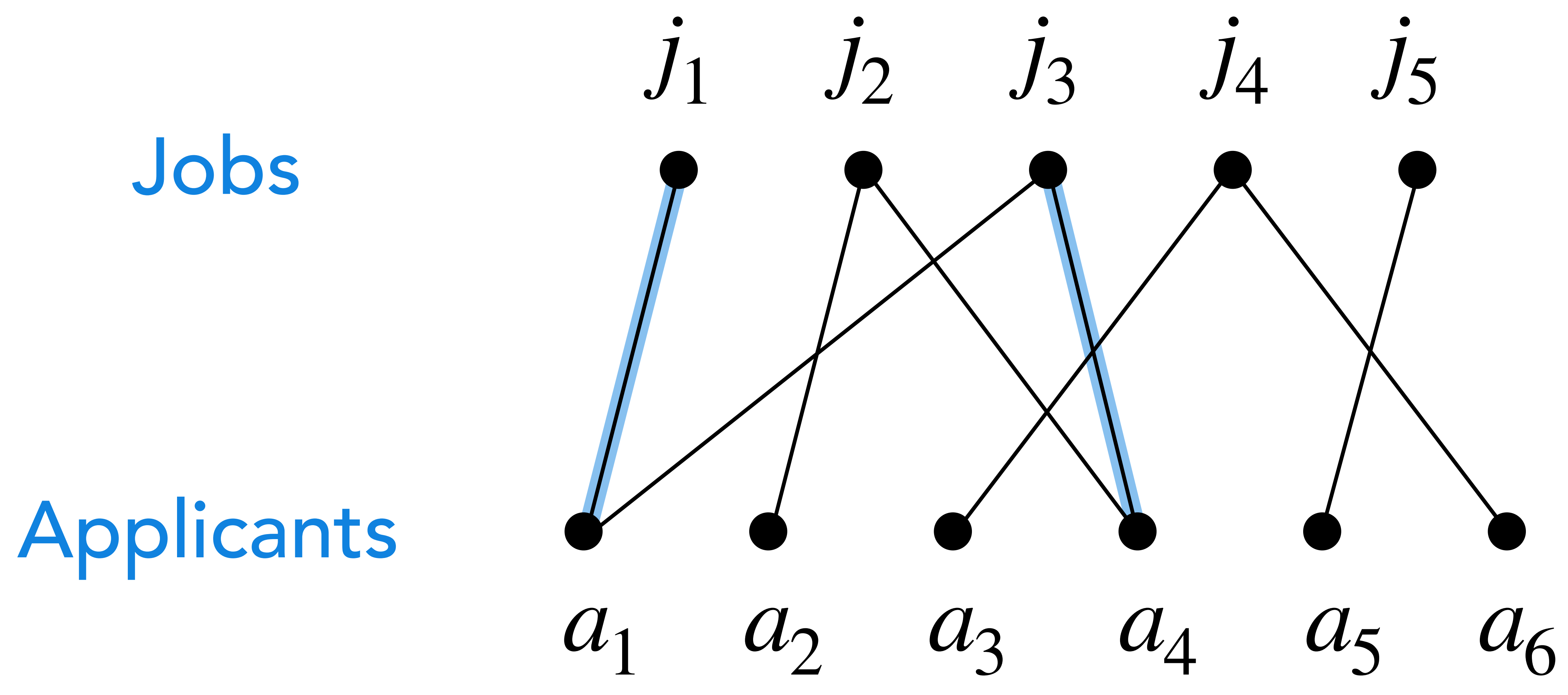
Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by hiring exactly one applicant and one applicant can do at most one job.



Is it possible to fill all the job openings?

Graph in Real Life

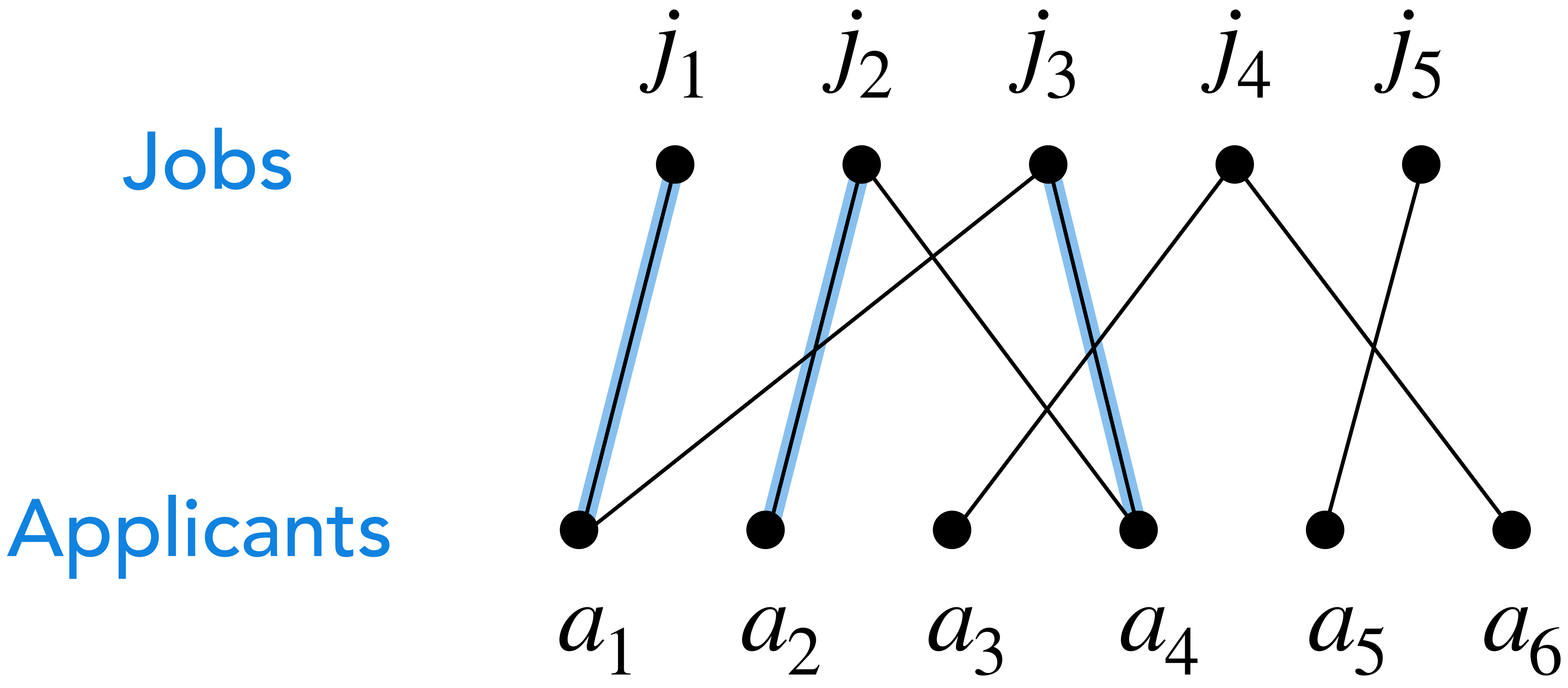
Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by hiring exactly one applicant and one applicant can do at most one job.



Is it possible to fill all the job openings?

Graph in Real Life

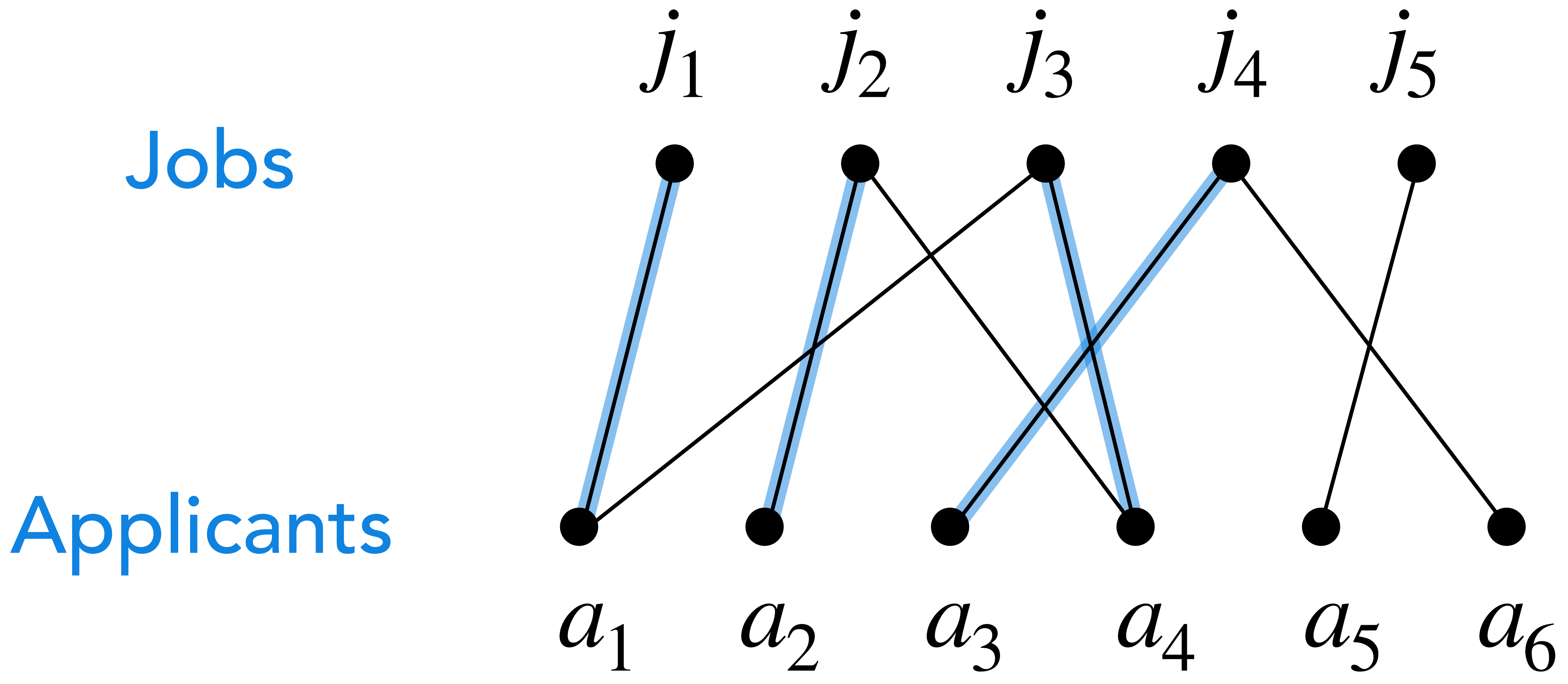
Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by hiring exactly one applicant and one applicant can do at most one job.



Is it possible to fill all the job openings?

Graph in Real Life

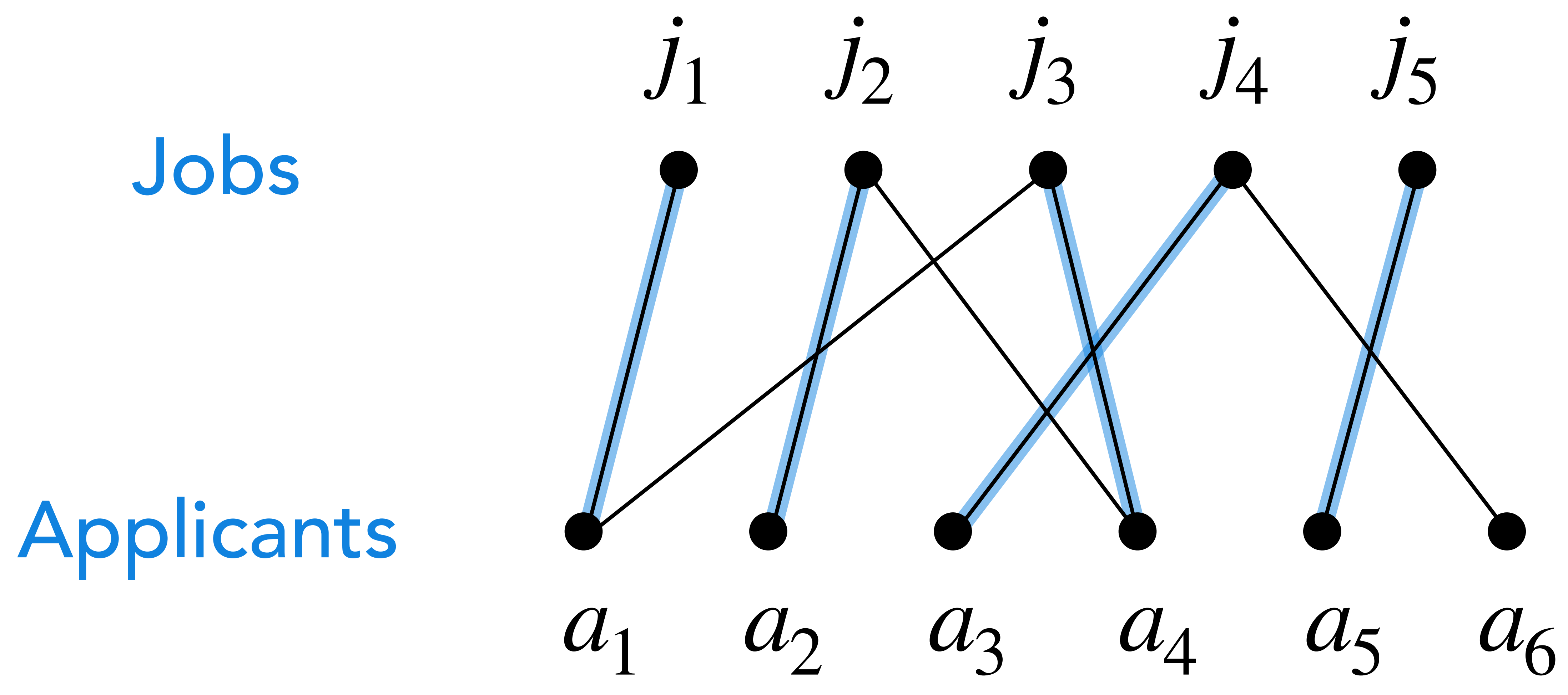
Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by hiring exactly one applicant and one applicant can do at most one job.



Is it possible to fill all the job openings?

Graph in Real Life

Suppose there are 5 job openings and 6 applicants. We want to fill each job opening by hiring exactly one applicant and one applicant can do at most one job.



Is it possible to fill all the job openings?

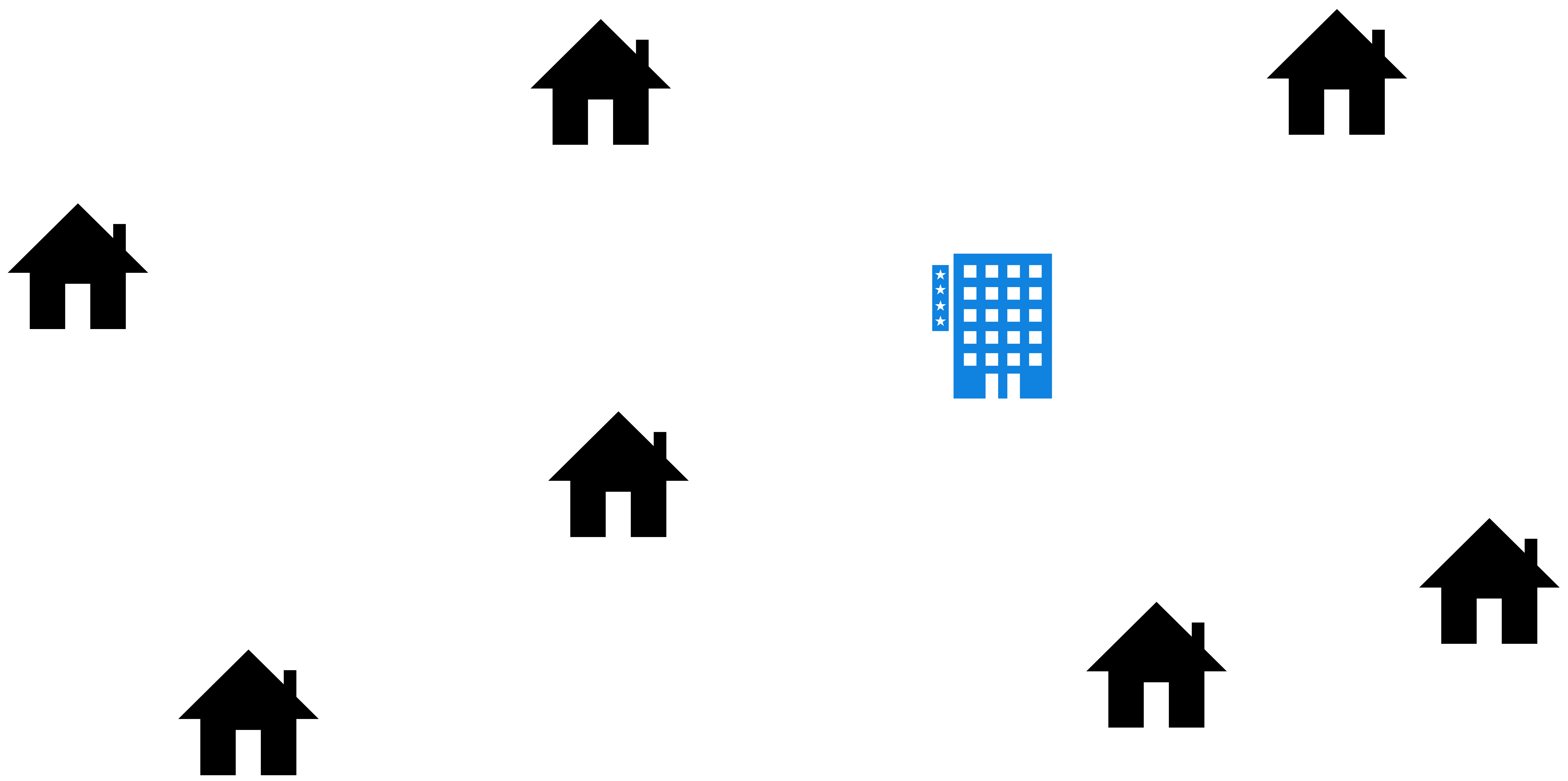
Graph in Real Life

Graph in Real Life

How can we connect houses with network provider with least total wire length?

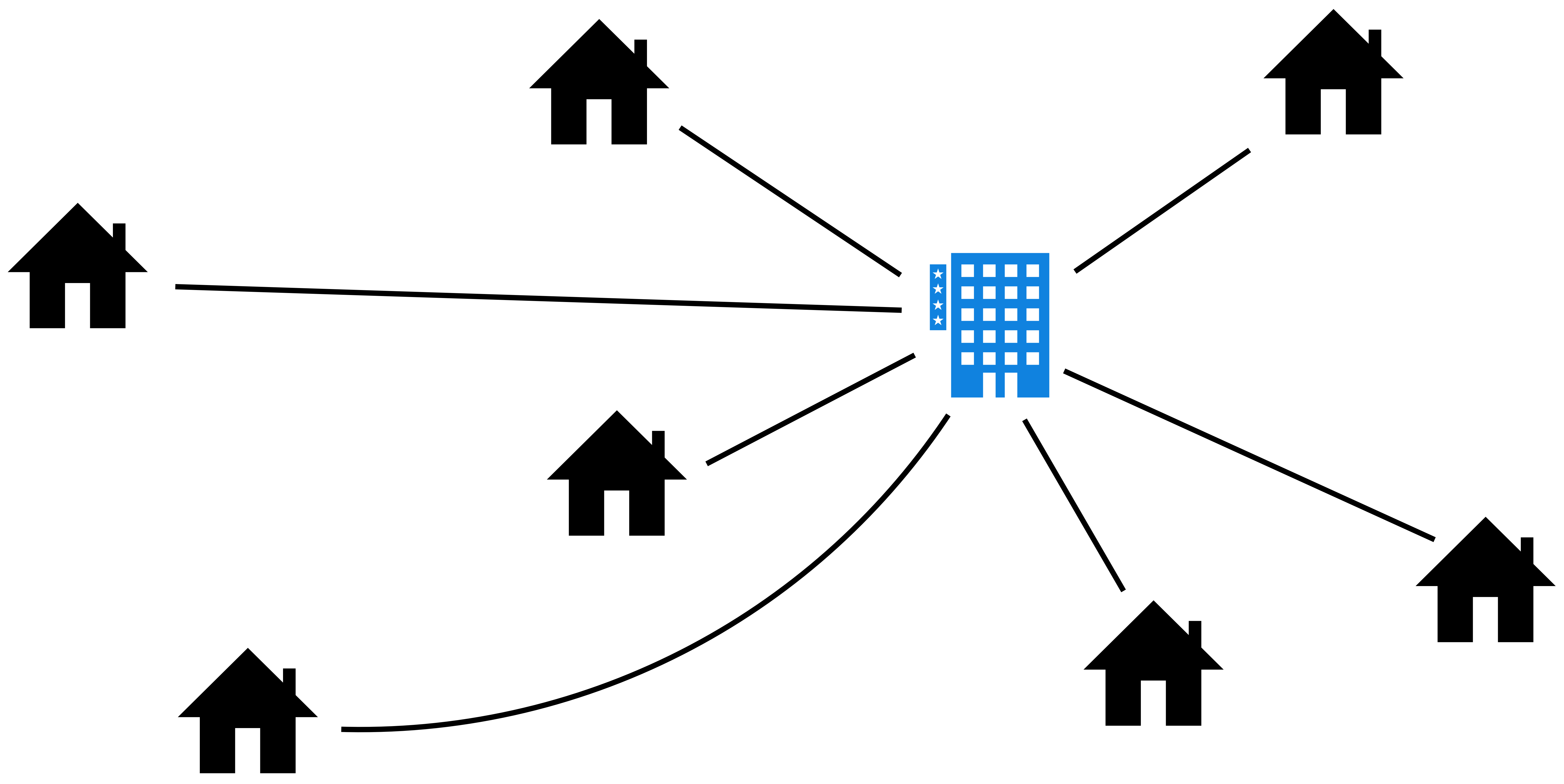
Graph in Real Life

How can we connect houses with network provider with least total wire length?



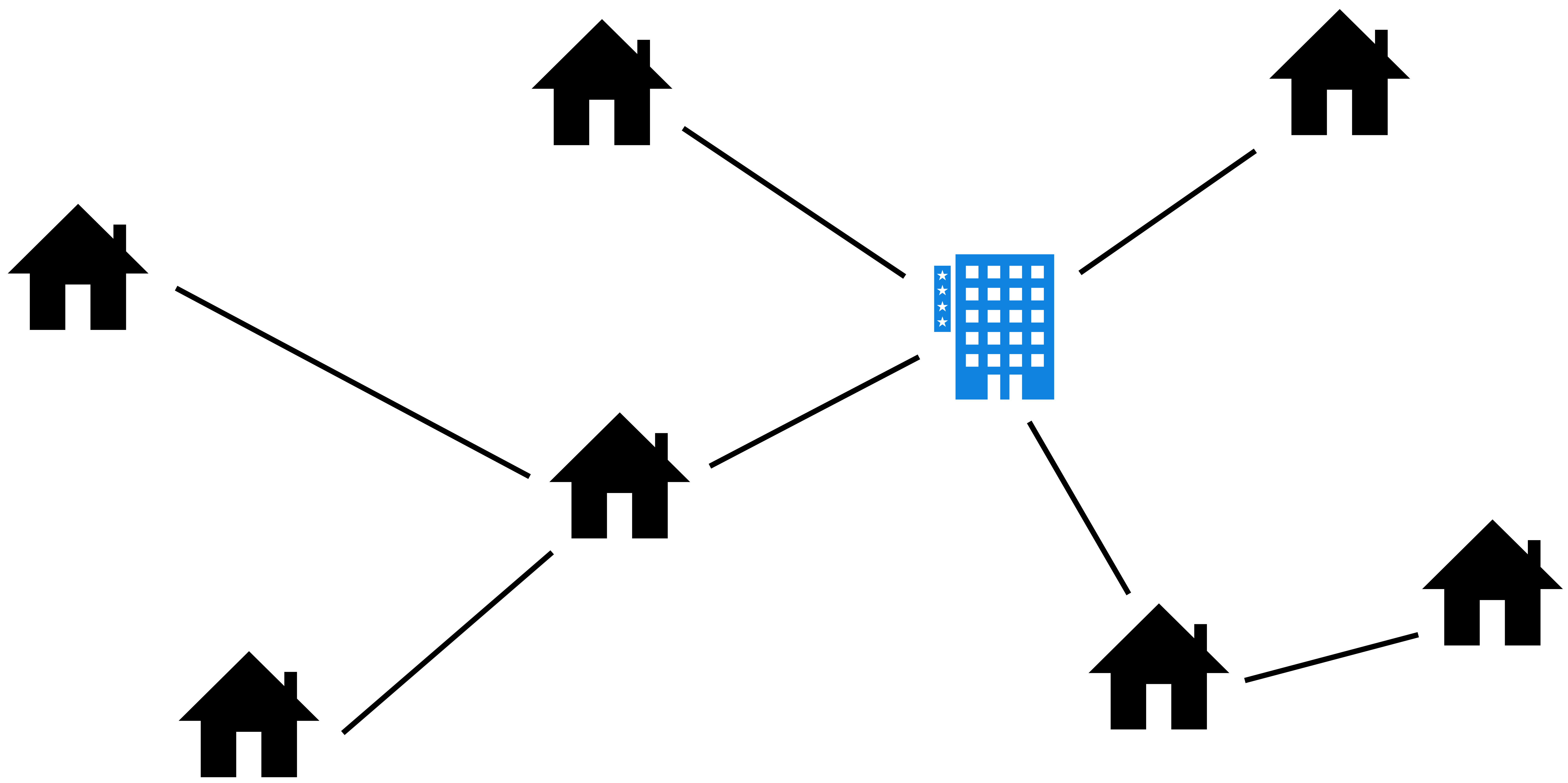
Graph in Real Life

How can we connect houses with network provider with least total wire length?



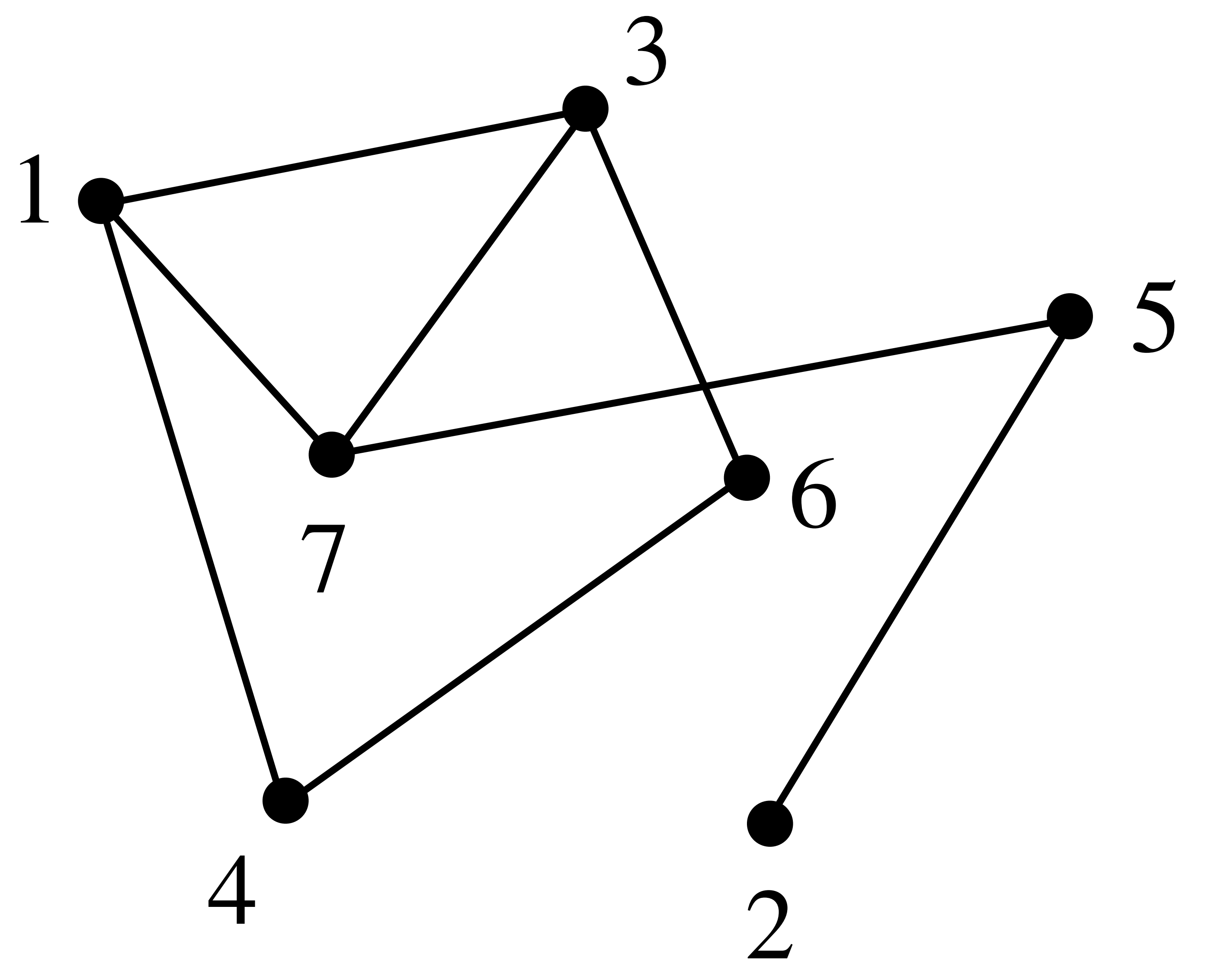
Graph in Real Life

How can we connect houses with network provider with least total wire length?



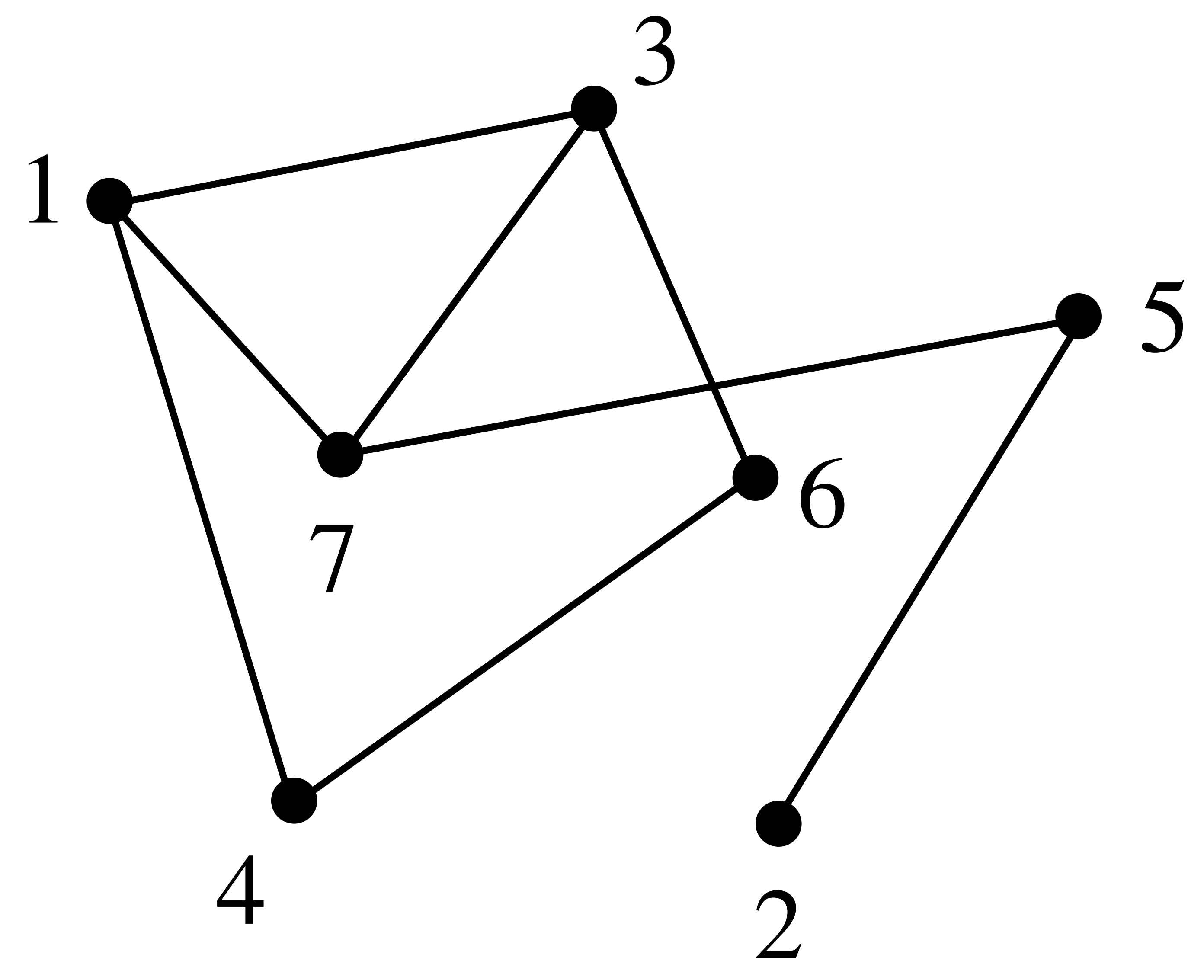
Graphs: Basics

Graphs: Basics



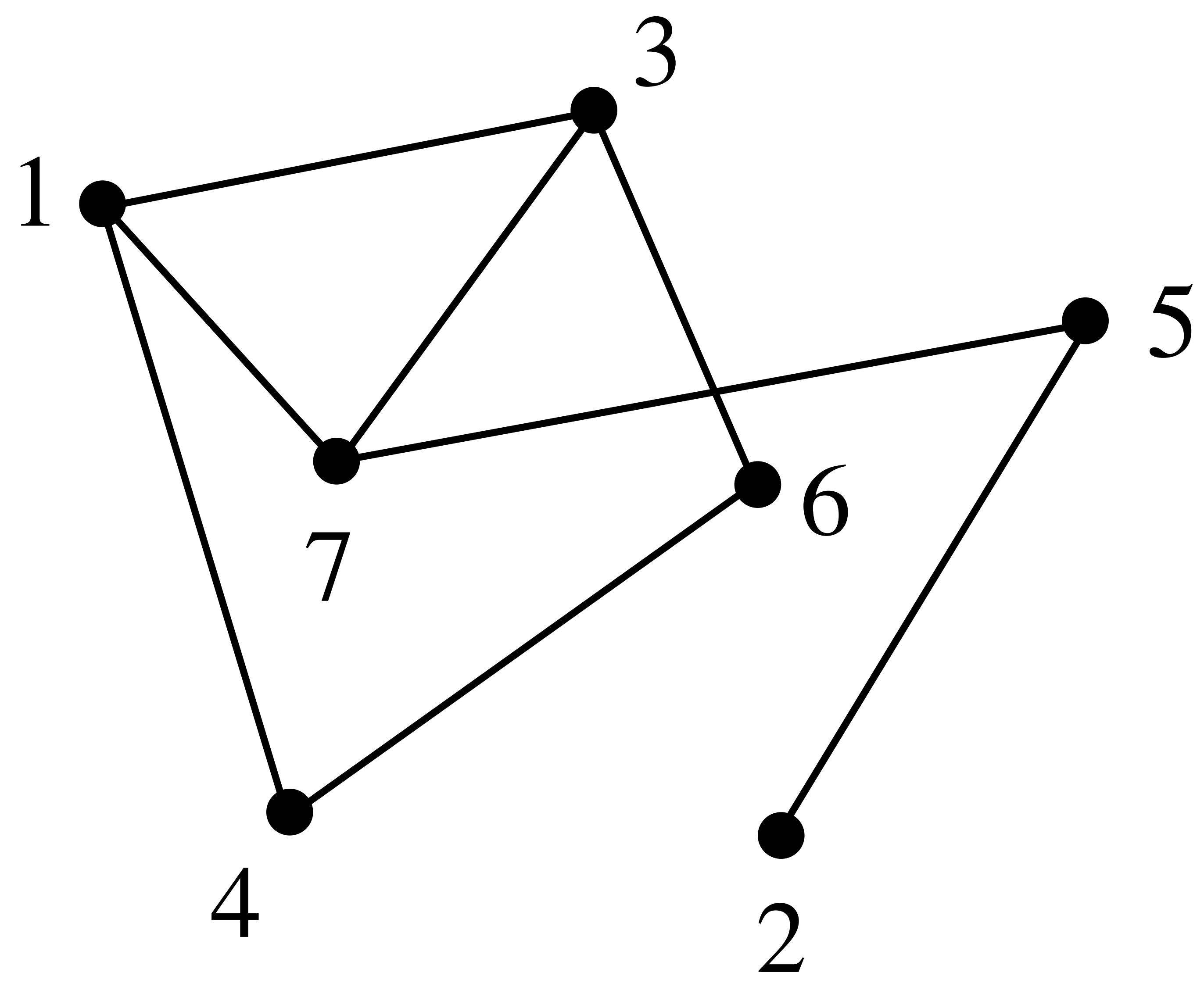
Graphs: Basics

Defn: An undirected graph G consists of a finite nonempty set V of objects called vertices and



Graphs: Basics

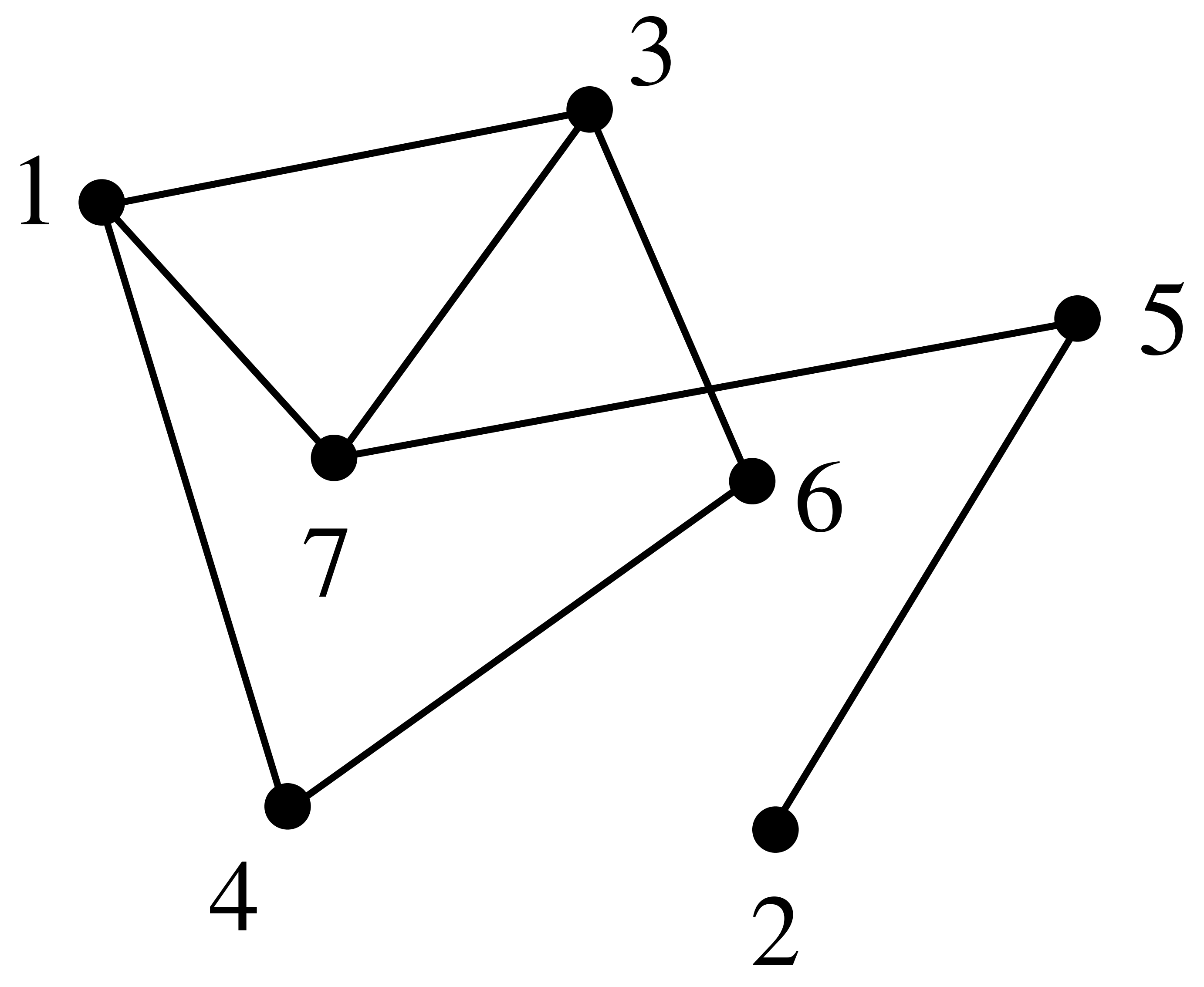
Defn: An undirected graph G consists of a finite nonempty set V of objects called vertices and set E of 2-element subsets of V called edges.



Graphs: Basics

Defn: An undirected graph G consists of a finite nonempty set V of objects called vertices and set E of 2-element subsets of V called edges.

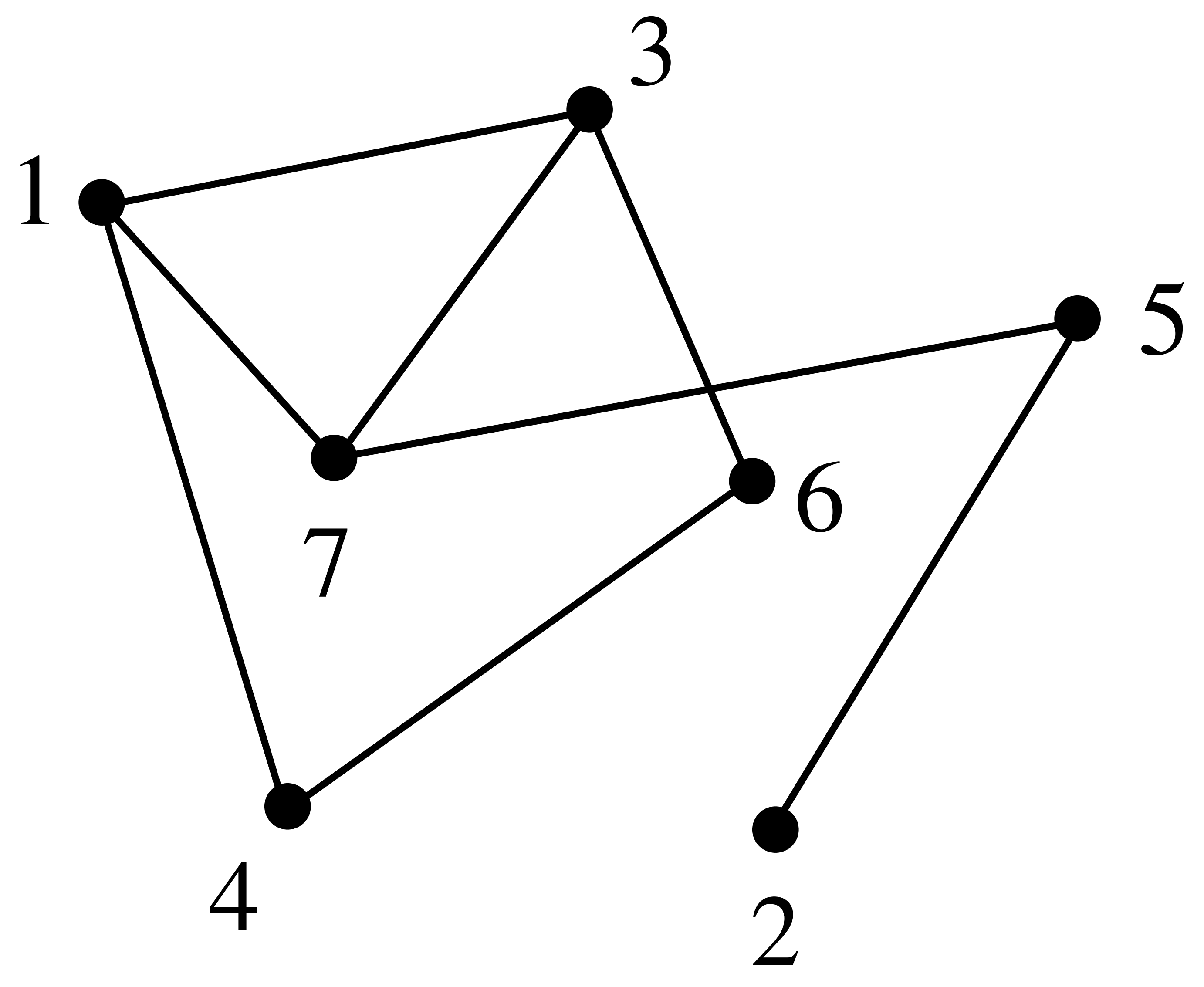
Example:



Graphs: Basics

Defn: An undirected graph G consists of a finite nonempty set V of objects called vertices and set E of 2-element subsets of V called edges.

Example: $G = (V, E)$

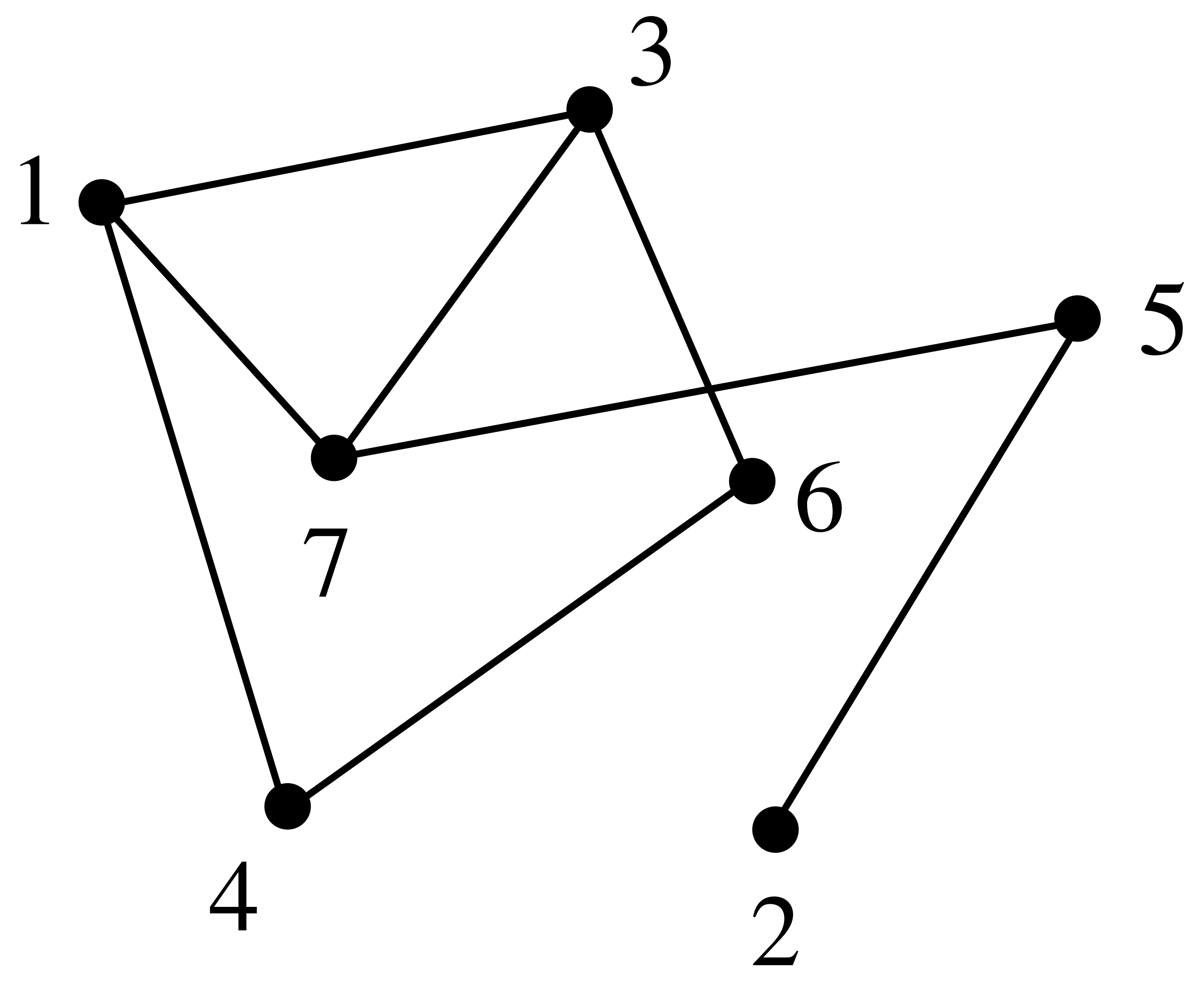


Graphs: Basics

Defn: An undirected graph G consists of a finite nonempty set V of objects called vertices and set E of 2-element subsets of V called edges.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$



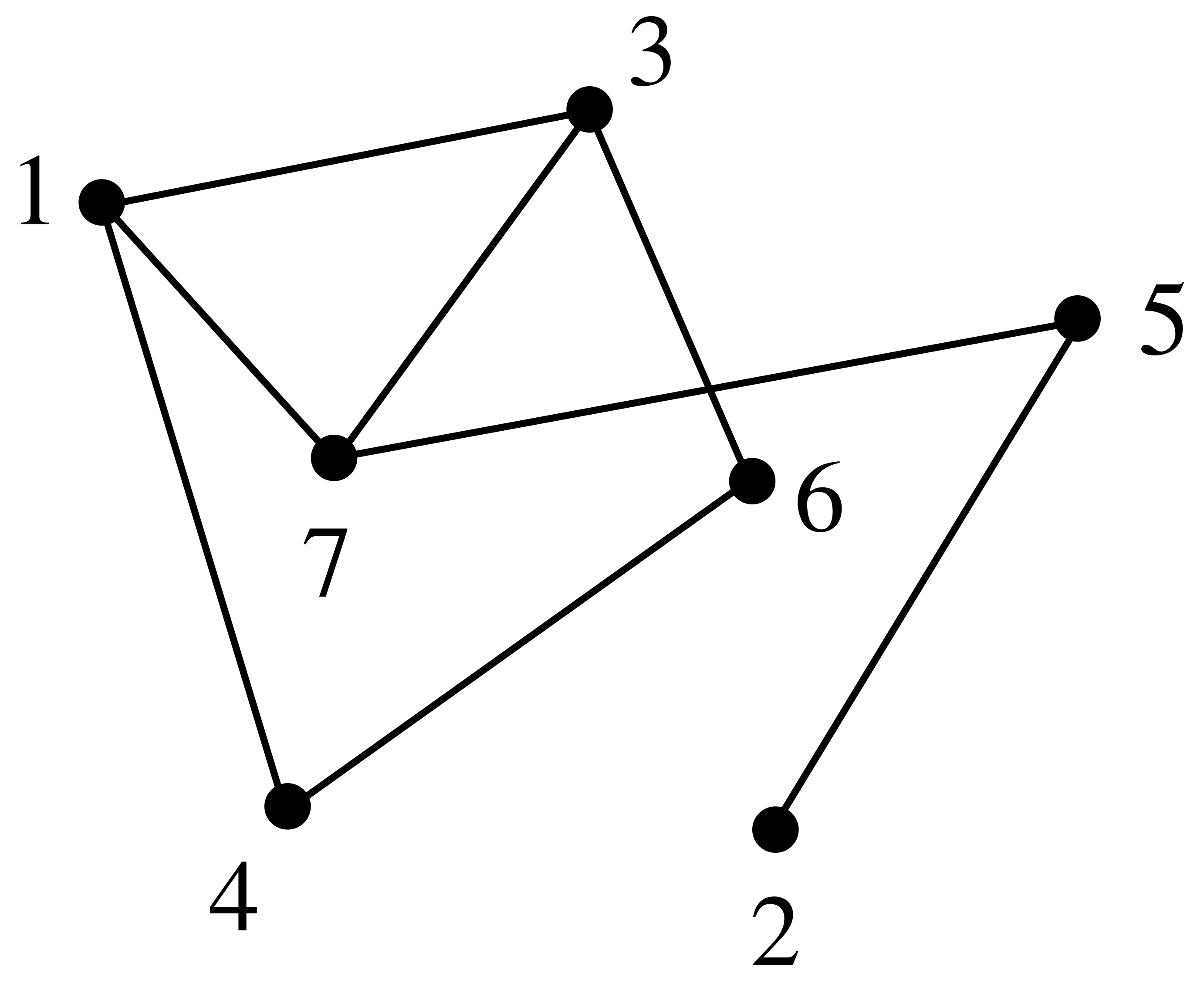
Graphs: Basics

Defn: An undirected graph G consists of a finite nonempty set V of objects called vertices and set E of 2-element subsets of V called edges.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$

$$E = \{\{1,4\}, \{1,3\}, \{1,7\}, \{2,5\}, \{3,7\}, \{3,6\}, \{4,6\}, \{5,7\}\}$$



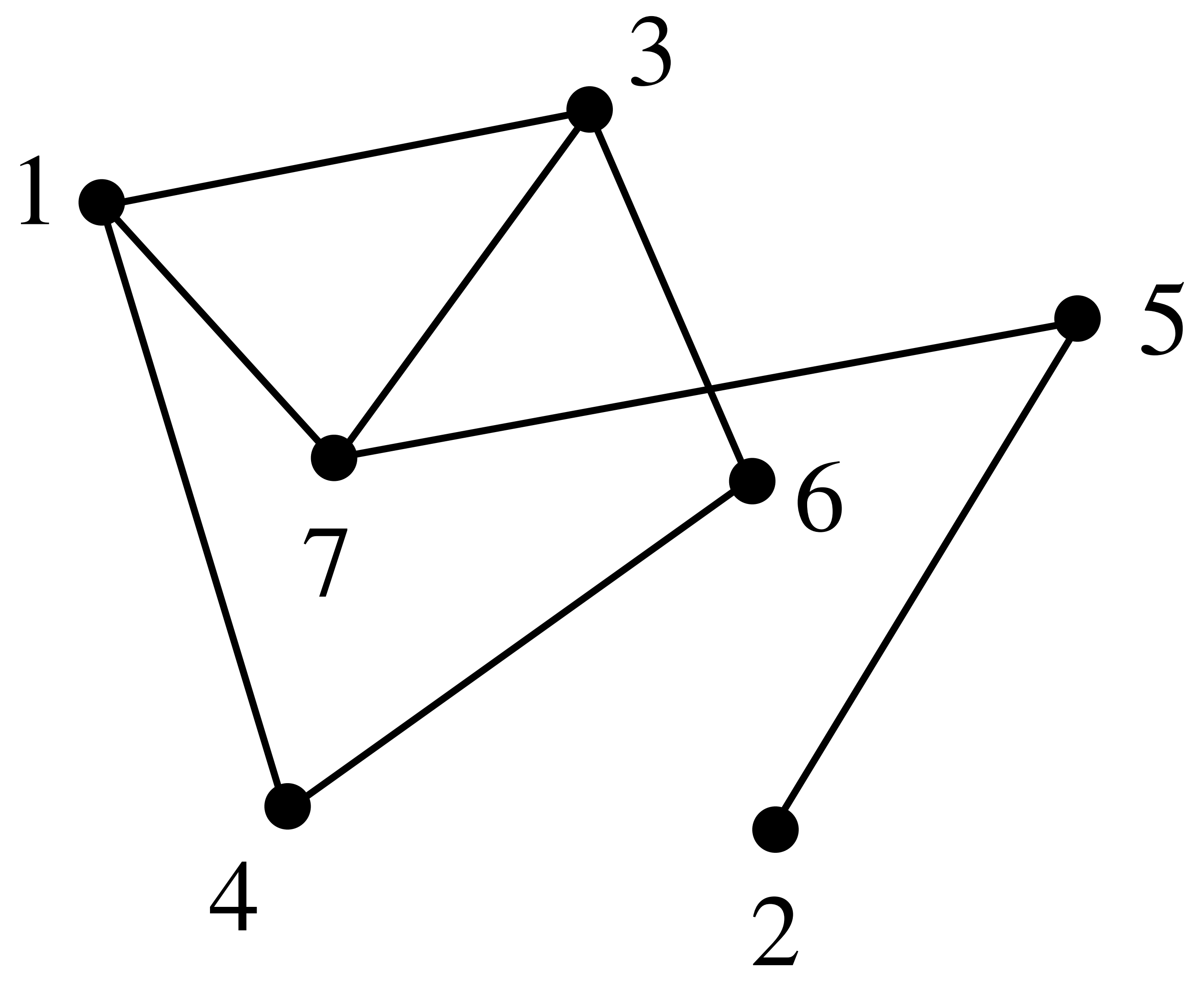
Graphs: Basics

Defn: An undirected graph G consists of a finite nonempty set V of objects called vertices and set E of 2-element subsets of V called edges.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$

$$E = \{\{1,4\}, \{1,3\}, \{1,7\}, \{2,5\}, \{3,7\}, \{3,6\}, \{4,6\}, \{5,7\}\}$$



Note: Undirected graphs do not have self loops,

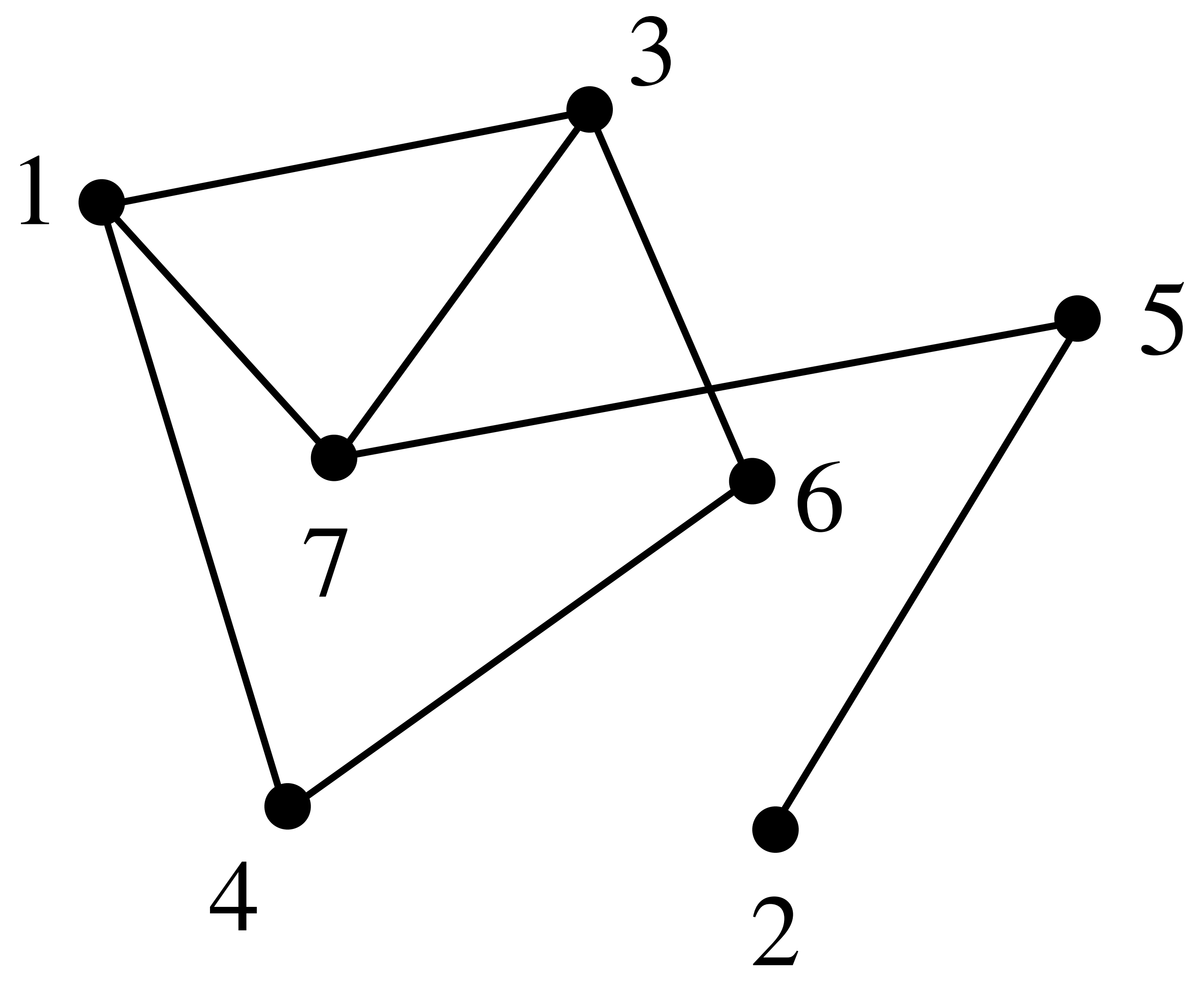
Graphs: Basics

Defn: An undirected graph G consists of a finite nonempty set V of objects called vertices and set E of 2-element subsets of V called edges.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$

$$E = \{\{1,4\}, \{1,3\}, \{1,7\}, \{2,5\}, \{3,7\}, \{3,6\}, \{4,6\}, \{5,7\}\}$$



Note: Undirected graphs do not have self loops, i.e, an edge from a vertex to itself

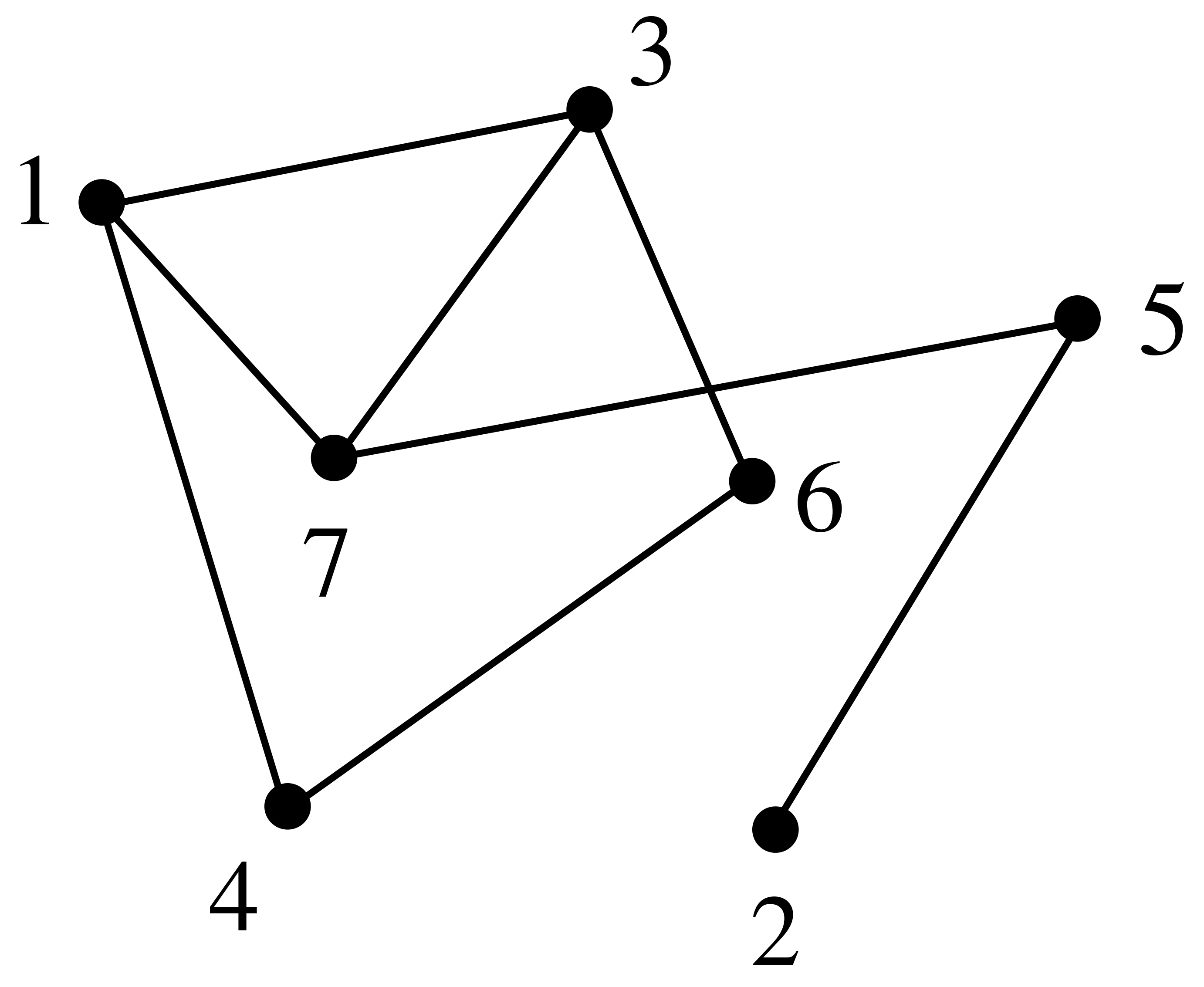
Graphs: Basics

Defn: An undirected graph G consists of a finite nonempty set V of objects called vertices and set E of 2-element subsets of V called edges.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$

$$E = \{\{1,4\}, \{1,3\}, \{1,7\}, \{2,5\}, \{3,7\}, \{3,6\}, \{4,6\}, \{5,7\}\}$$



Note: Undirected graphs do not have self loops, i.e, an edge from a vertex to itself and parallel edges,

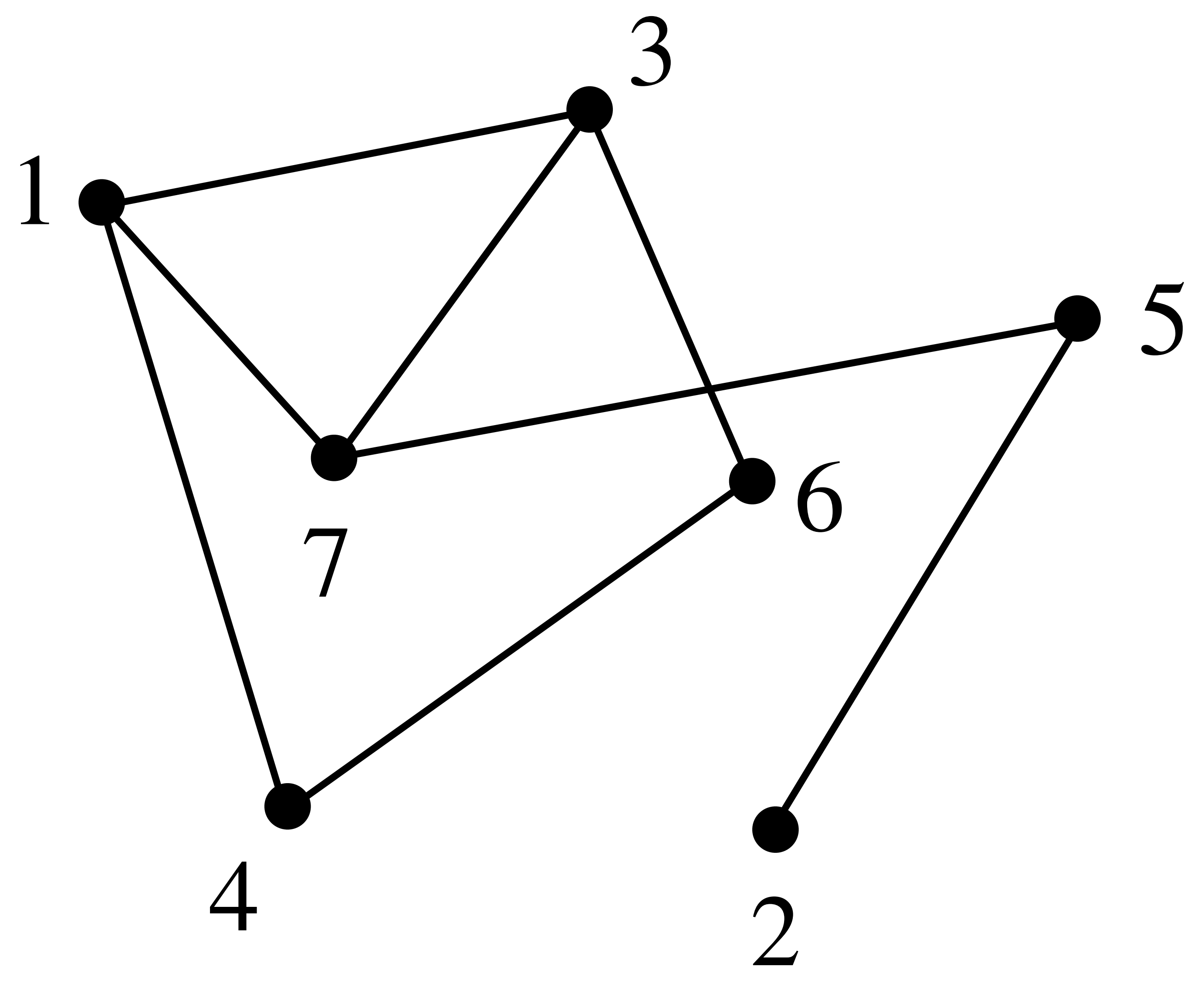
Graphs: Basics

Defn: An undirected graph G consists of a finite nonempty set V of objects called vertices and set E of 2-element subsets of V called edges.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$

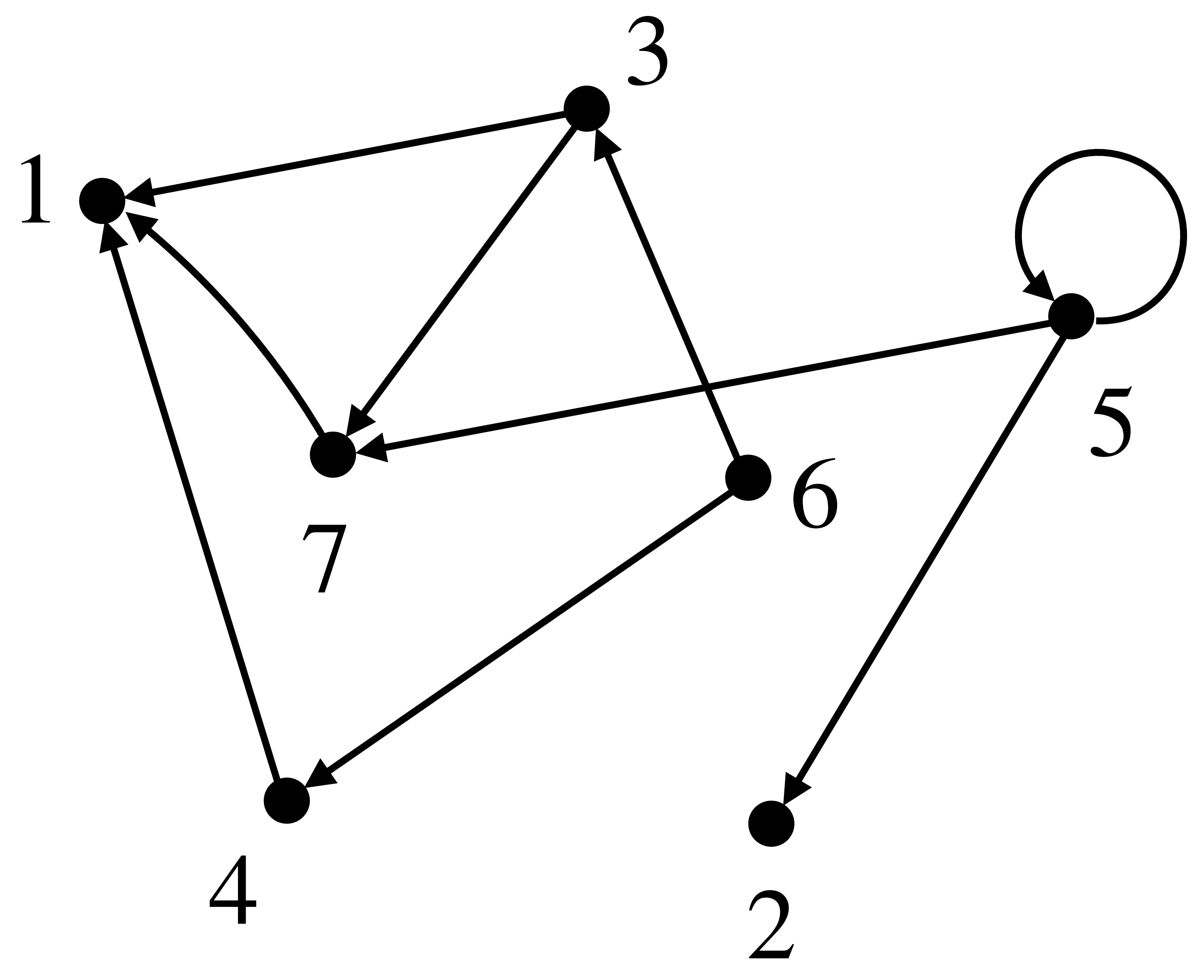
$$E = \{\{1,4\}, \{1,3\}, \{1,7\}, \{2,5\}, \{3,7\}, \{3,6\}, \{4,6\}, \{5,7\}\}$$



Note: Undirected graphs do not have self loops, i.e, an edge from a vertex to itself and parallel edges, i.e., more than one edges between two vertices.

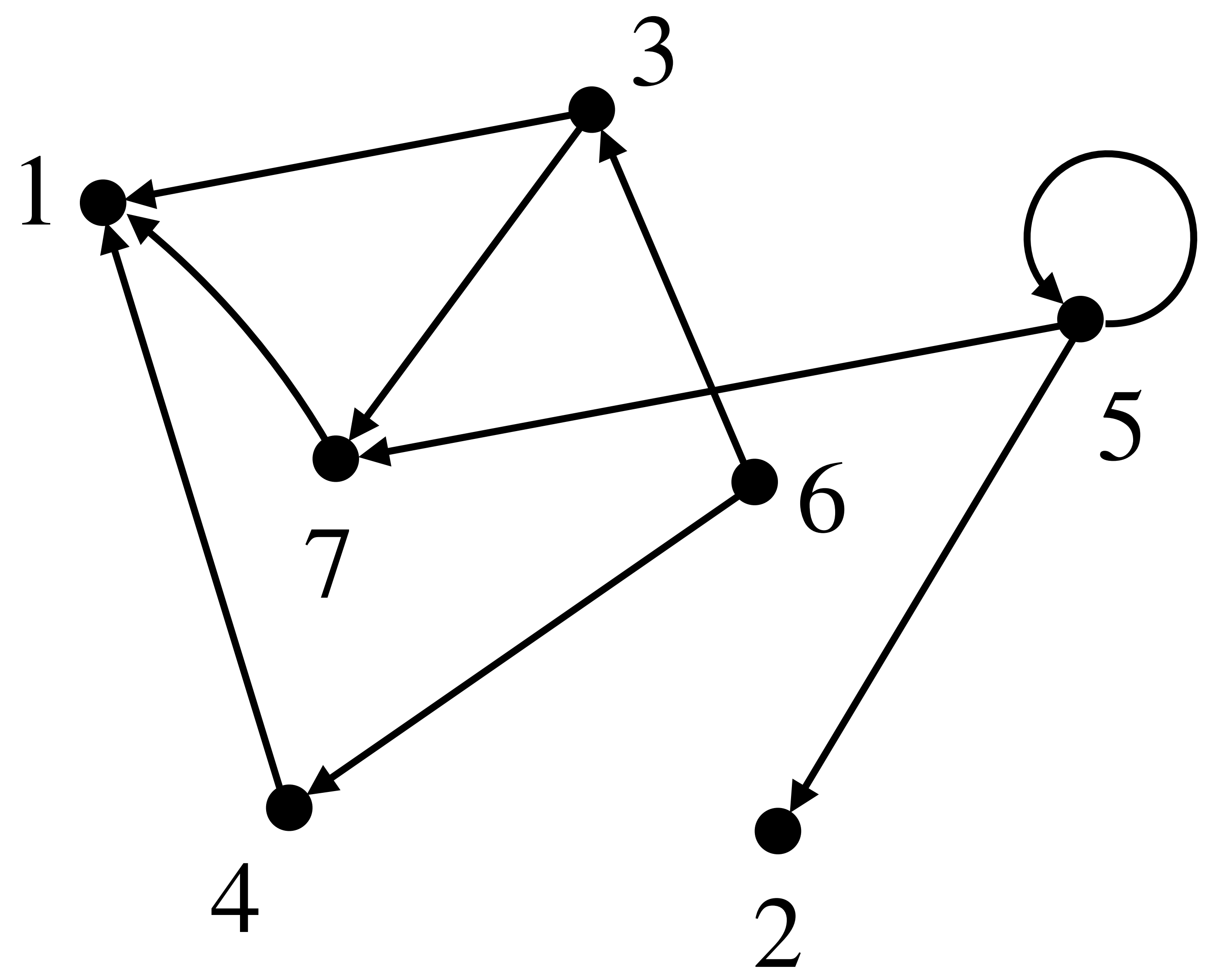
Graphs: Basics

Graphs: Basics



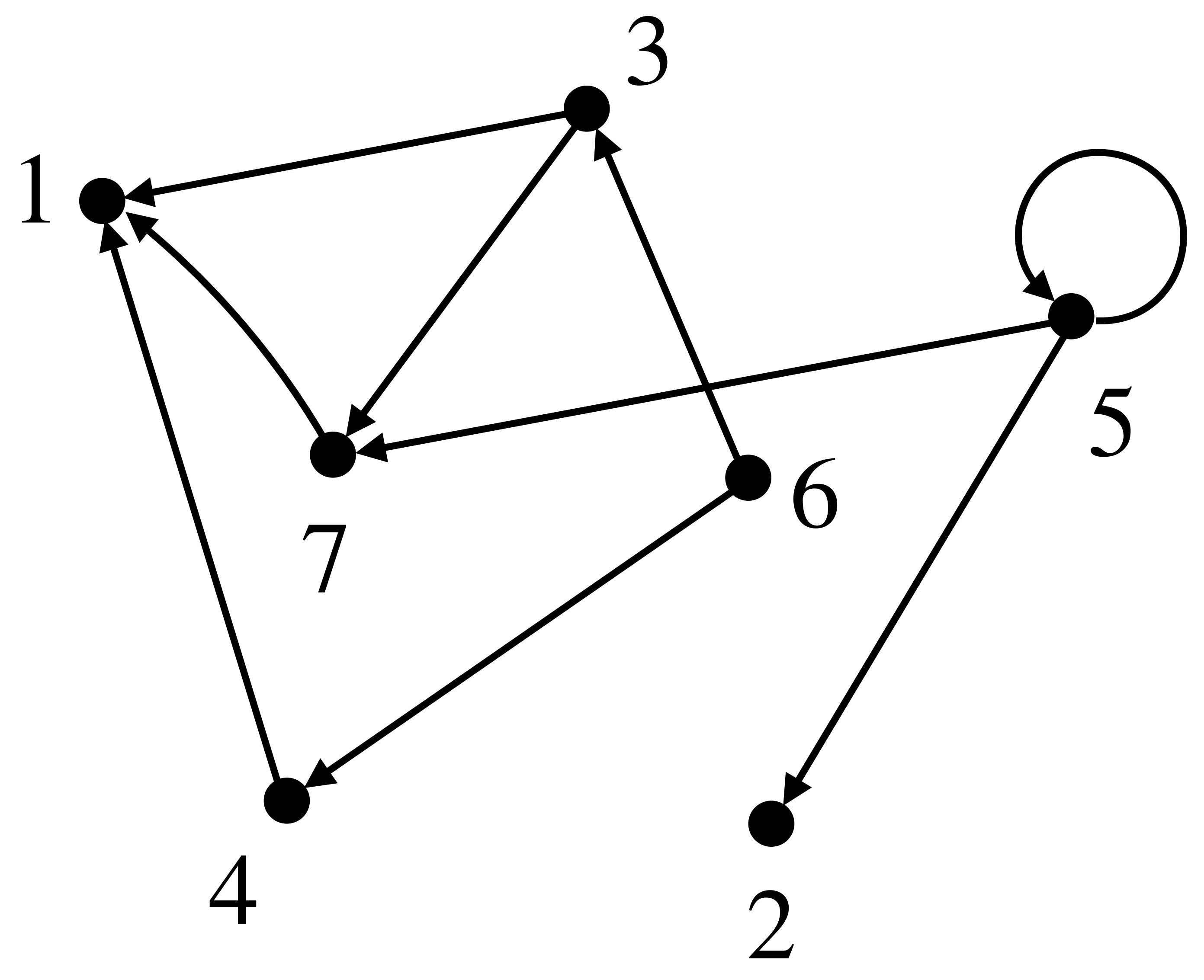
Graphs: Basics

Defn: A directed graph G consists of a finite nonempty set V of objects called vertices and



Graphs: Basics

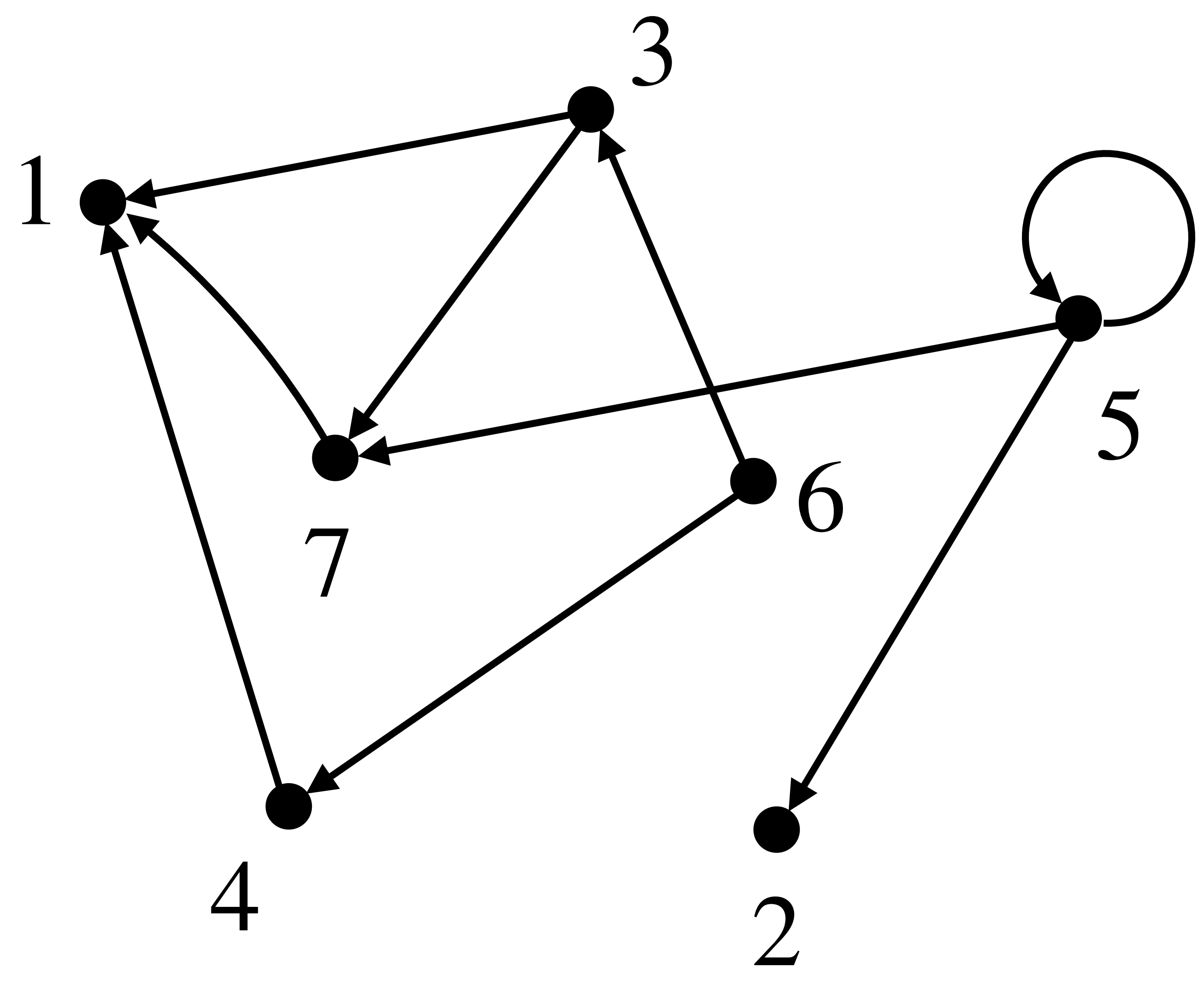
Defn: A directed graph G consists of a finite nonempty set V of objects called vertices and set E called edges which is a binary relation on V.



Graphs: Basics

Defn: A directed graph G consists of a finite nonempty set V of objects called vertices and set E called edges which is a binary relation on V.

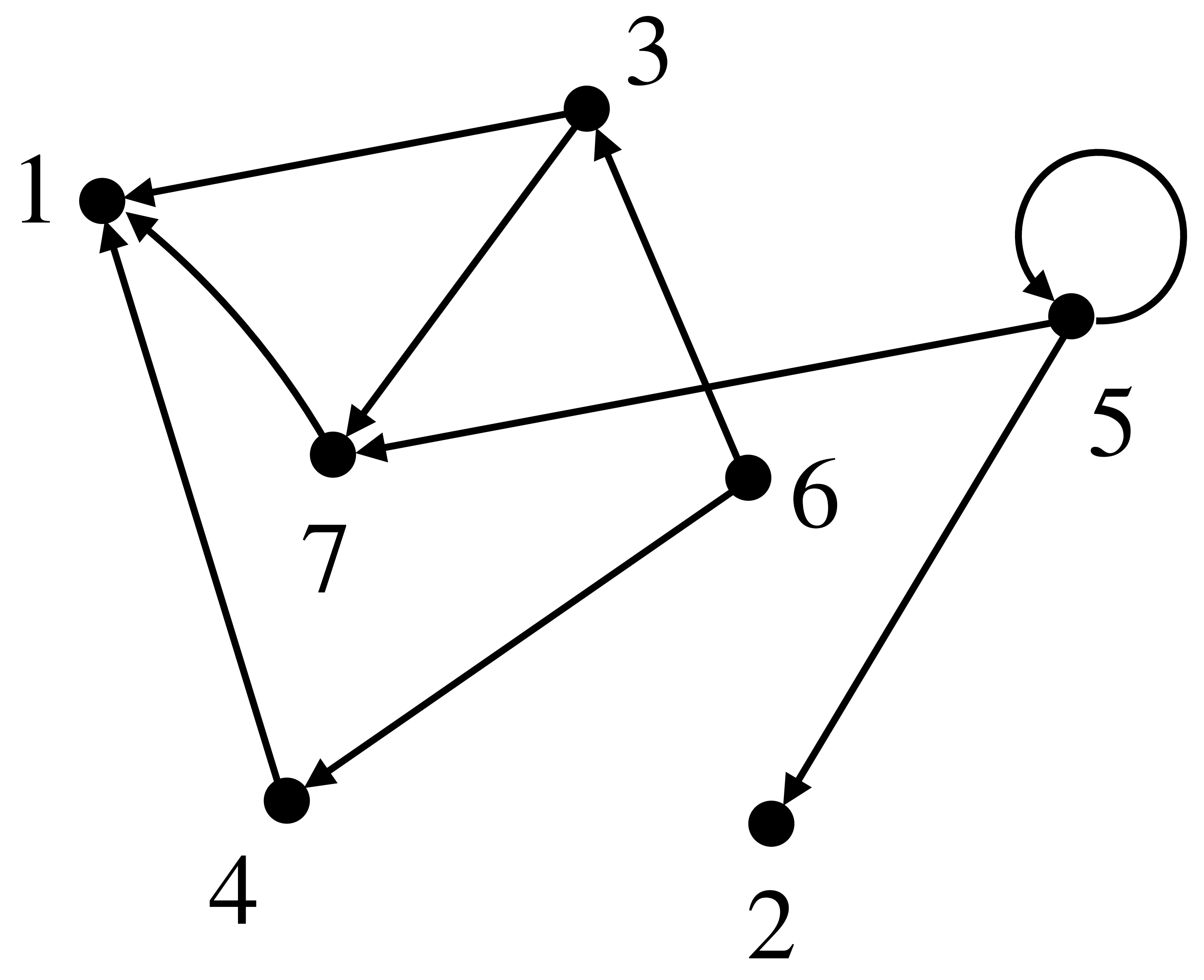
Example:



Graphs: Basics

Defn: A directed graph G consists of a finite nonempty set V of objects called vertices and set E called edges which is a binary relation on V.

Example: $G = (V, E)$

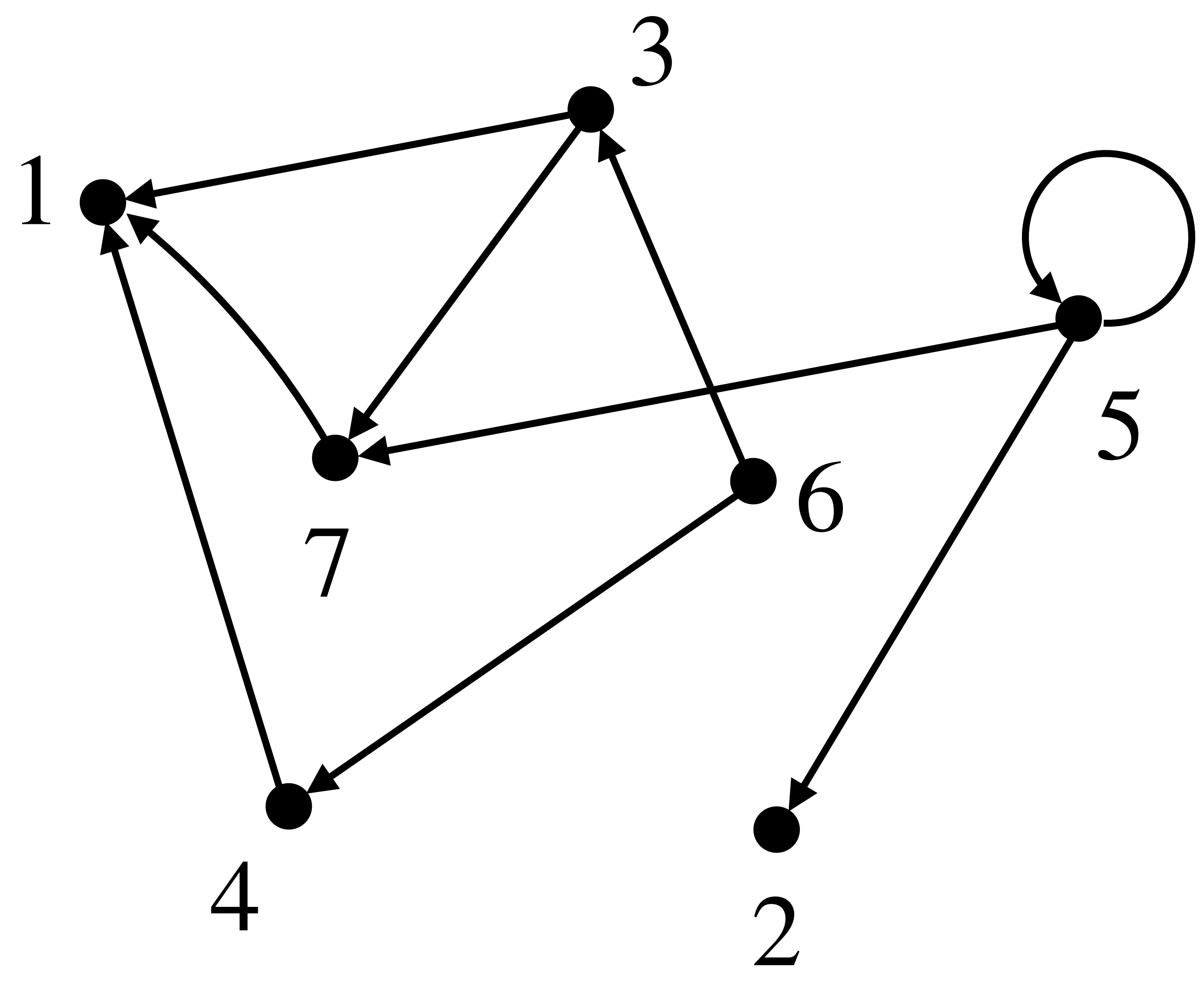


Graphs: Basics

Defn: A directed graph G consists of a finite nonempty set V of objects called vertices and set E called edges which is a binary relation on V.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$



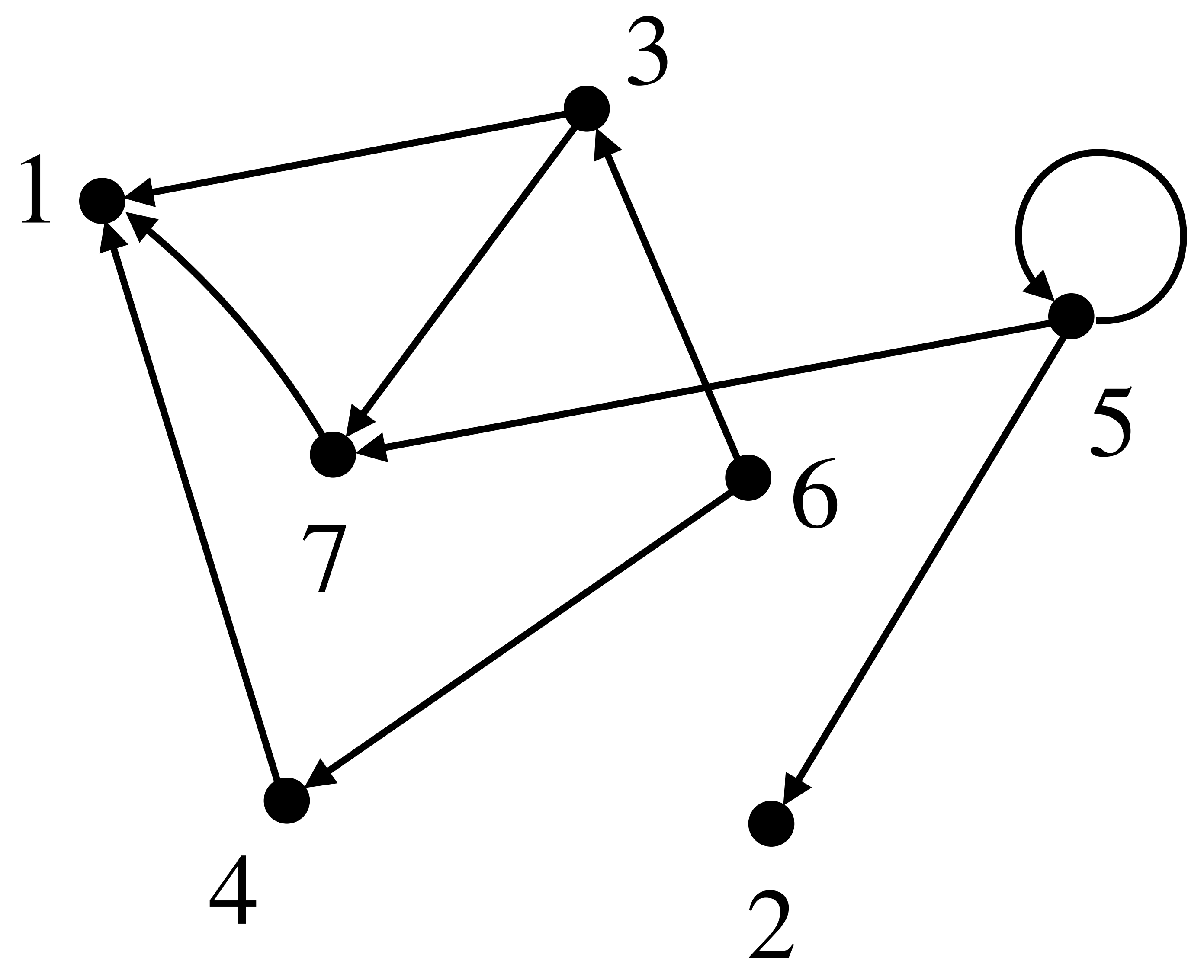
Graphs: Basics

Defn: A directed graph G consists of a finite nonempty set V of objects called vertices and set E called edges which is a binary relation on V.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$

$$E = \{(3,1), (3,7), (4,1), (5,2), (5,5), (5,7), (6,3), (6,4), (7,1)\}$$



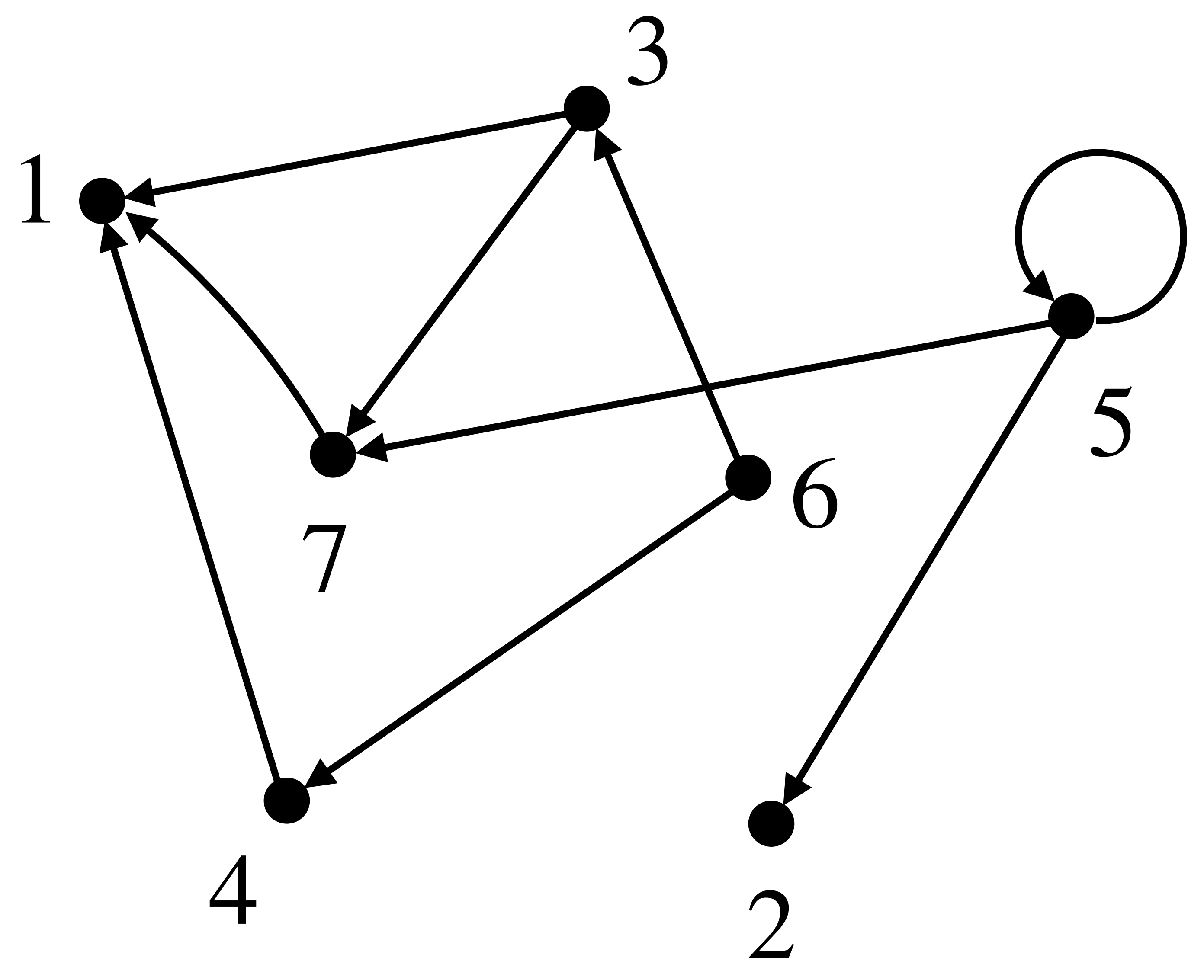
Graphs: Basics

Defn: A directed graph G consists of a finite nonempty set V of objects called vertices and set E called edges which is a binary relation on V.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$

$$E = \{(3,1), (3,7), (4,1), (5,2), (5,5), (5,7), (6,3), (6,4), (7,1)\}$$



Note: Directed graphs may have self loops,

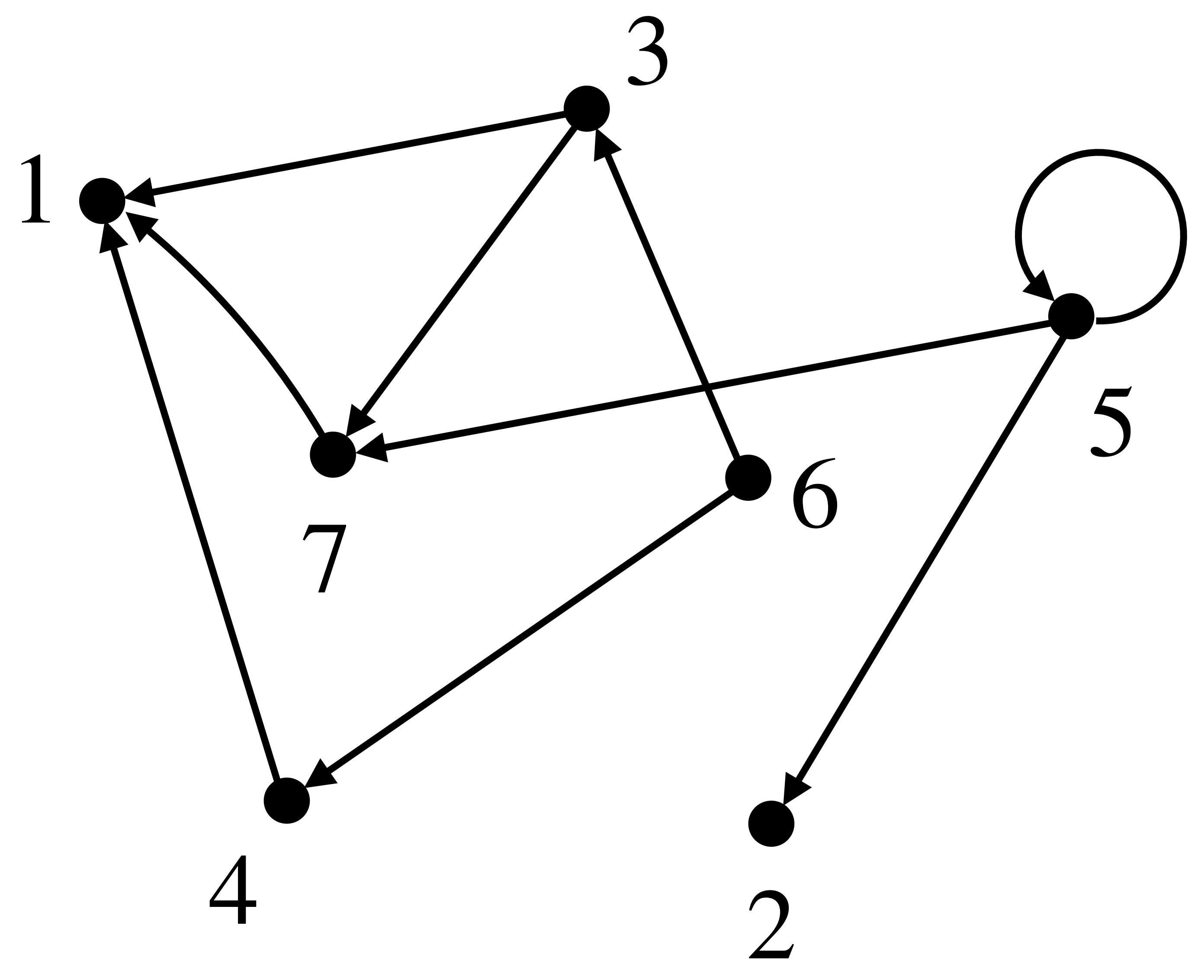
Graphs: Basics

Defn: A directed graph G consists of a finite nonempty set V of objects called vertices and set E called edges which is a binary relation on V.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$

$$E = \{(3,1), (3,7), (4,1), (5,2), (5,5), (5,7), (6,3), (6,4), (7,1)\}$$



Note: Directed graphs may have self loops, i.e, an edge from a vertex to itself,

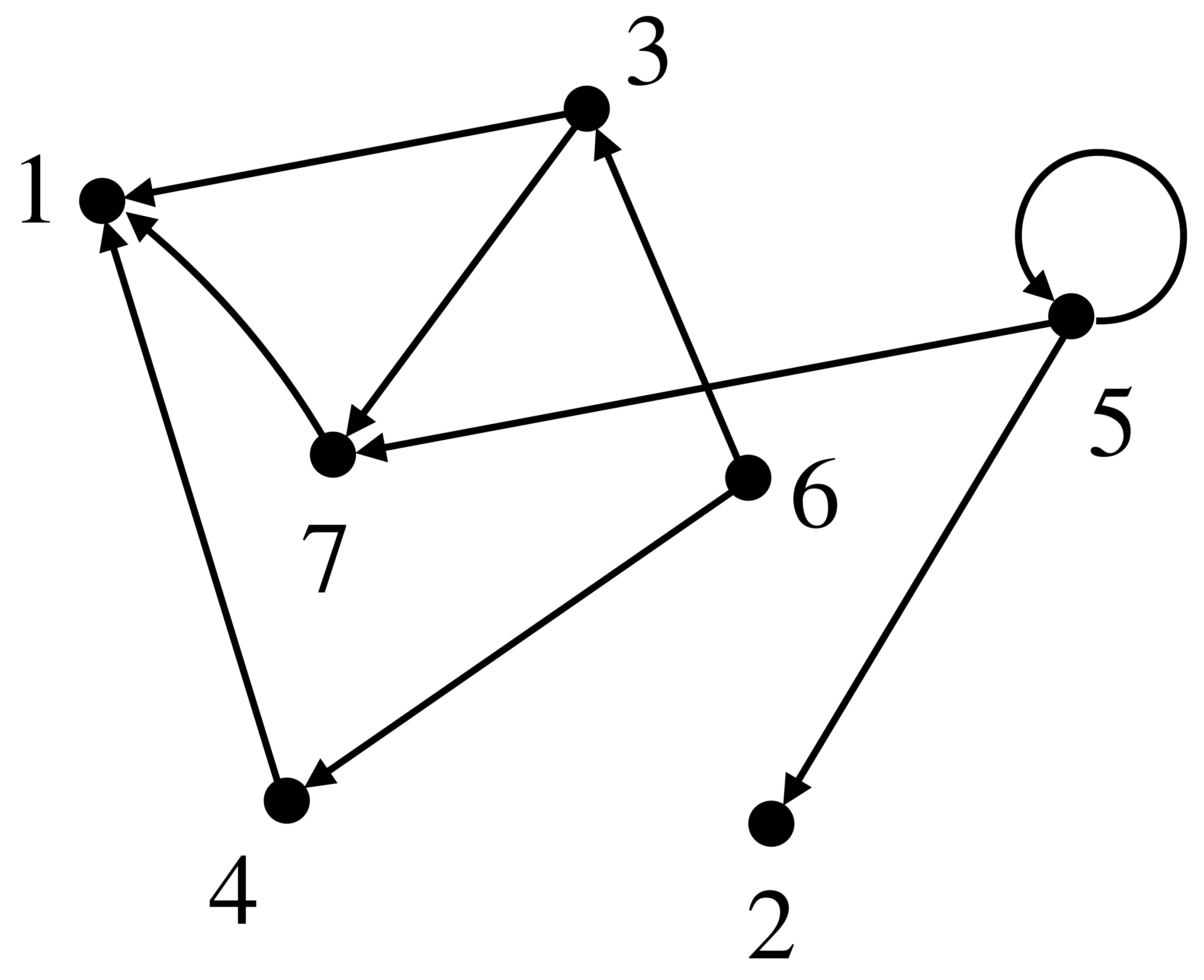
Graphs: Basics

Defn: A directed graph G consists of a finite nonempty set V of objects called vertices and set E called edges which is a binary relation on V.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$

$$E = \{(3,1), (3,7), (4,1), (5,2), (5,5), (5,7), (6,3), (6,4), (7,1)\}$$



Note: Directed graphs may have self loops, i.e, an edge from a vertex to itself, but no parallel edges,

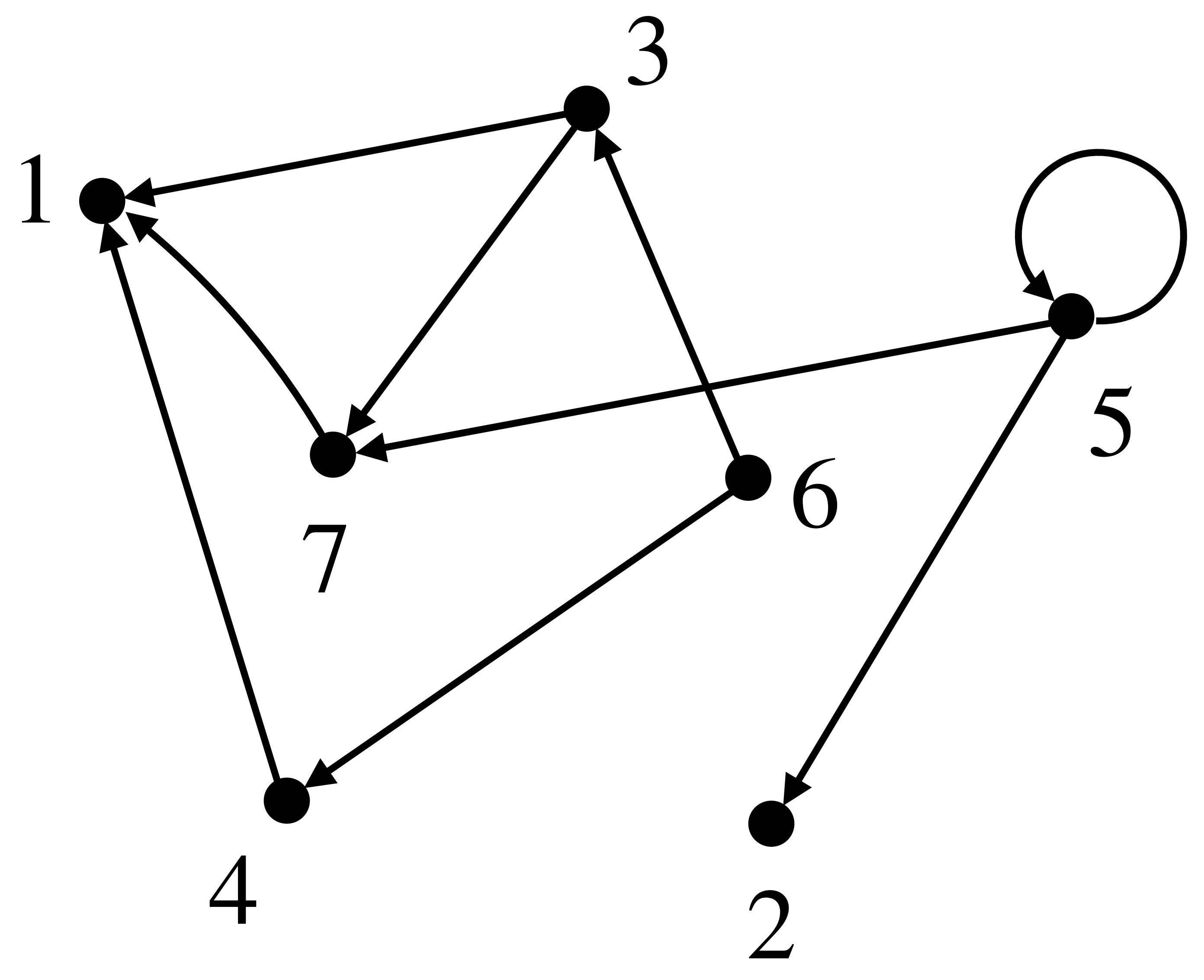
Graphs: Basics

Defn: A directed graph G consists of a finite nonempty set V of objects called vertices and set E called edges which is a binary relation on V.

Example: $G = (V, E)$

$$V = \{1,2,3,4,5,6,7\}$$

$$E = \{(3,1), (3,7), (4,1), (5,2), (5,5), (5,7), (6,3), (6,4), (7,1)\}$$



Note: Directed graphs may have self loops, i.e, an edge from a vertex to itself, but no parallel edges, i.e., more than one edges between two vertices.

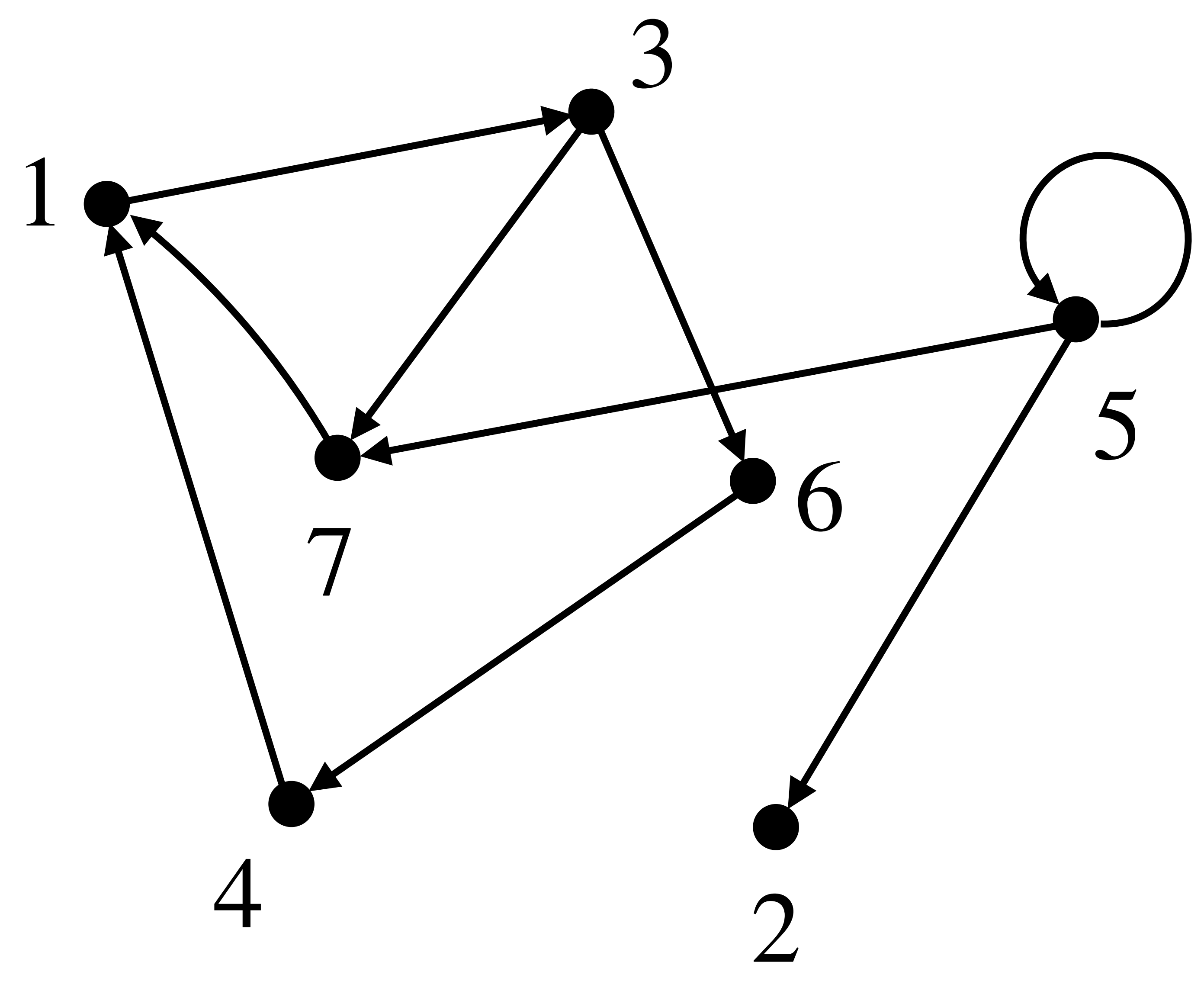
Graphs: Basic Terminology

Graphs: Basic Terminology

- Two vertices, say u and v, are called adjacent (or neighbours) if (u, v) or (v, u) is an edge.

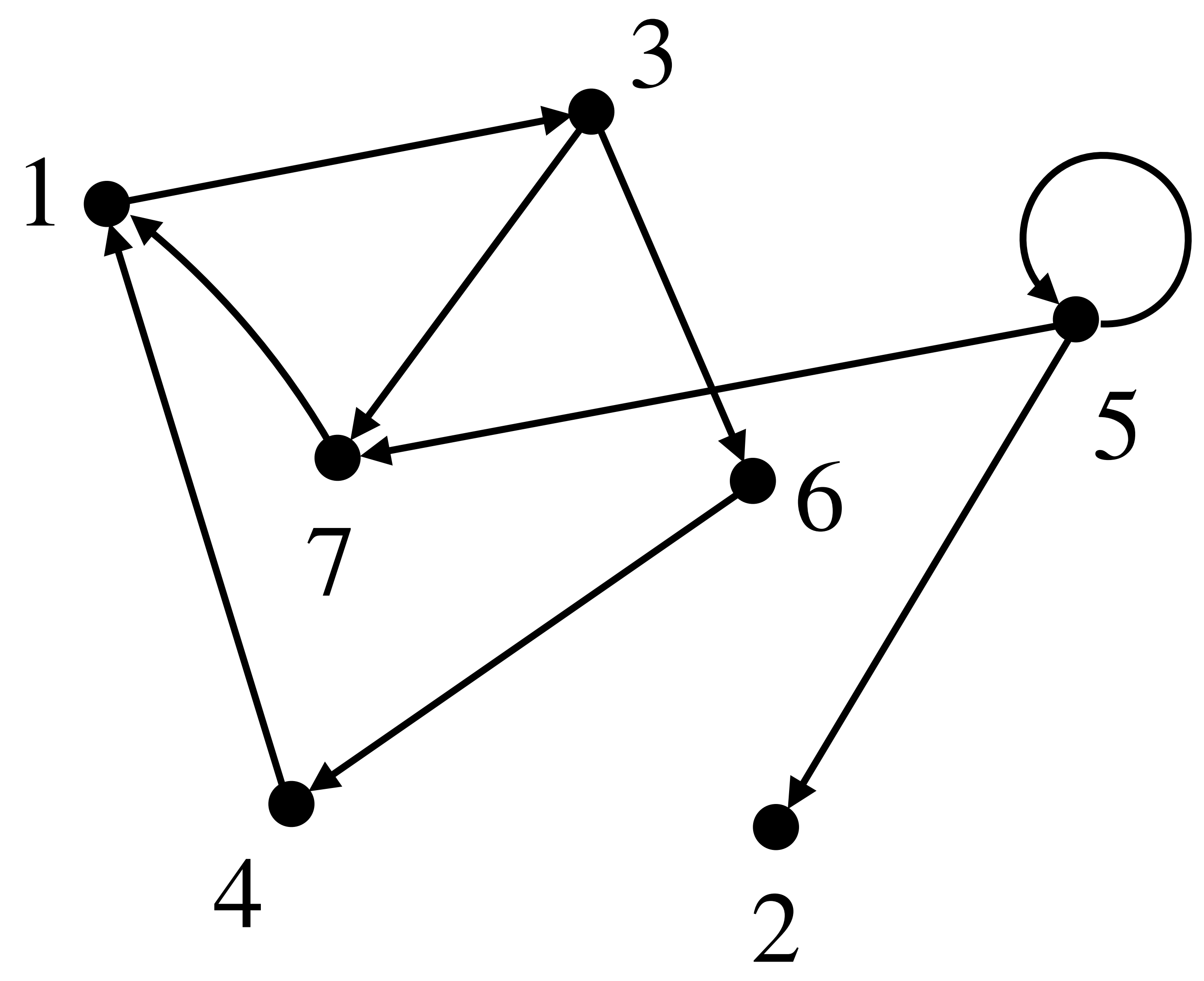
Graphs: Basic Terminology

- Two vertices, say u and v, are called adjacent (or neighbours) if (u, v) or (v, u) is an edge.



Graphs: Basic Terminology

- Two vertices, say u and v, are called adjacent (or neighbours) if (u, v) or (v, u) is an edge.

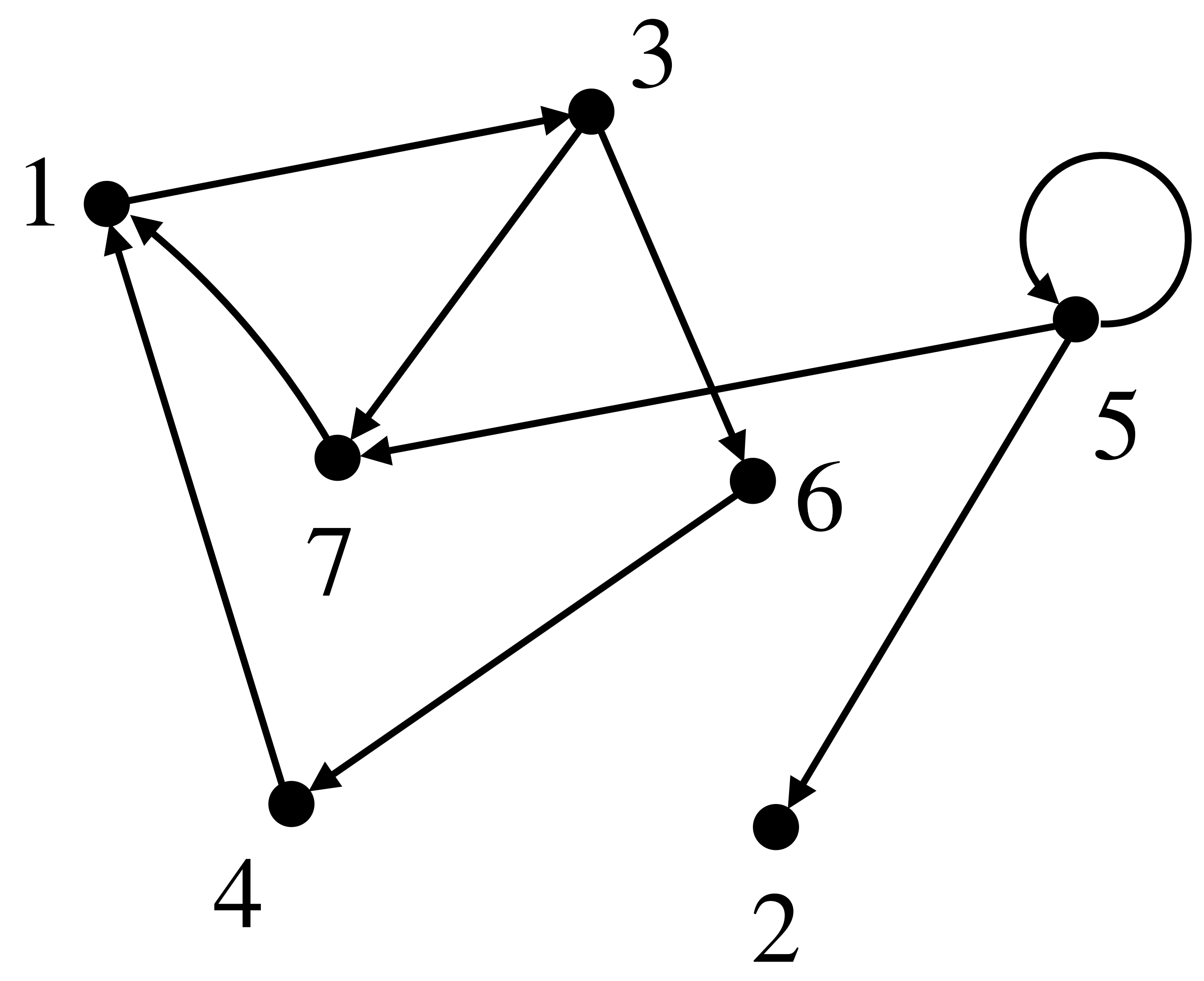


1 and 3 are adjacent

Graphs: Basic Terminology

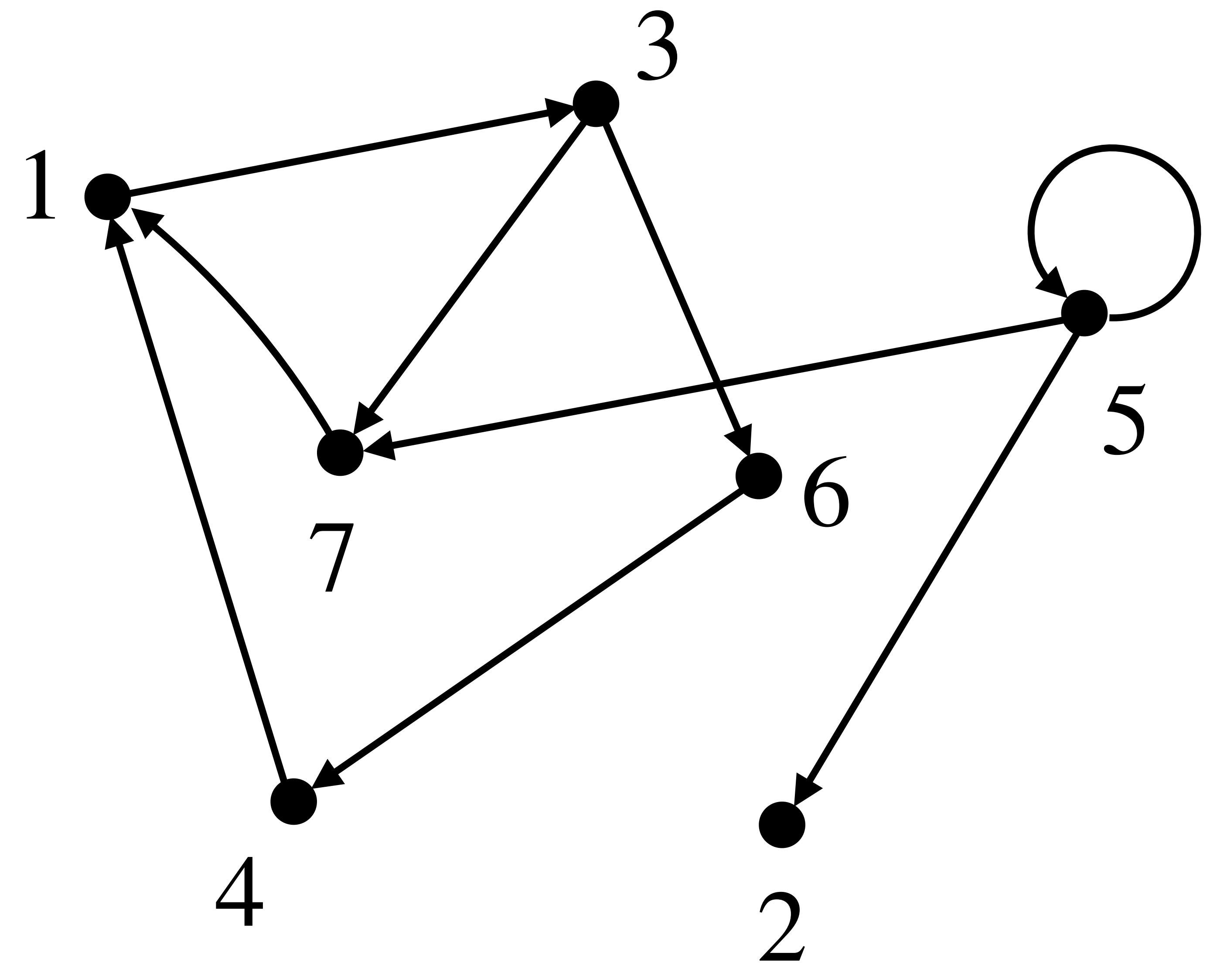
- Two vertices, say u and v, are called adjacent (or neighbours) if (u, v) or (v, u) is an edge.

- Vertices u and v are said to be incident on an edge e, if $e = (u, v)$ or $e = (v, u)$.



Graphs: Basic Terminology

- Two vertices, say u and v, are called adjacent (or neighbours) if (u, v) or (v, u) is an edge.
- Vertices u and v are said to be incident on an edge e, if $e = (u, v)$ or $e = (v, u)$.

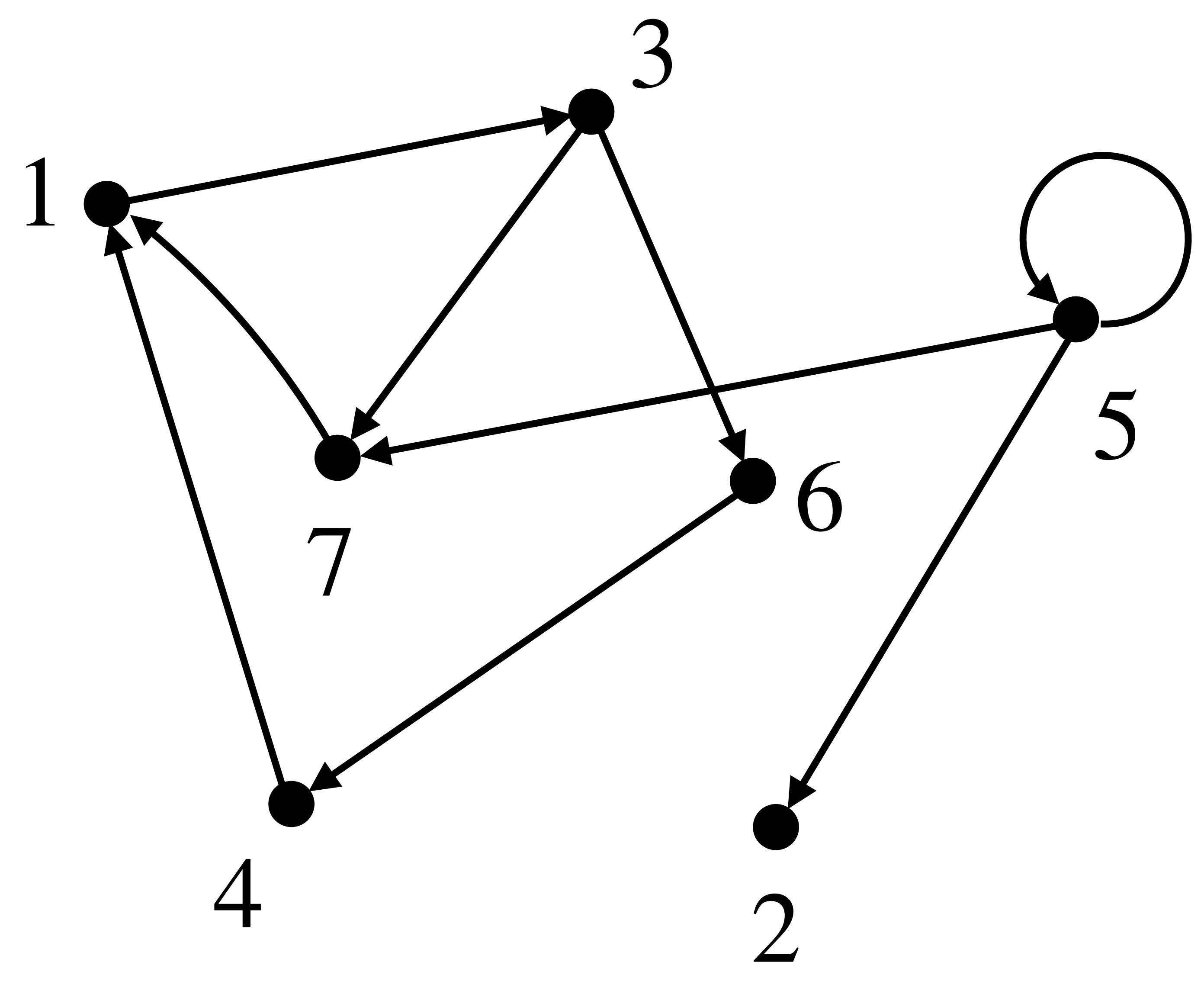


4 is incident on
(4,1), (6,4) edges

Graphs: Basic Terminology

- Two vertices, say u and v, are called adjacent (or neighbours) if (u, v) or (v, u) is an edge.

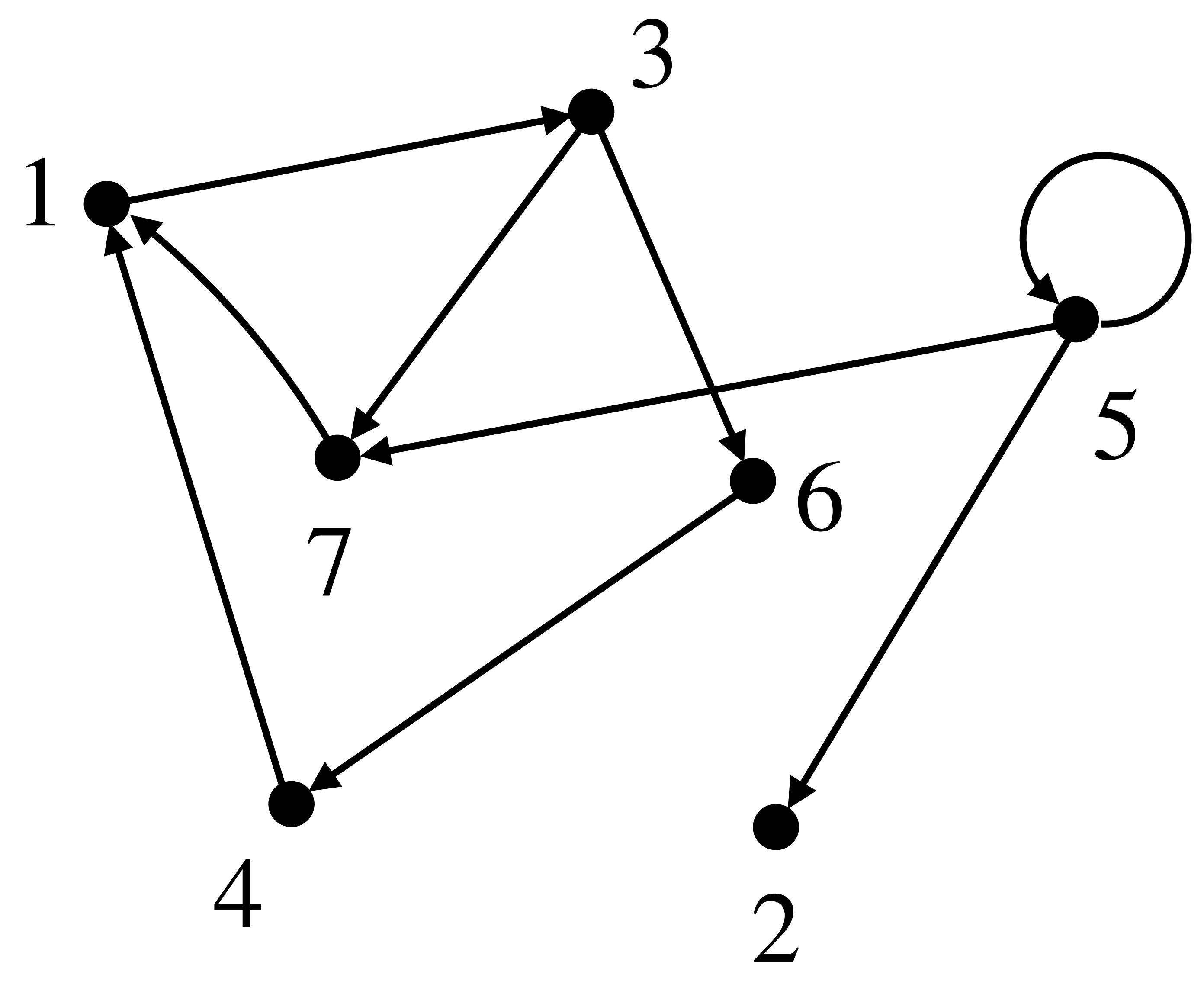
- Vertices u and v are said to be incident on an edge e, if $e = (u, v)$ or $e = (v, u)$.



- The degree of a vertex is the number of edges it is incident with.

Graphs: Basic Terminology

- Two vertices, say u and v, are called adjacent (or neighbours) if (u, v) or (v, u) is an edge.
- Vertices u and v are said to be incident on an edge e, if $e = (u, v)$ or $e = (v, u)$.

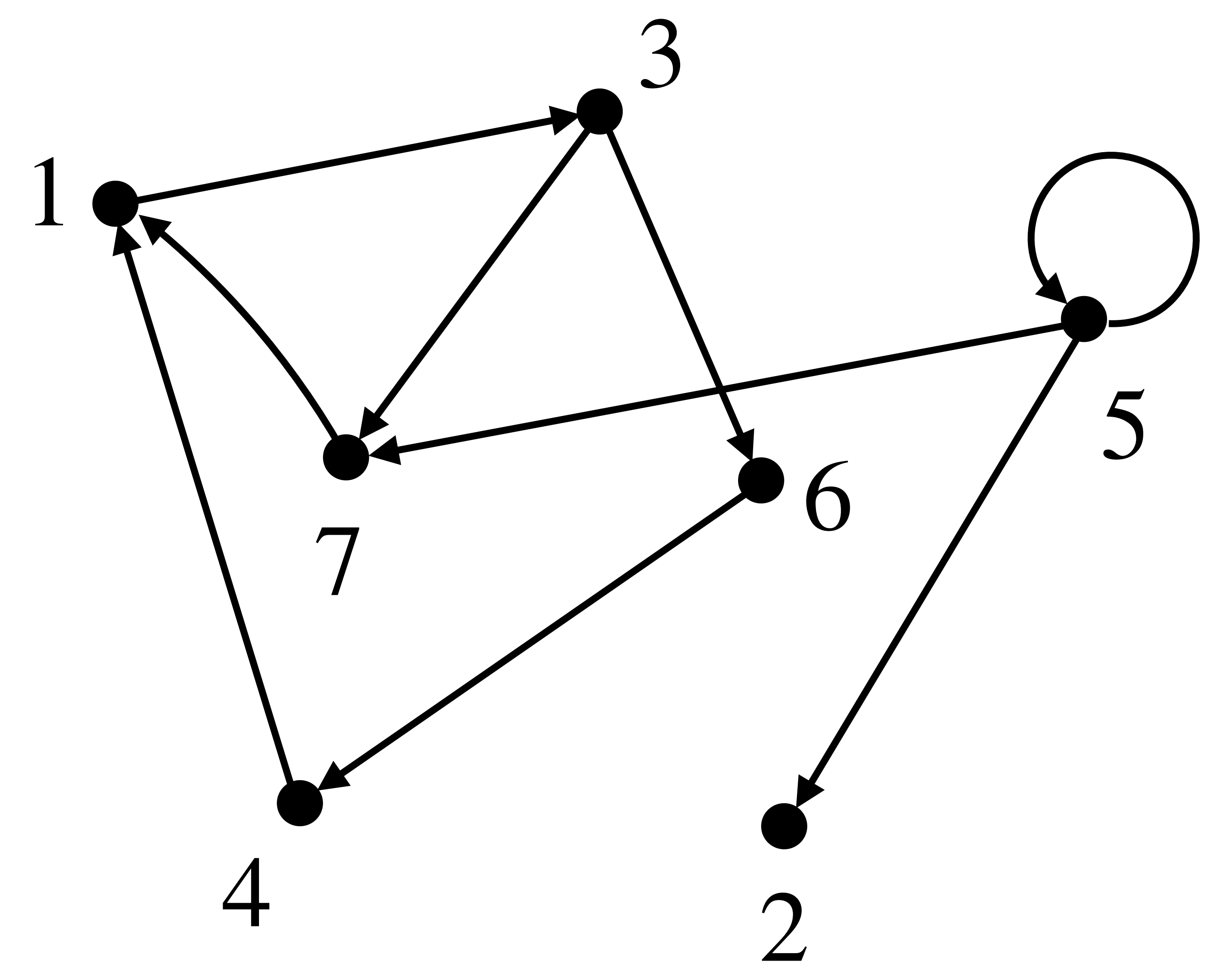


$degree(1) = 3$

- The degree of a vertex is the number of edges it is incident with.

Graphs: Basic Terminology

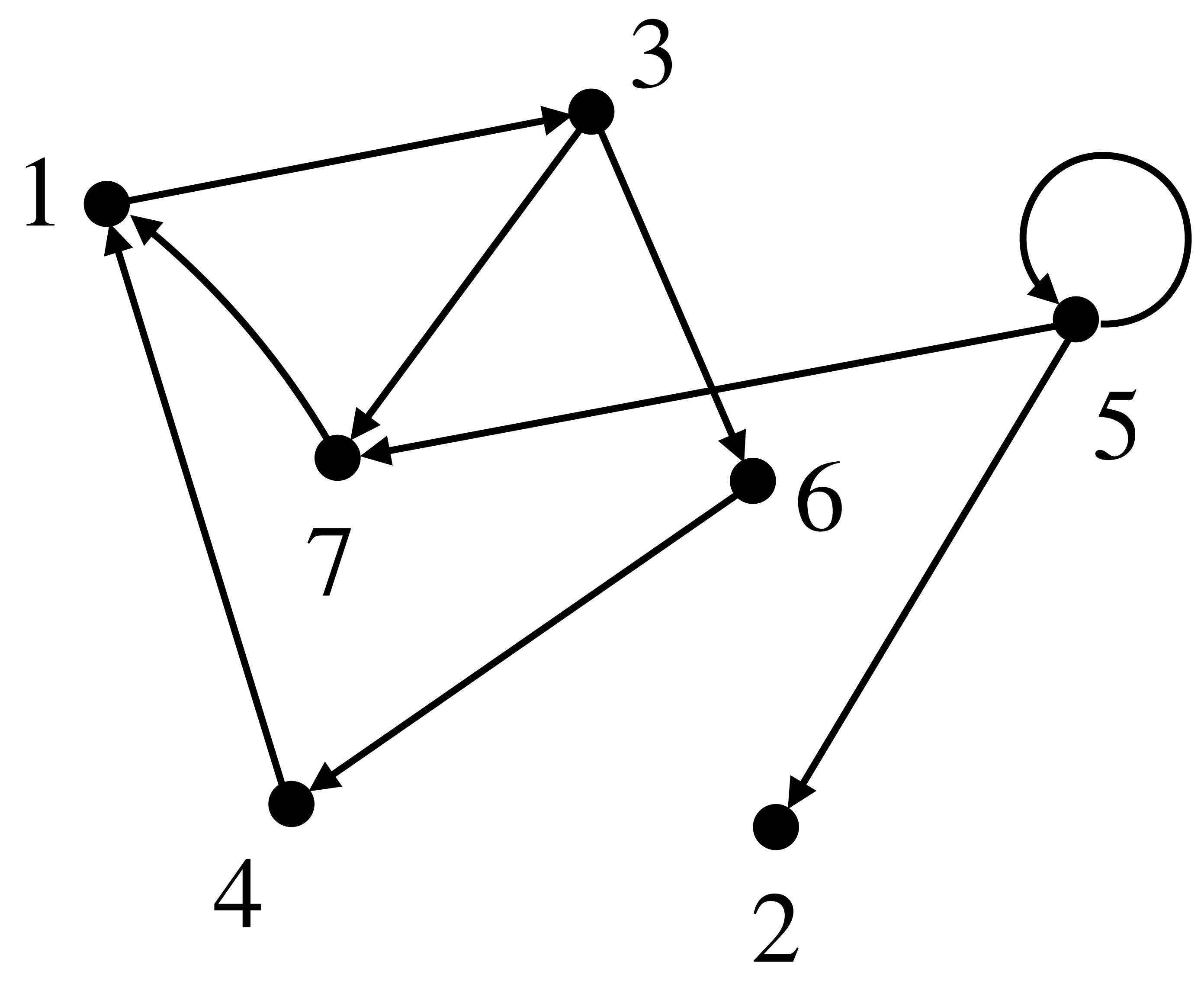
- Two vertices, say u and v, are called adjacent (or neighbours) if (u, v) or (v, u) is an edge.
- Vertices u and v are said to be incident on an edge e, if $e = (u, v)$ or $e = (v, u)$.



- The degree of a vertex is the number of edges it is incident with.
- A path from a vertex u to a vertex v is a sequence of vertices $\langle v_0, v_1, v_2, \ldots, v_k \rangle$ such that

Graphs: Basic Terminology

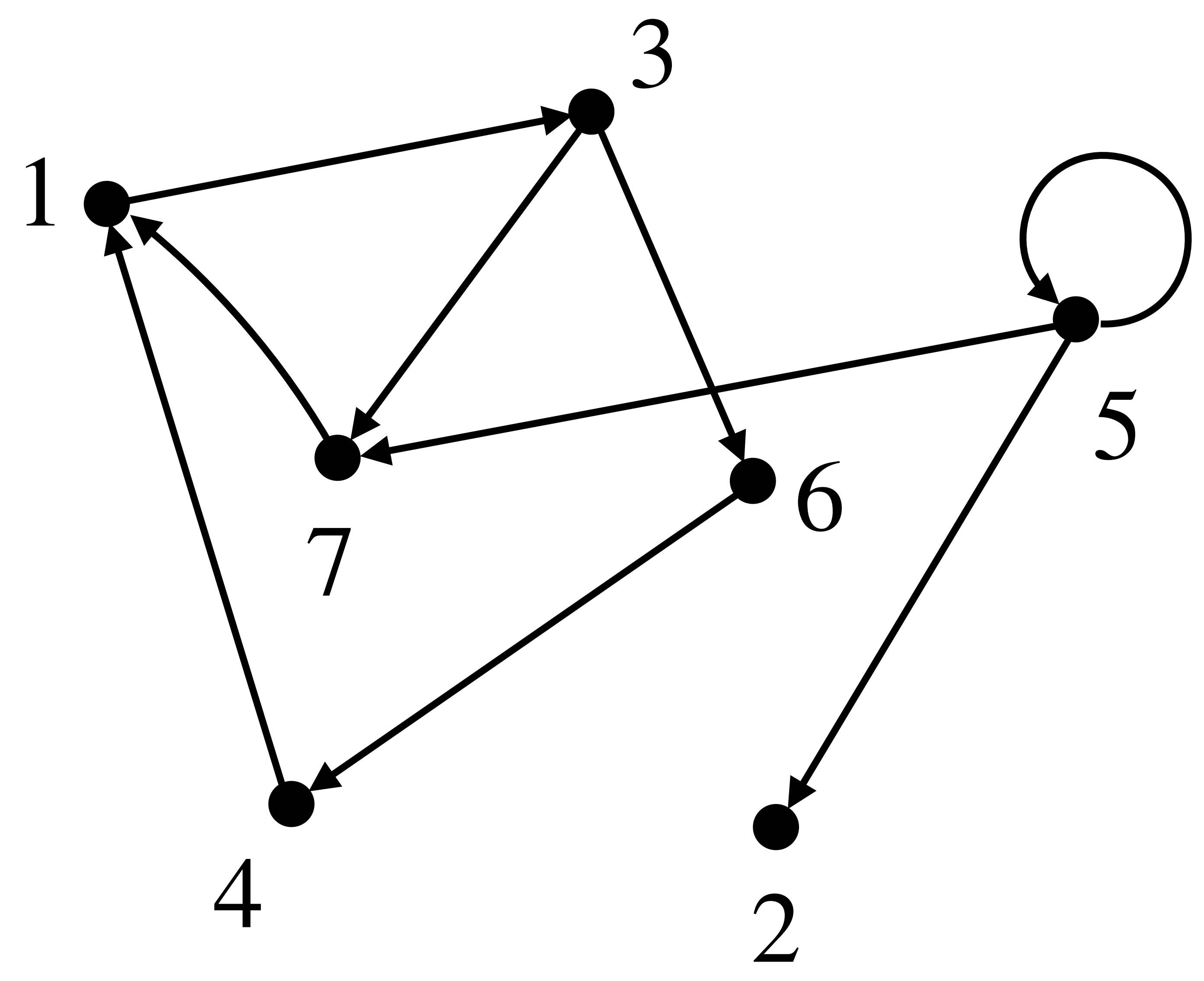
- Two vertices, say u and v, are called adjacent (or neighbours) if (u, v) or (v, u) is an edge.
- Vertices u and v are said to be incident on an edge e, if $e = (u, v)$ or $e = (v, u)$.



- The degree of a vertex is the number of edges it is incident with.
- A path from a vertex u to a vertex v is a sequence of vertices $\langle v_0, v_1, v_2, \ldots, v_k \rangle$ such that $v_0 = u$, $v_k = v$, (v_{i-1}, v_i) is an edge and no vertex is repeated in the sequence.

Graphs: Basic Terminology

- Two vertices, say u and v, are called adjacent (or neighbours) if (u, v) or (v, u) is an edge.
- Vertices u and v are said to be incident on an edge e, if $e = (u, v)$ or $e = (v, u)$.



$\langle 5,7,1,3,6 \rangle$ is a path

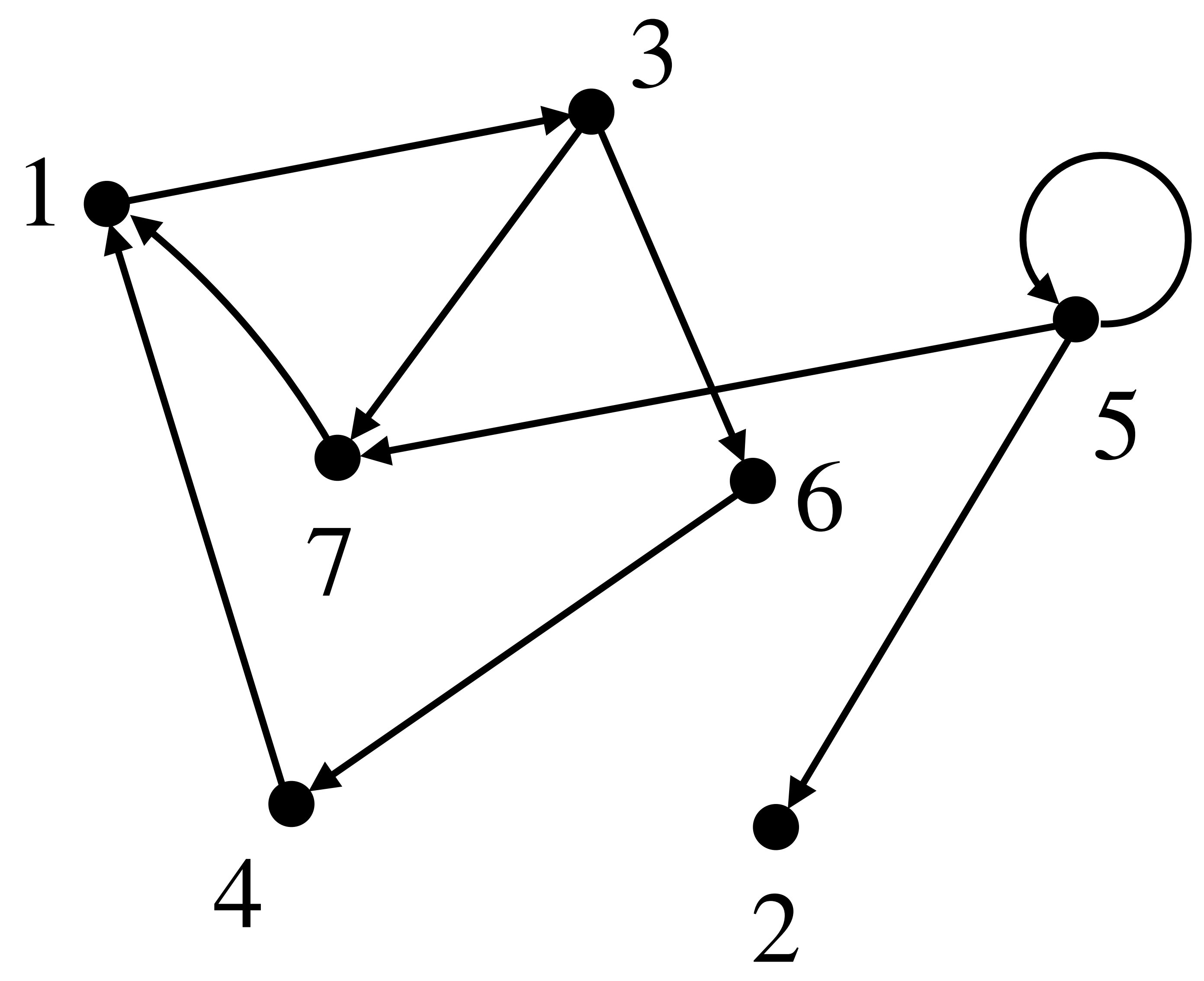
- The degree of a vertex is the number of edges it is incident with.
- A path from a vertex u to a vertex v is a sequence of vertices $\langle v_0, v_1, v_2, \ldots, v_k \rangle$ such that $v_0 = u$, $v_k = v$, (v_{i-1}, v_i) is an edge and no vertex is repeated in the sequence.

Graphs: Basic Terminology

- The length of a path is the number of edges in it.

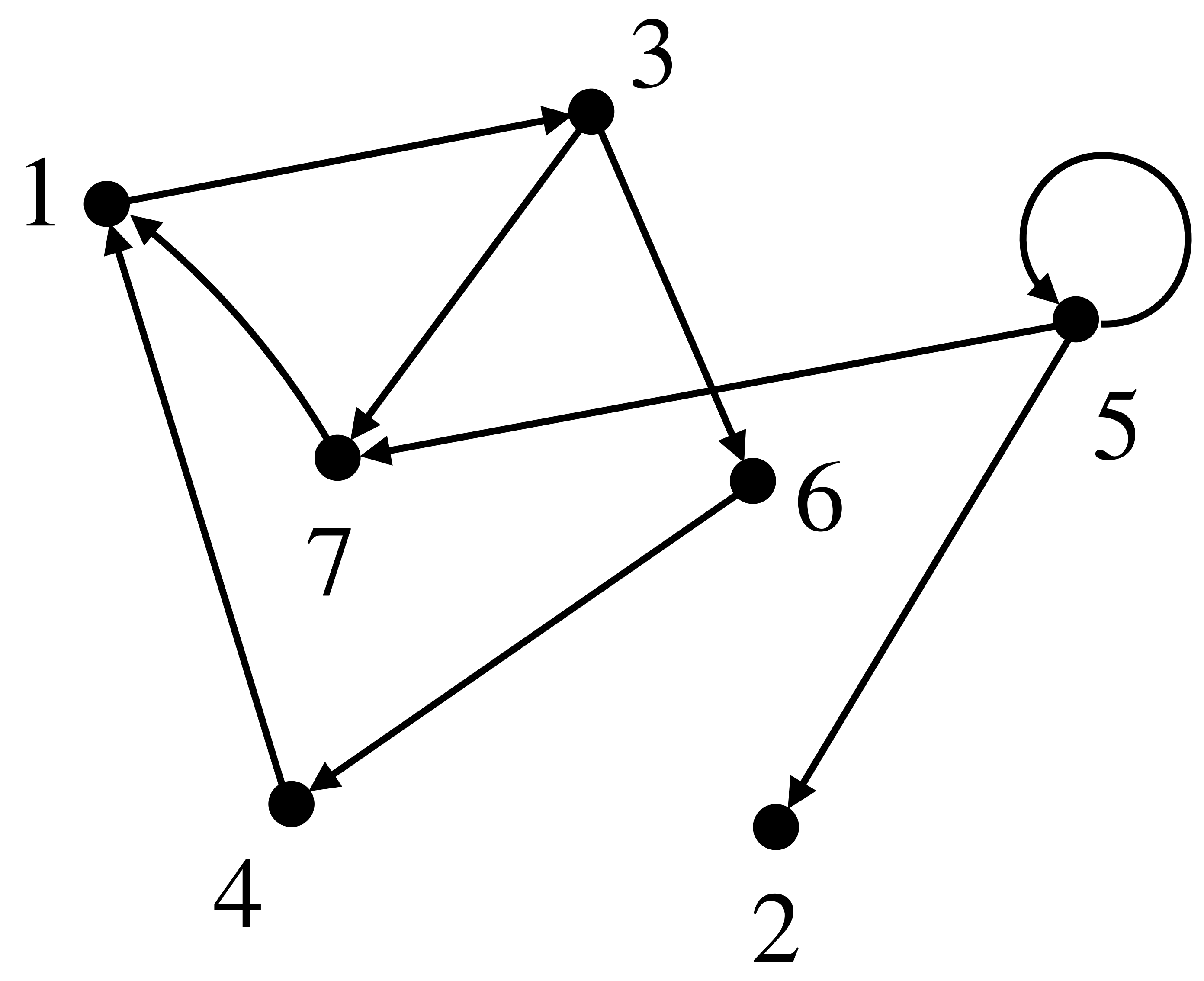
Graphs: Basic Terminology

- The length of a path is the number of edges in it.



Graphs: Basic Terminology

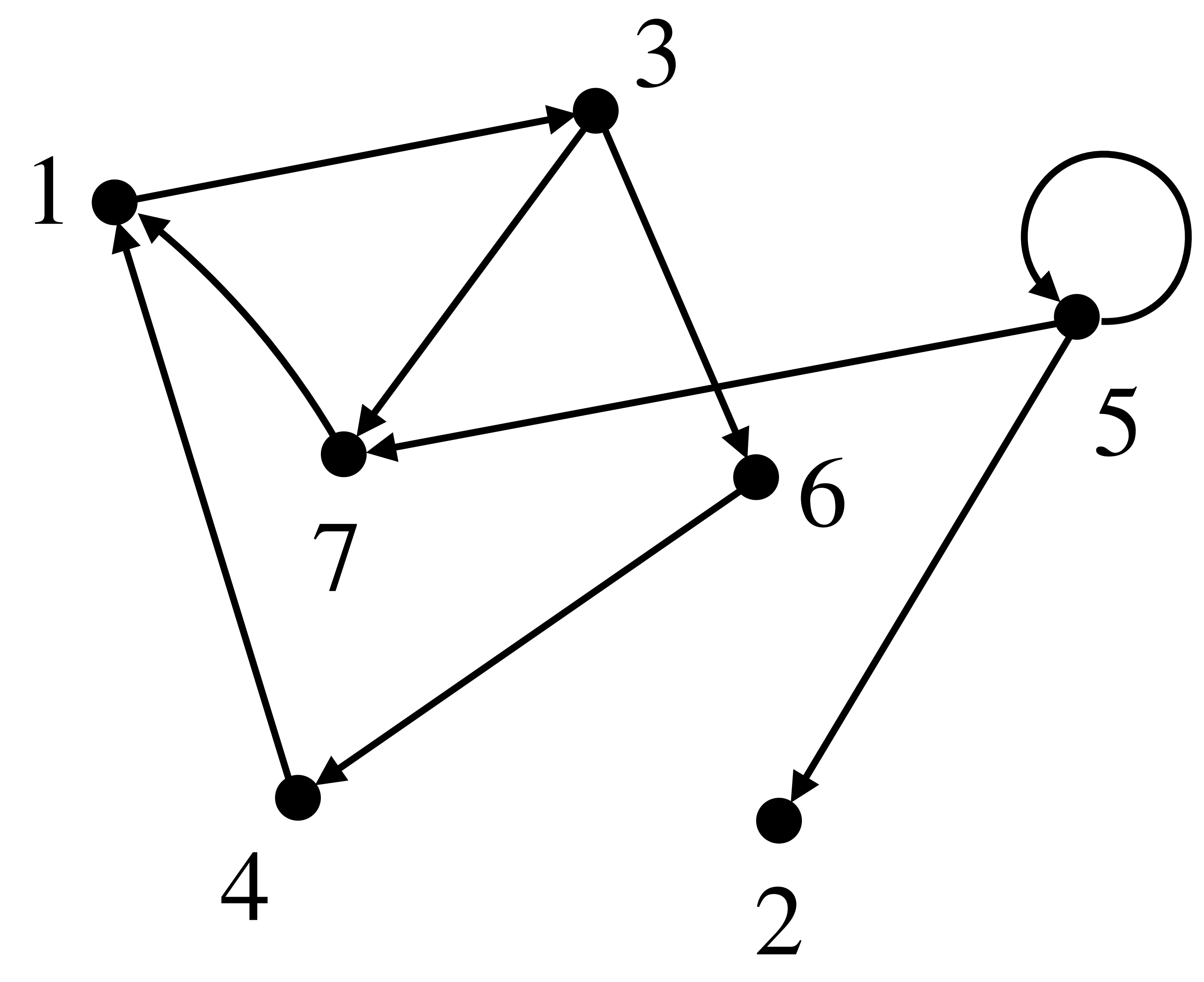
- The length of a path is the number of edges in it.



$\langle 5,7,1,3,6 \rangle$ is a path of length 4

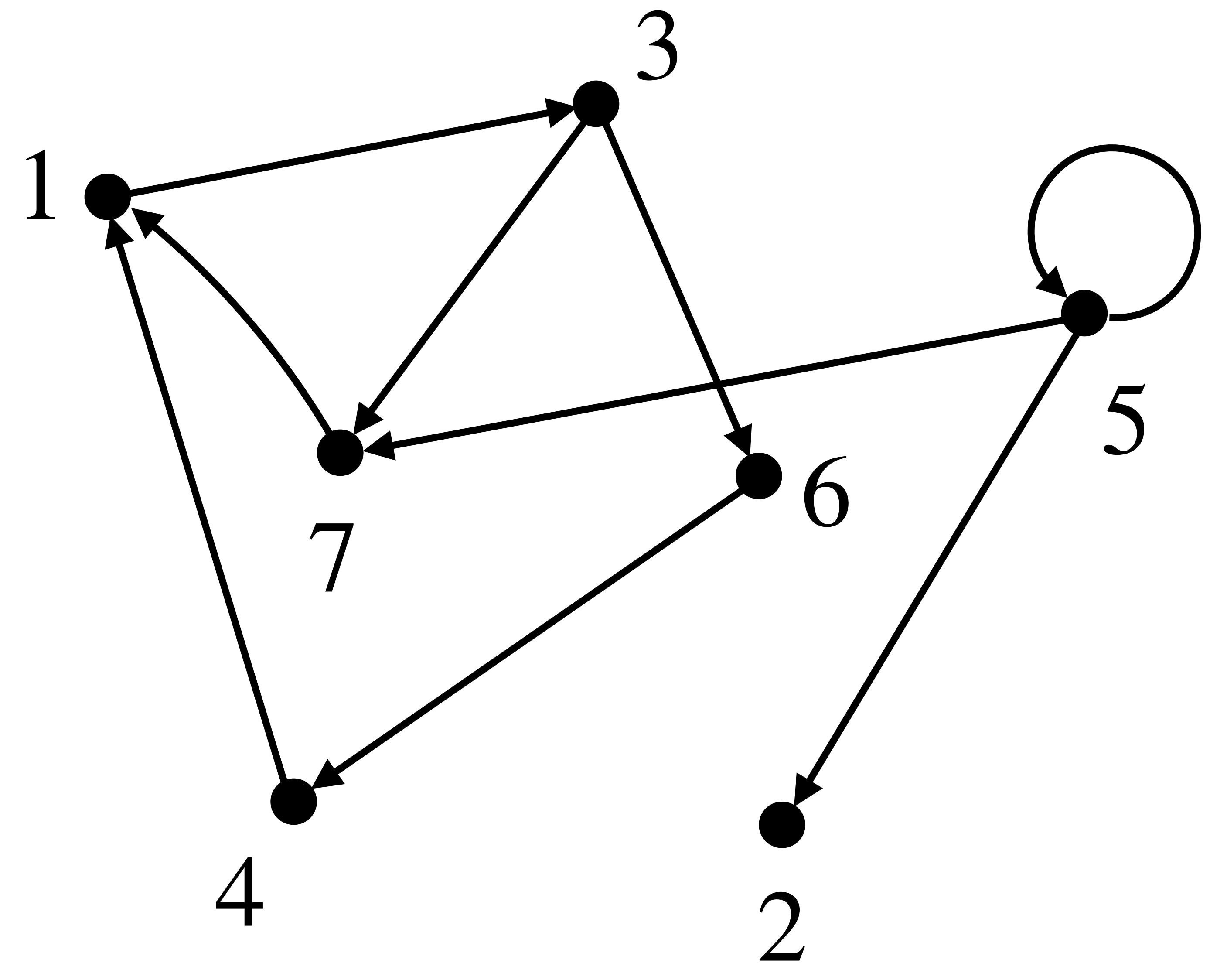
Graphs: Basic Terminology

- The length of a path is the number of edges in it. The distance from a vertex i to a vertex j is the length of the shortest path from i to j.



Graphs: Basic Terminology

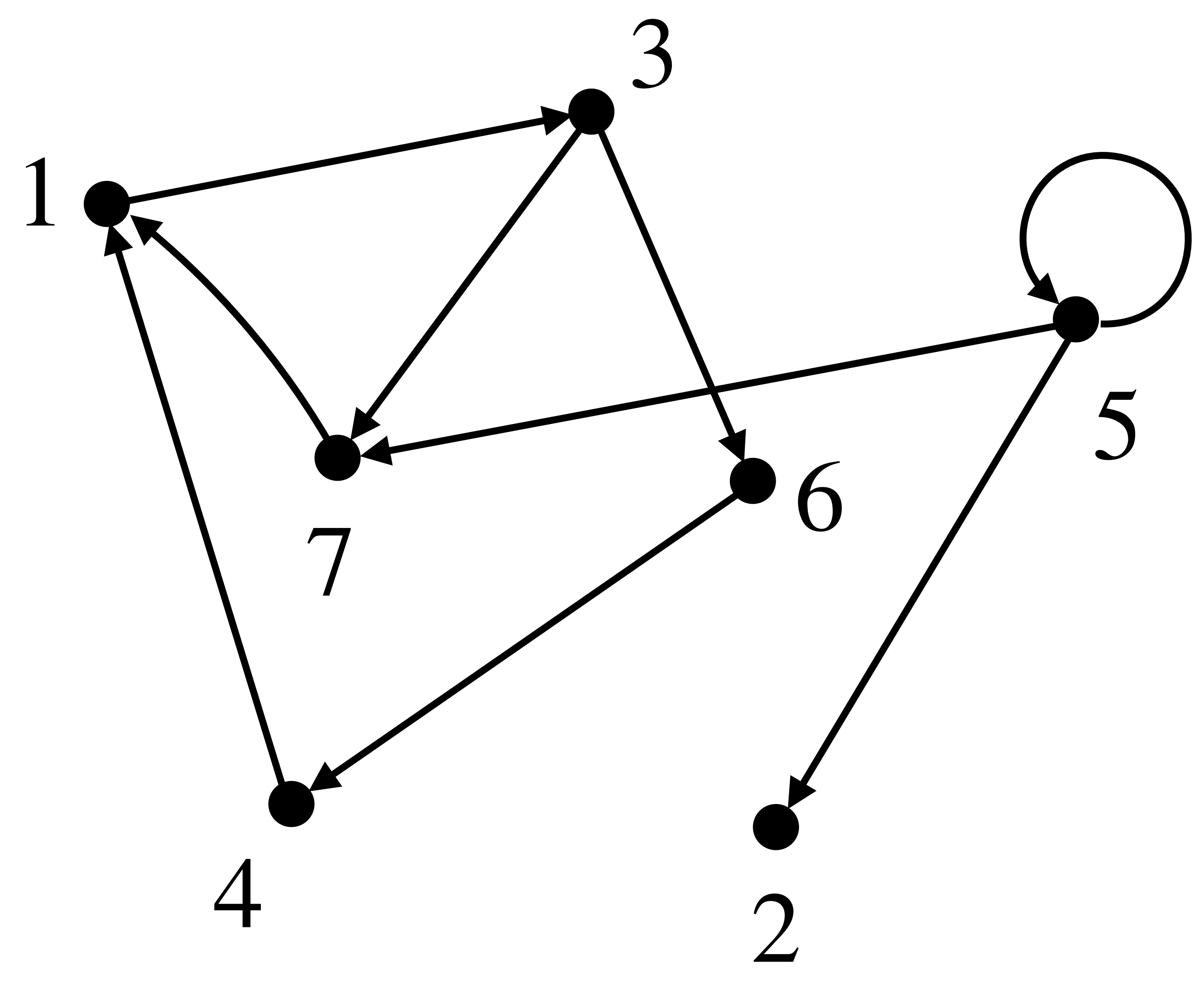
- The length of a path is the number of edges in it. The distance from a vertex i to a vertex j is the length of the shortest path from i to j.



Distance from 6 to 1 is 2

Graphs: Basic Terminology

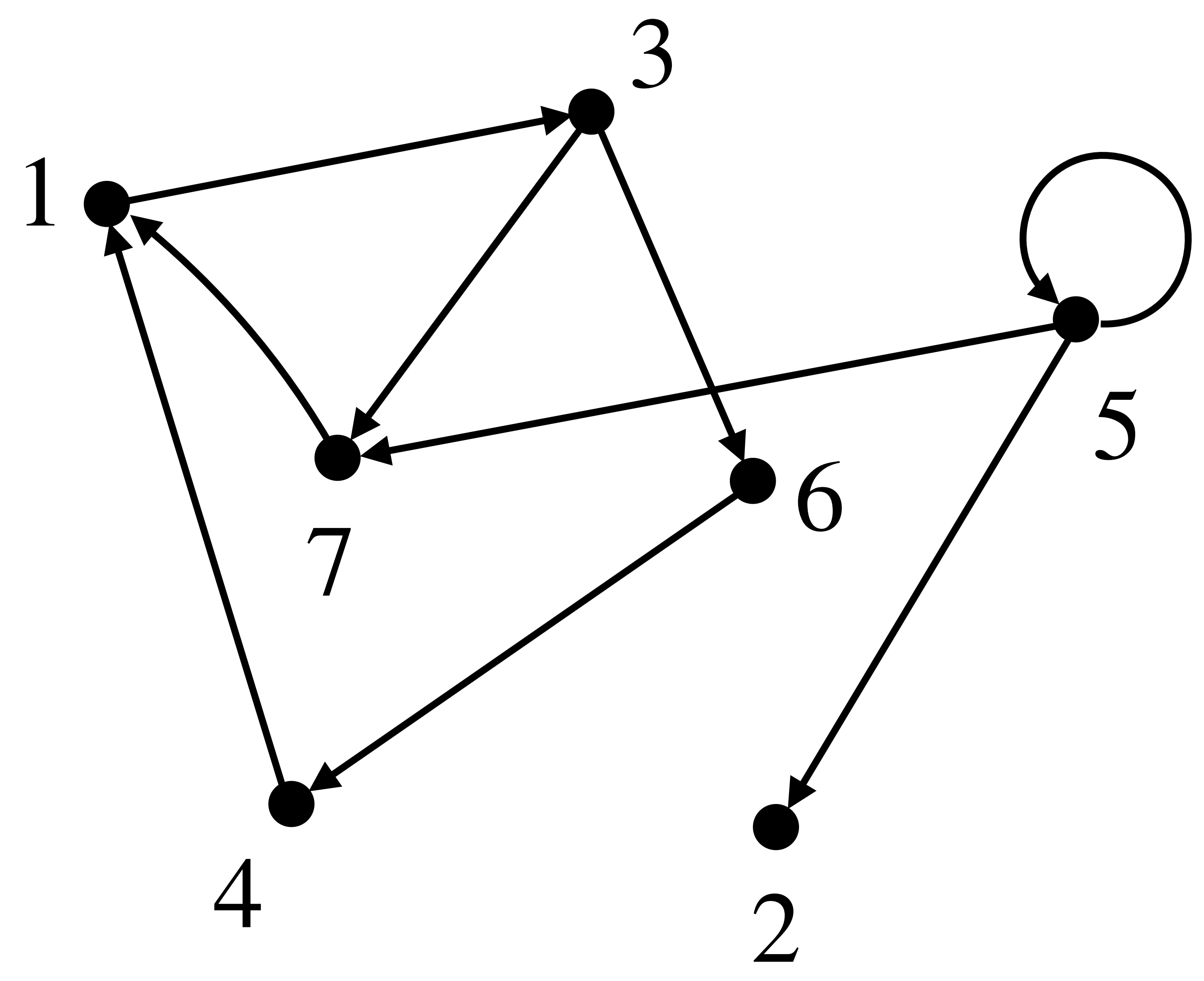
- The length of a path is the number of edges in it. The distance from a vertex i to a vertex j is the length of the shortest path from i to j.



- A cycle is a sequence of vertices $\langle v_0, v_1, v_2, \ldots, v_k \rangle$ such that $v_0 = v_k$, (v_{i-1}, v_i) is an edge and

Graphs: Basic Terminology

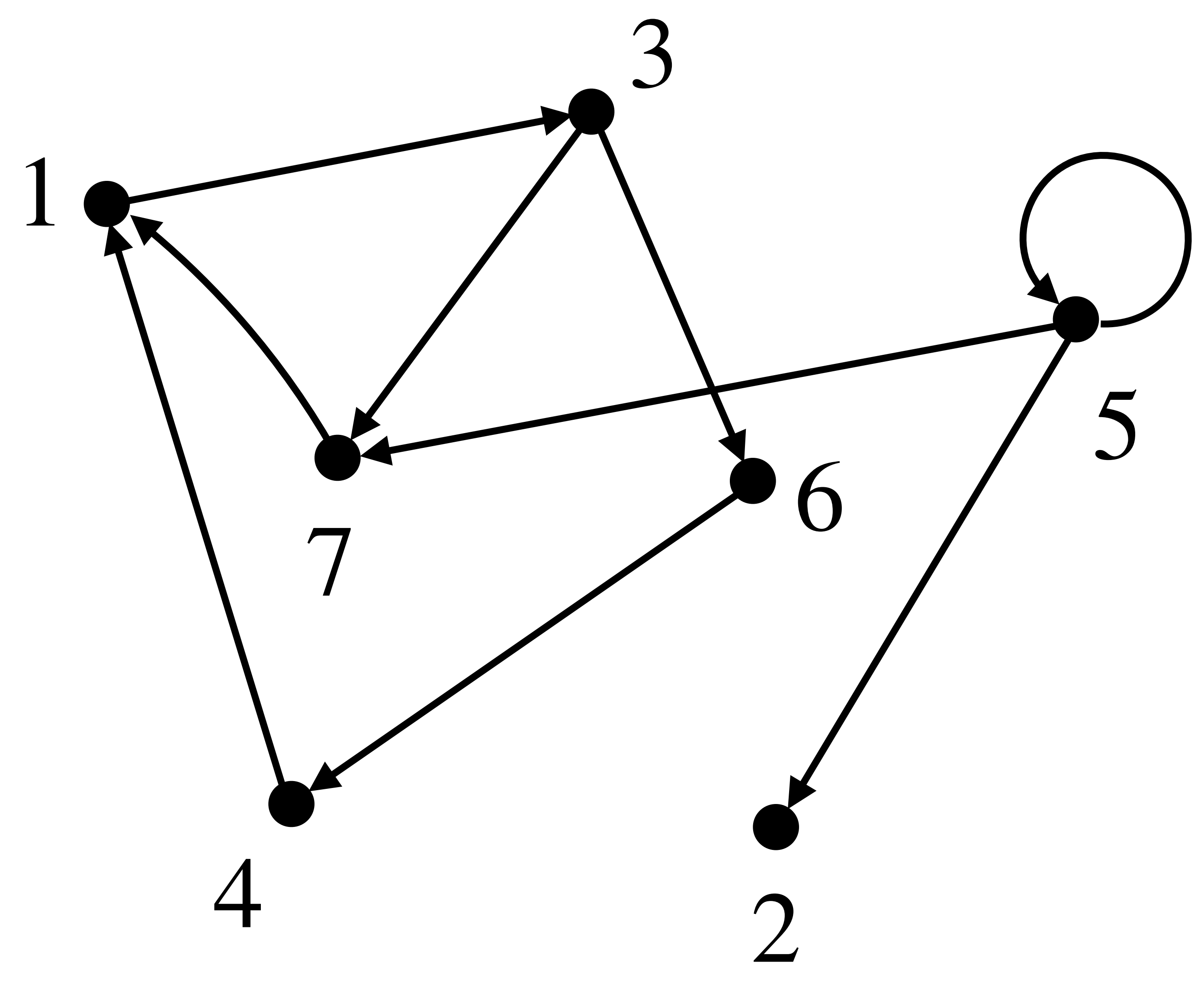
- The length of a path is the number of edges in it. The distance from a vertex i to a vertex j is the length of the shortest path from i to j.



- A cycle is a sequence of vertices $\langle v_0, v_1, v_2, \ldots, v_k \rangle$ such that $v_0 = v_k$, (v_{i-1}, v_i) is an edge and no vertex apart from v_0 and v_k is repeated in the sequence.

Graphs: Basic Terminology

- The length of a path is the number of edges in it. The distance from a vertex i to a vertex j is the length of the shortest path from i to j.



$\langle 4,1,3,6,4 \rangle$ is a cycle

- A cycle is a sequence of vertices $\langle v_0, v_1, v_2, \ldots, v_k \rangle$ such that $v_0 = v_k$, (v_{i-1}, v_i) is an edge and no vertex apart from v_0 and v_k is repeated in the sequence.

Graphs: Representation

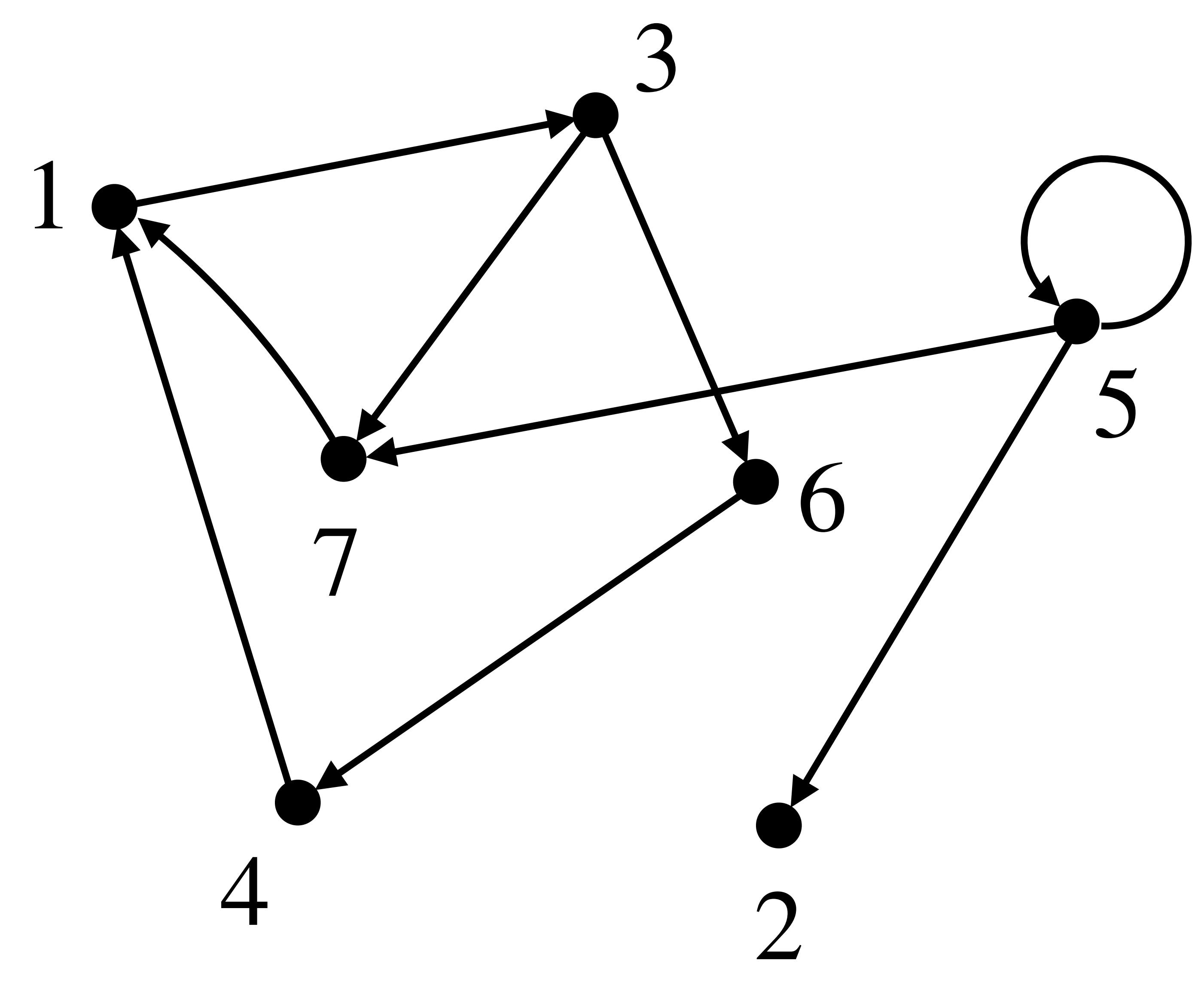
Graphs: Representation

Adjacency-list representation:

Graphs: Representation

Adjacency-list representation:

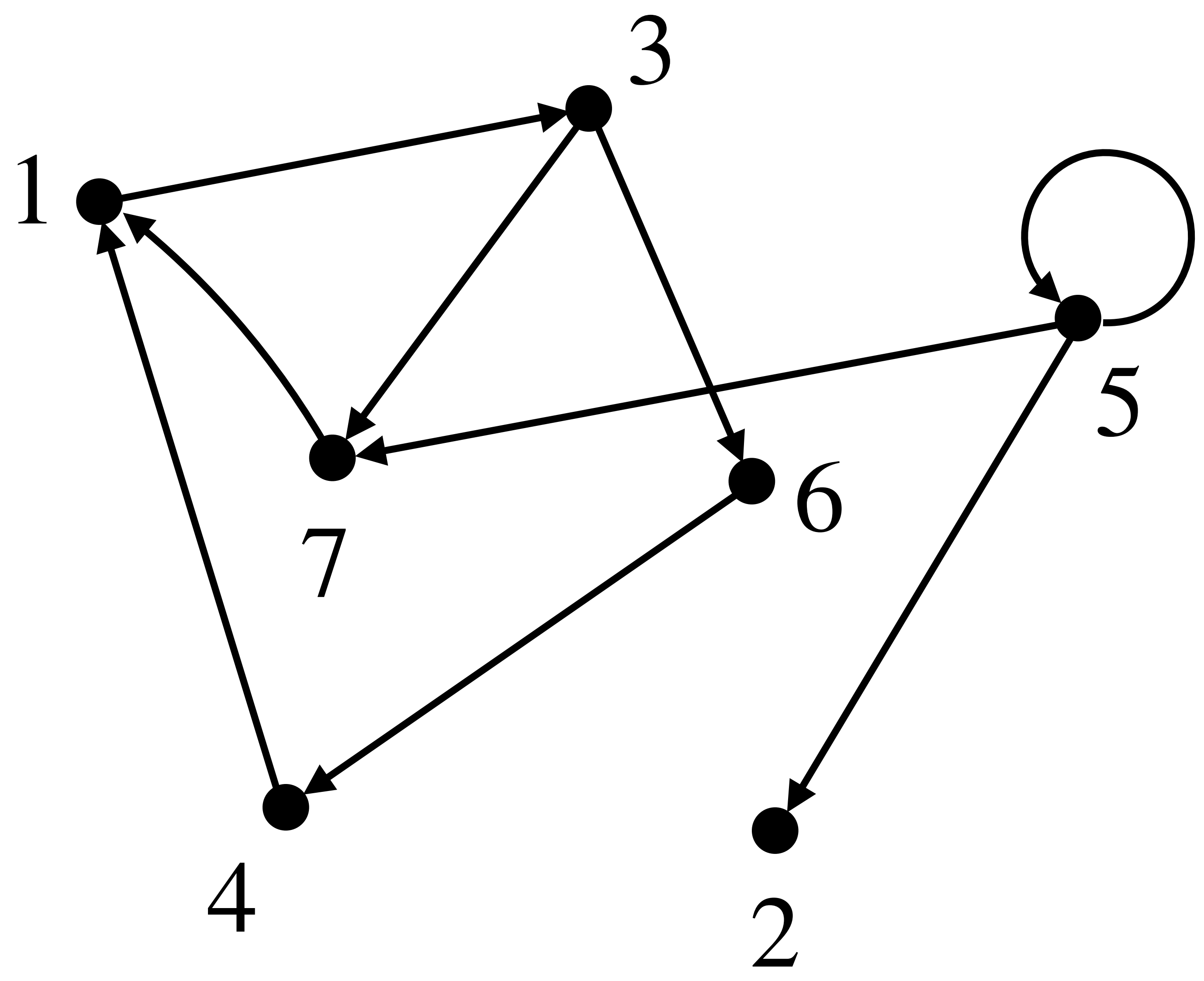
Graph G



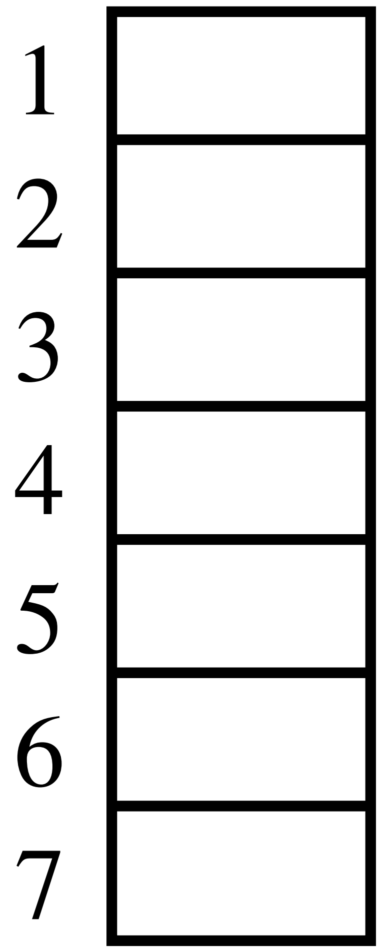
Graphs: Representation

Adjacency-list representation:

Graph G



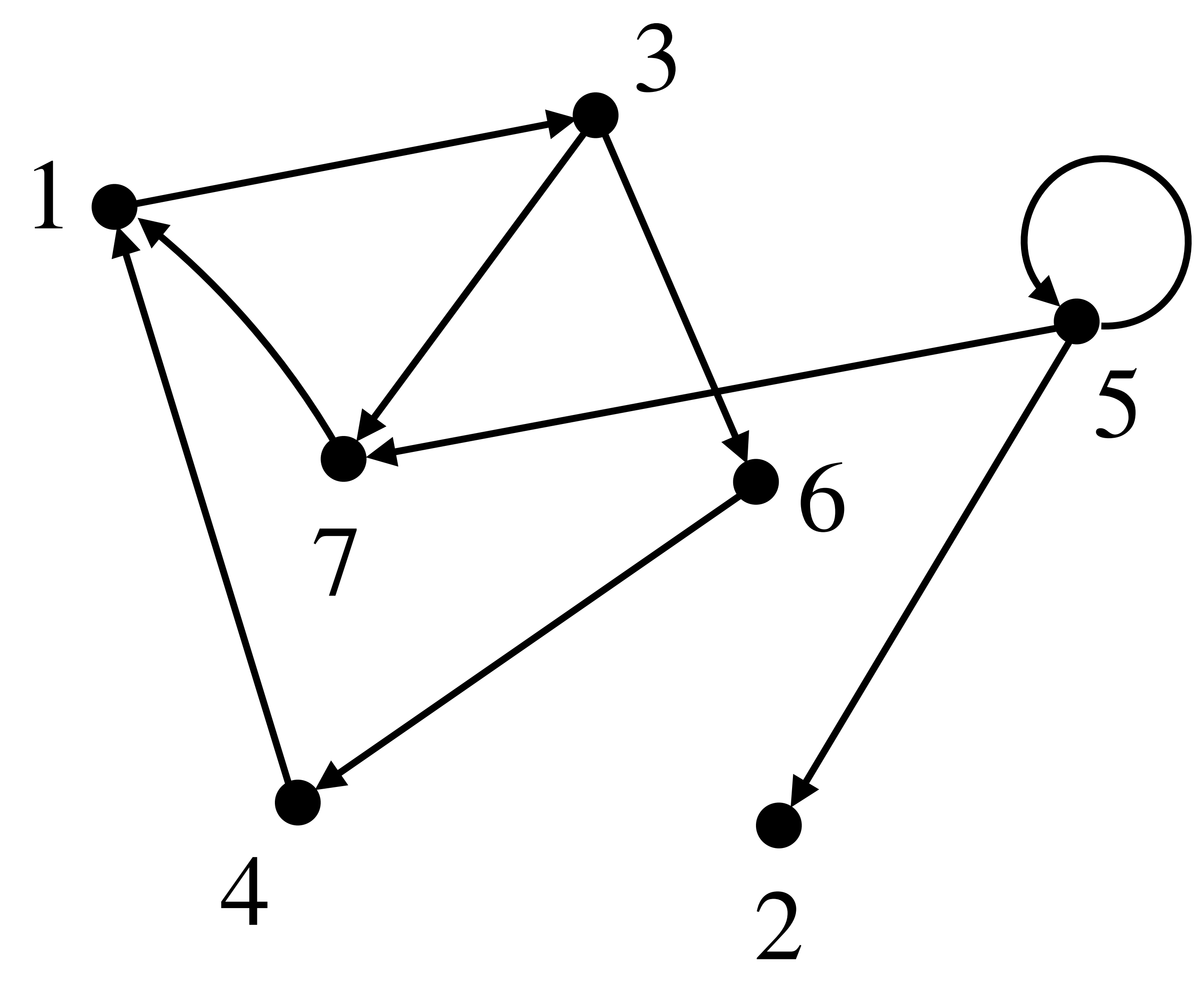
G's adjacency list



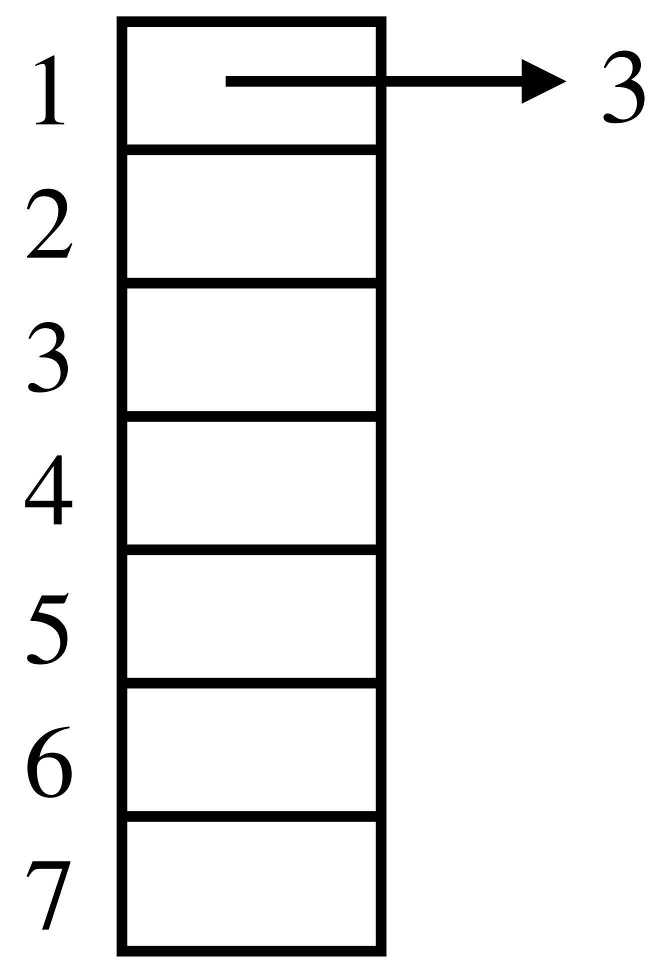
Graphs: Representation

Adjacency-list representation:

Graph G



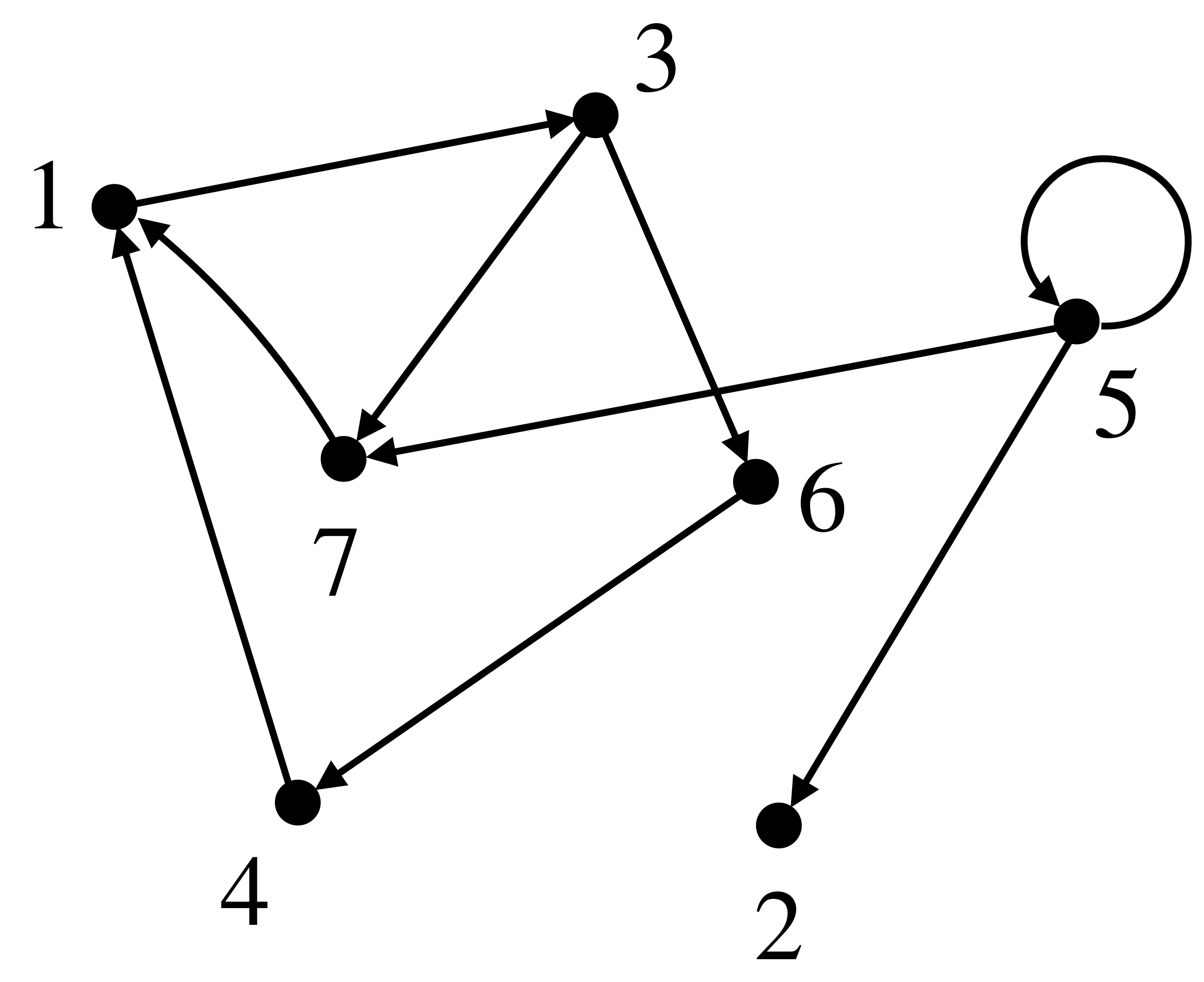
G's adjacency list



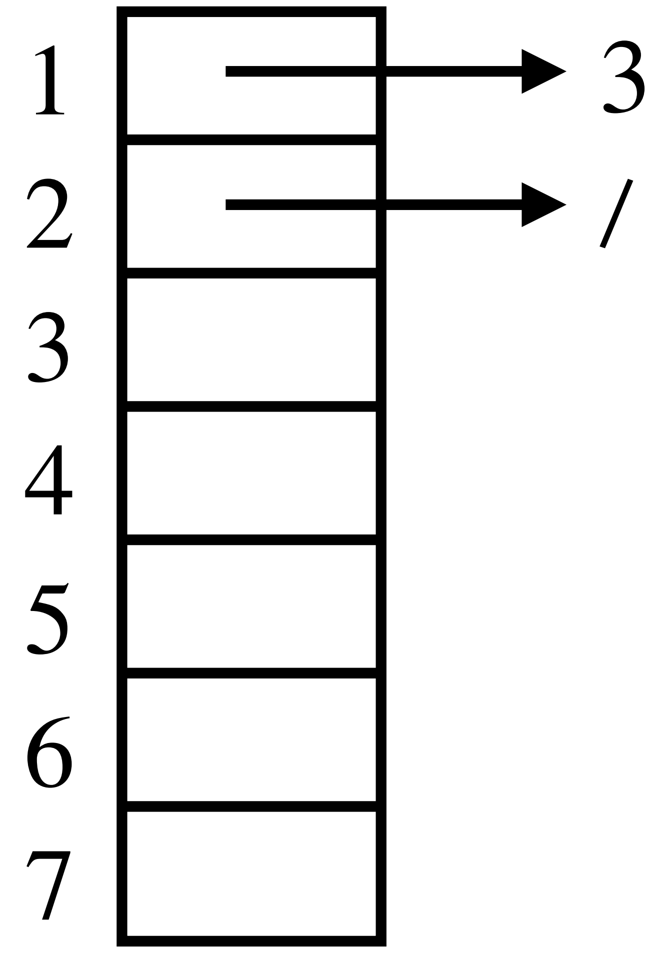
Graphs: Representation

Adjacency-list representation:

Graph G



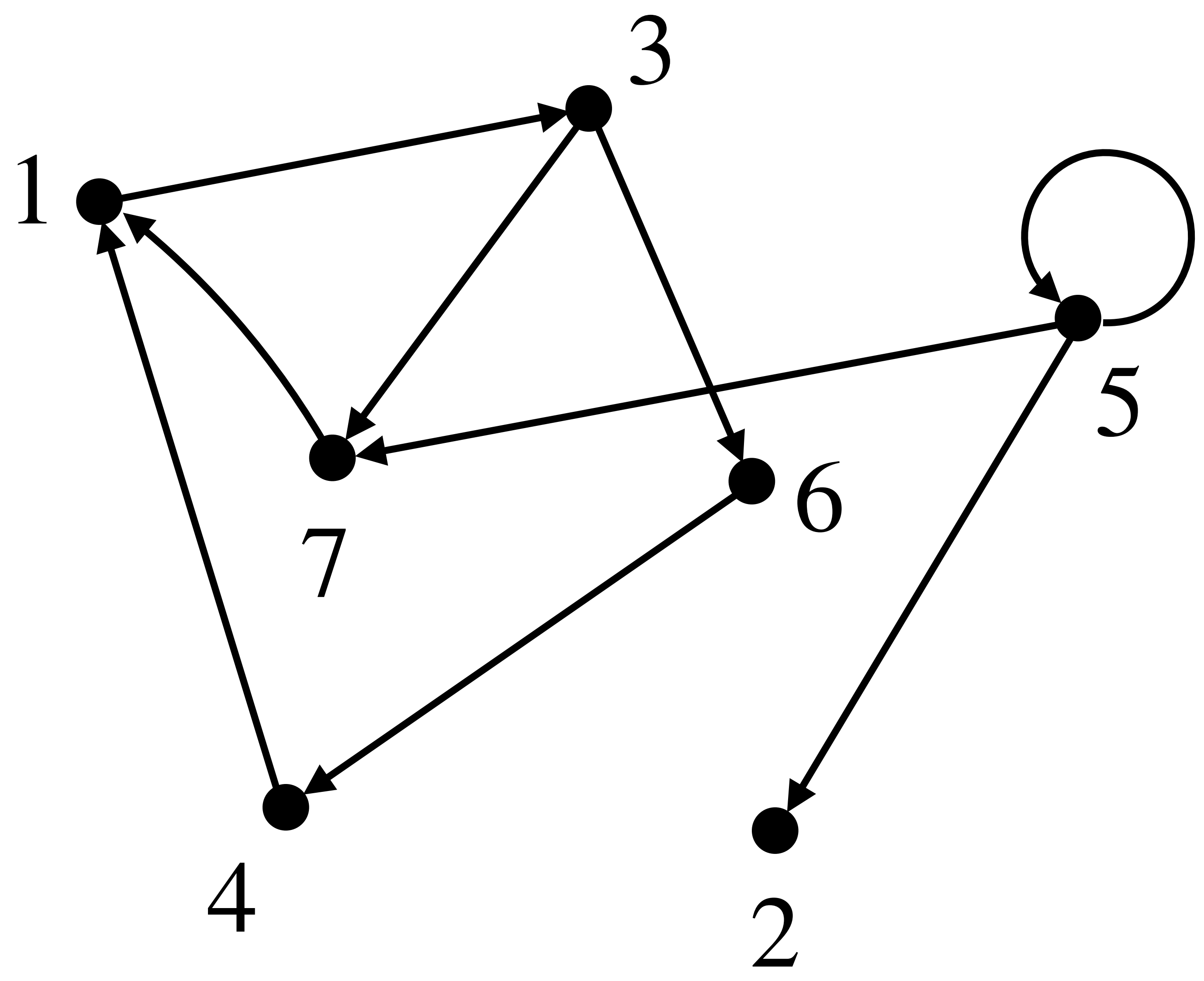
G's adjacency list



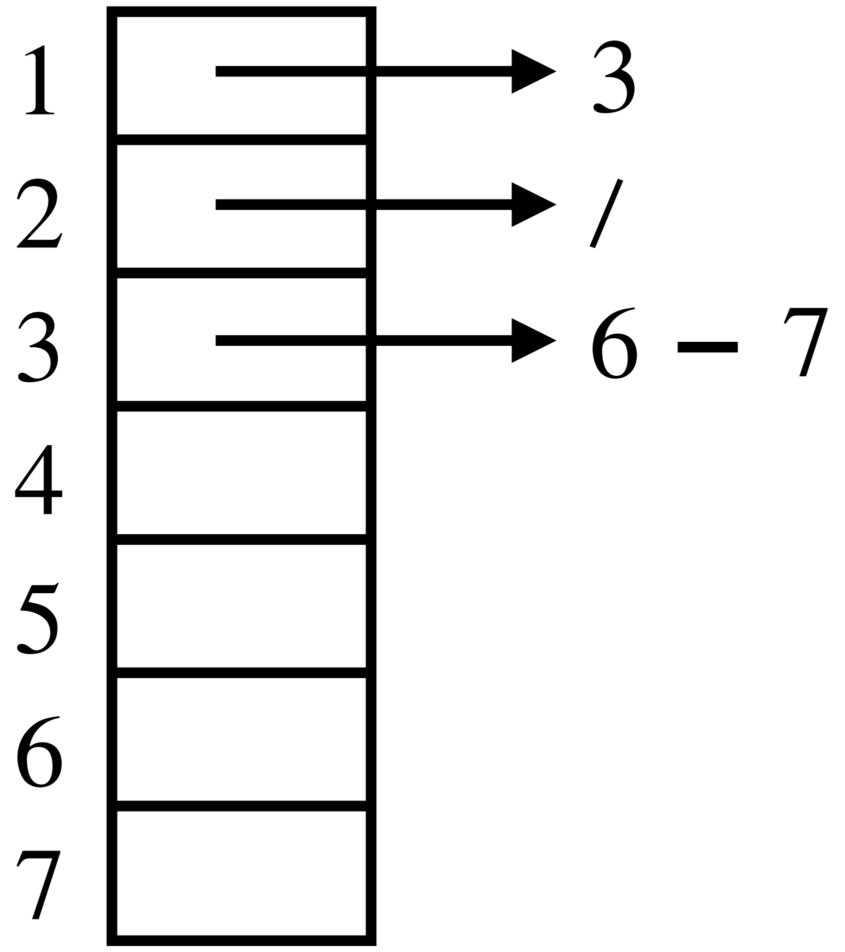
Graphs: Representation

Adjacency-list representation:

Graph G



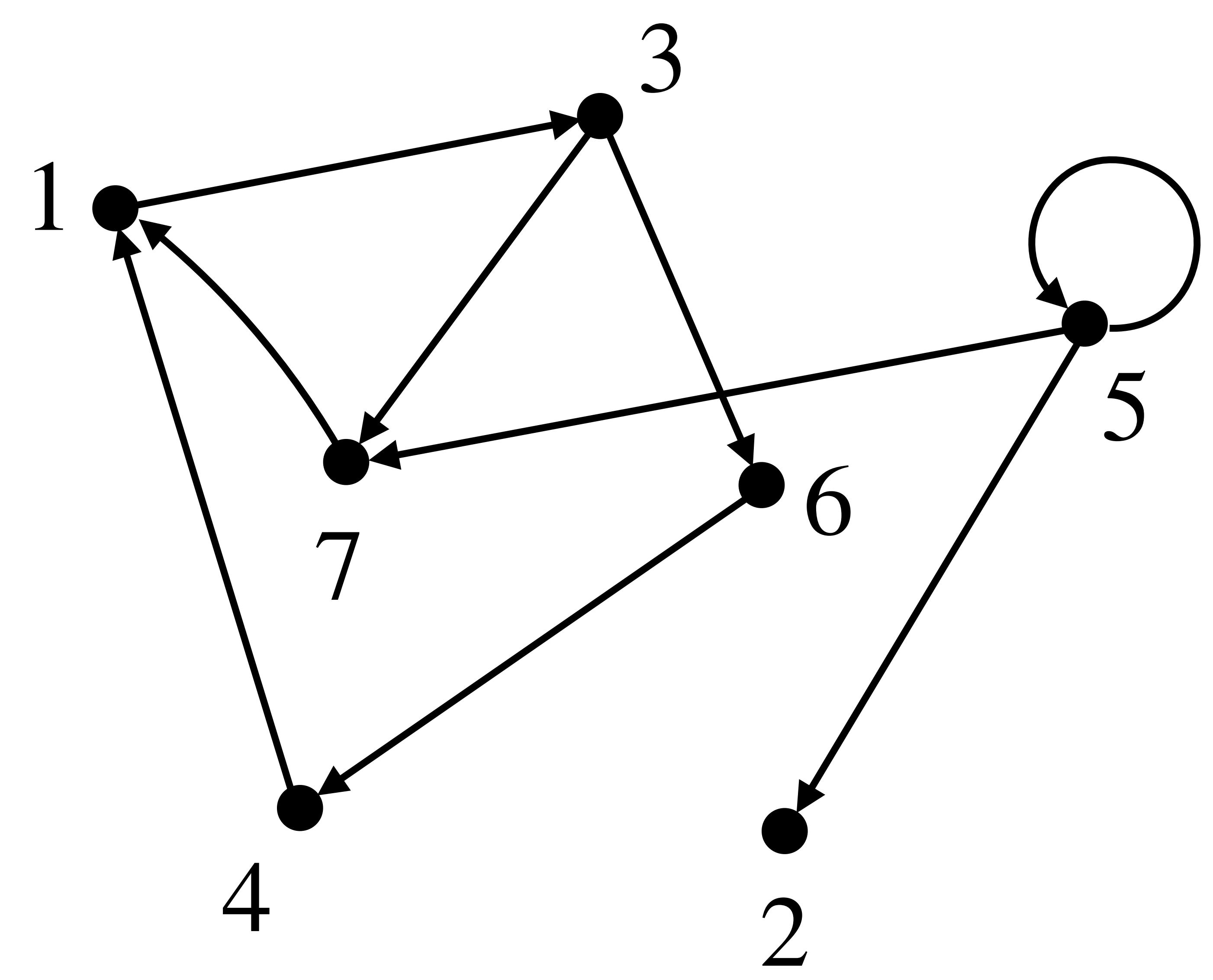
G's adjacency list



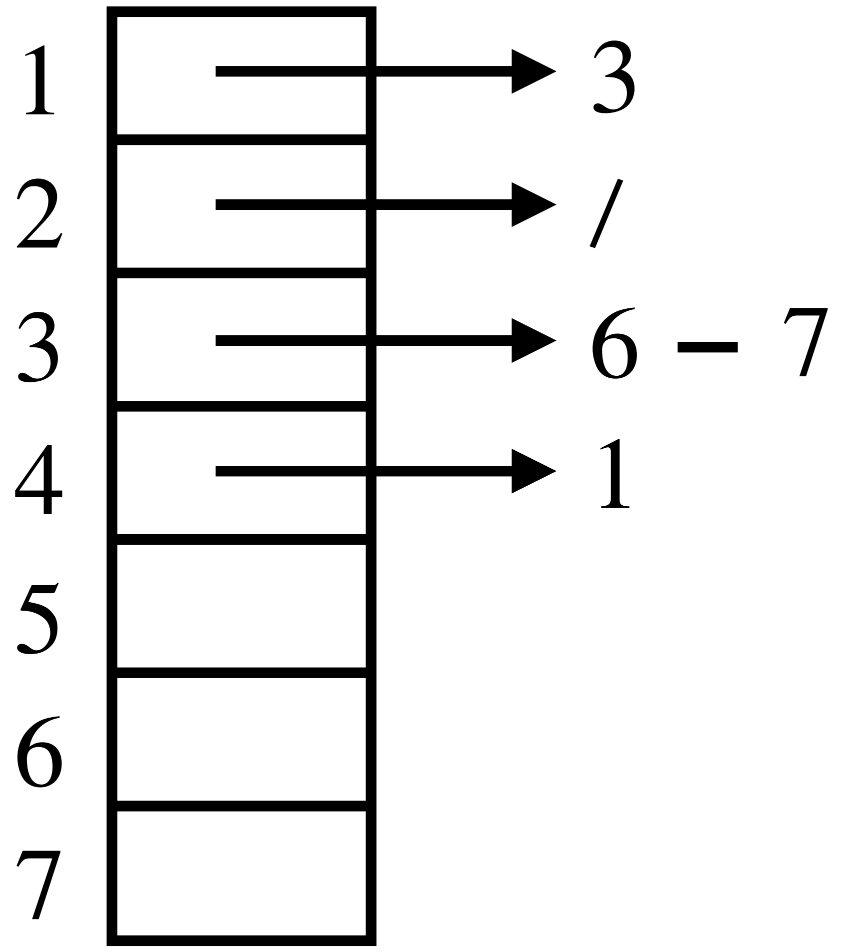
Graphs: Representation

Adjacency-list representation:

Graph G



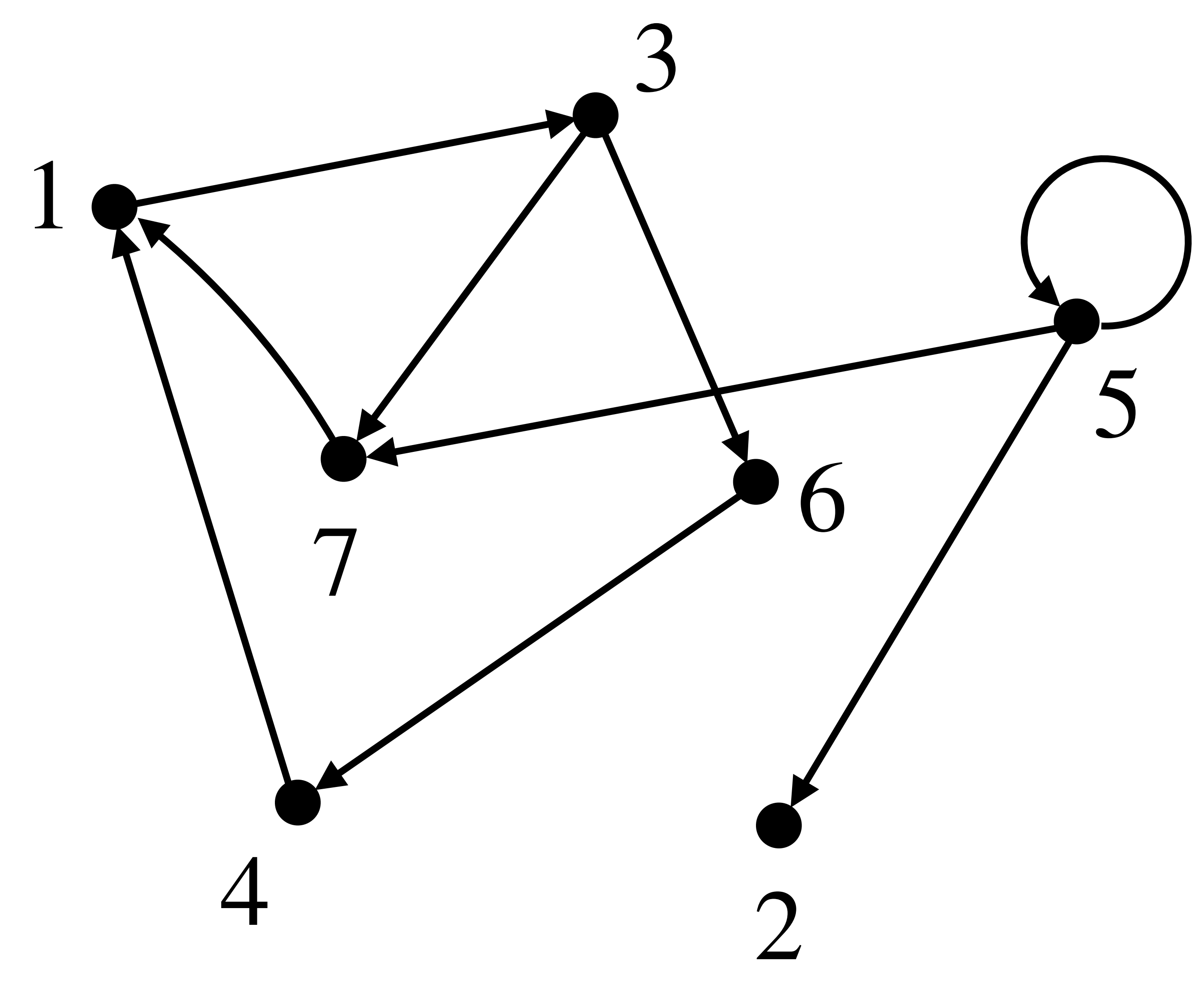
G's adjacency list



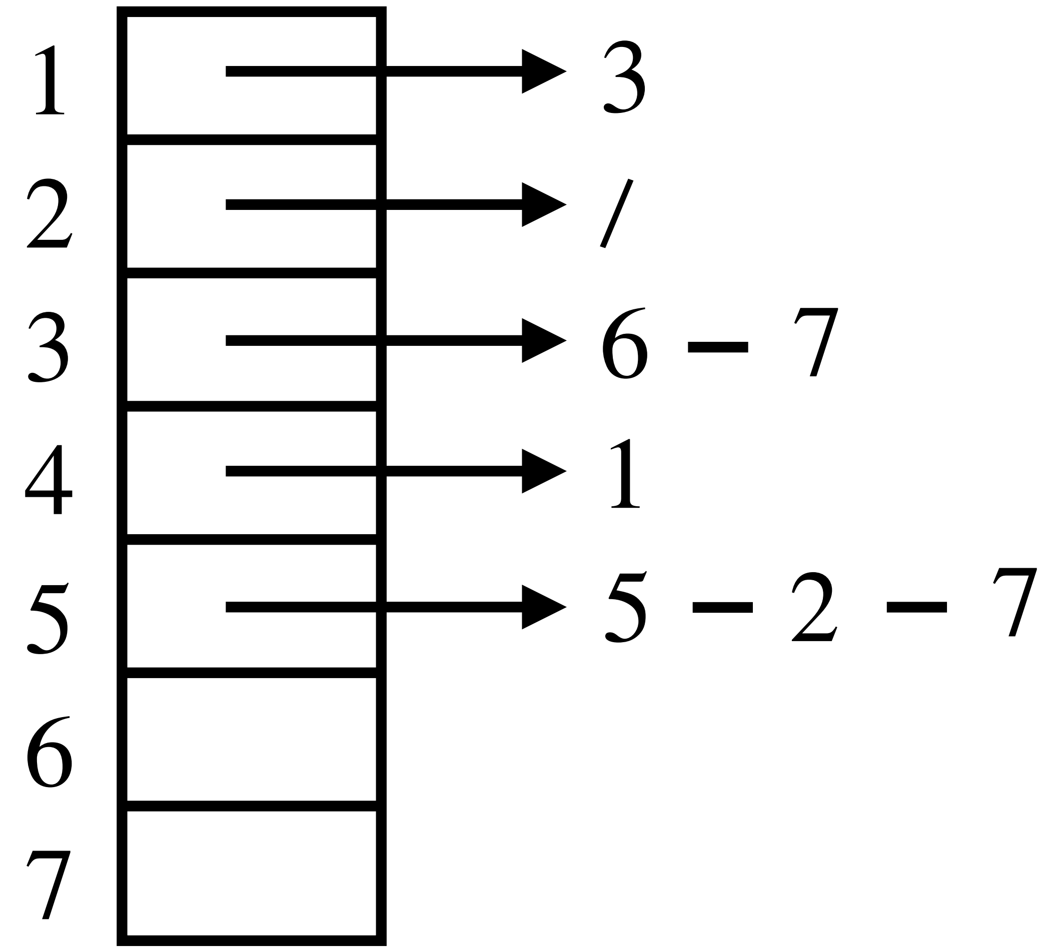
Graphs: Representation

Adjacency-list representation:

Graph G



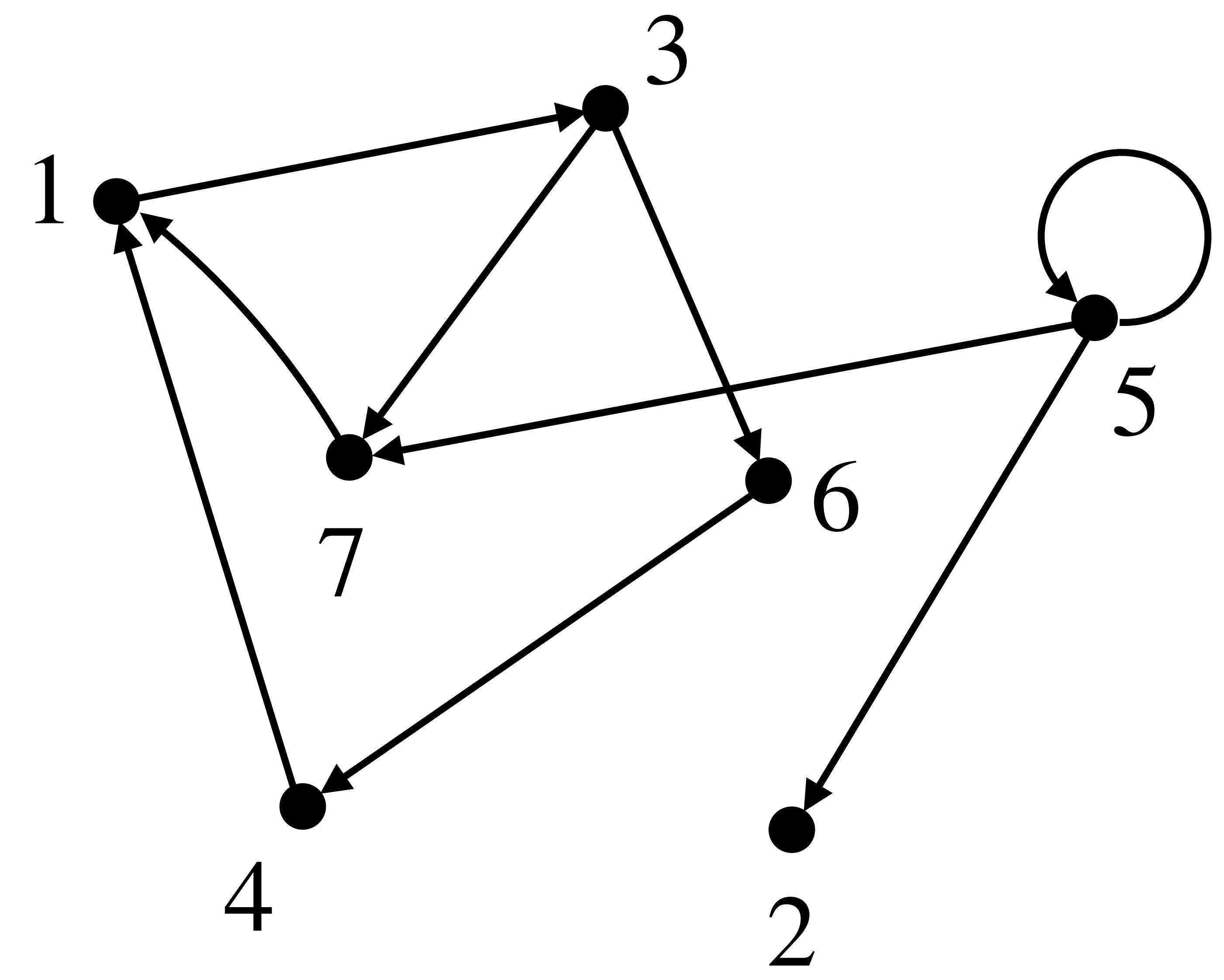
G's adjacency list



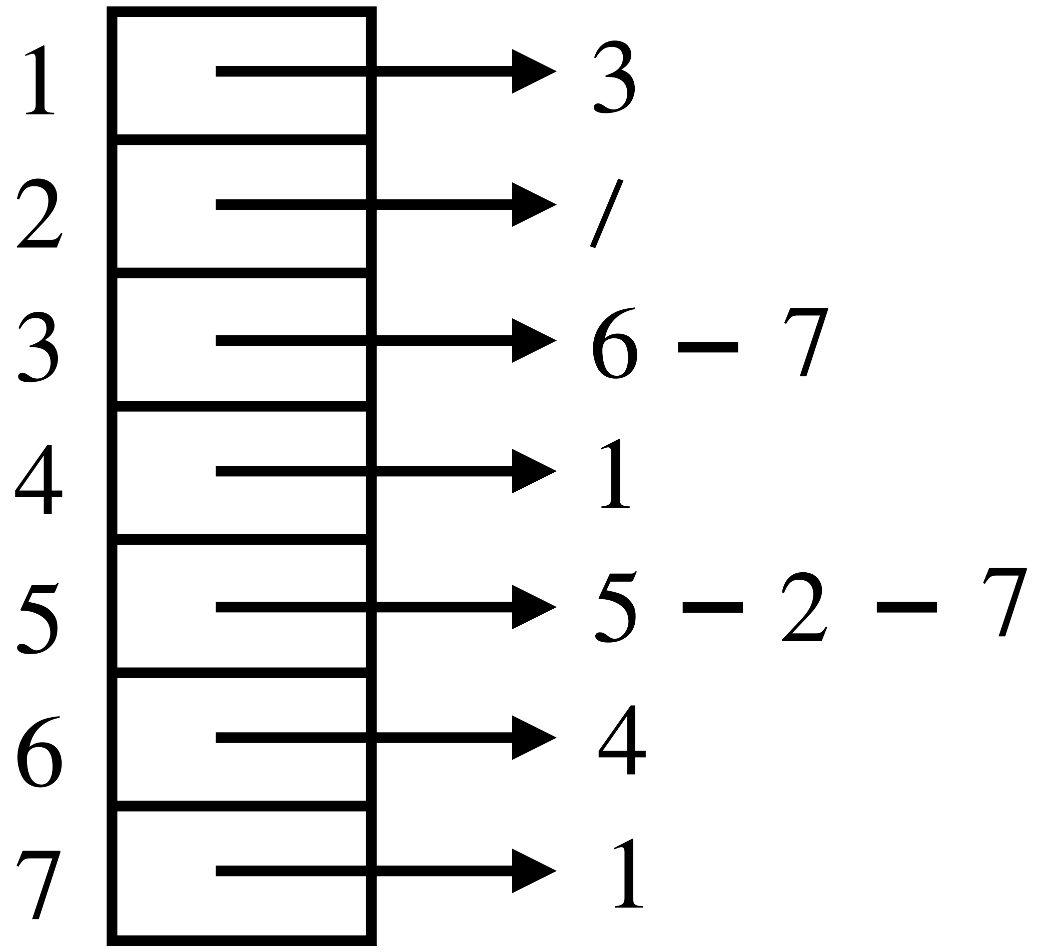
Graphs: Representation

Adjacency-list representation:

Graph G



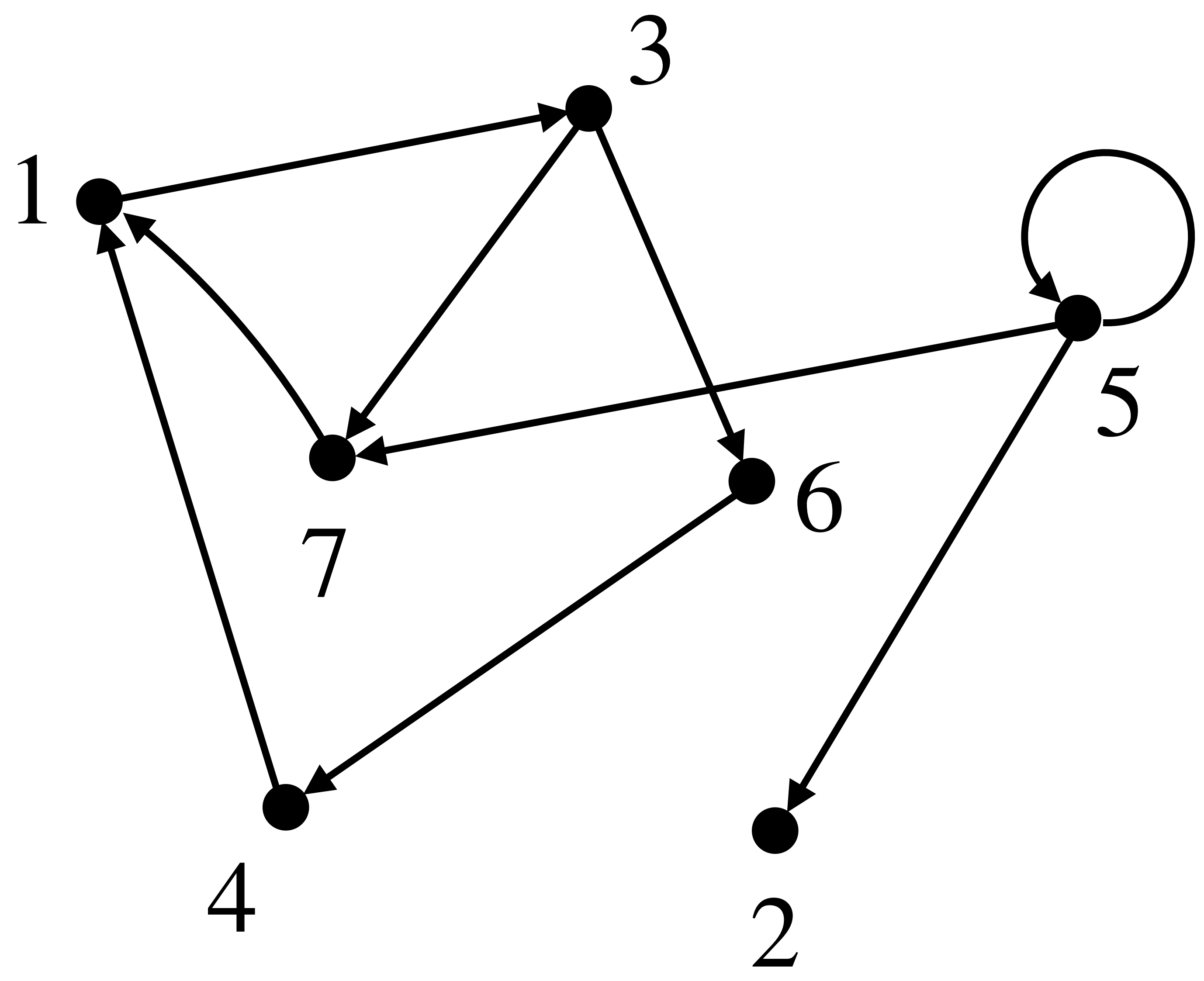
G's adjacency list



Graphs: Representation

Adjacency-list representation:

Graph G



G's adjacency list

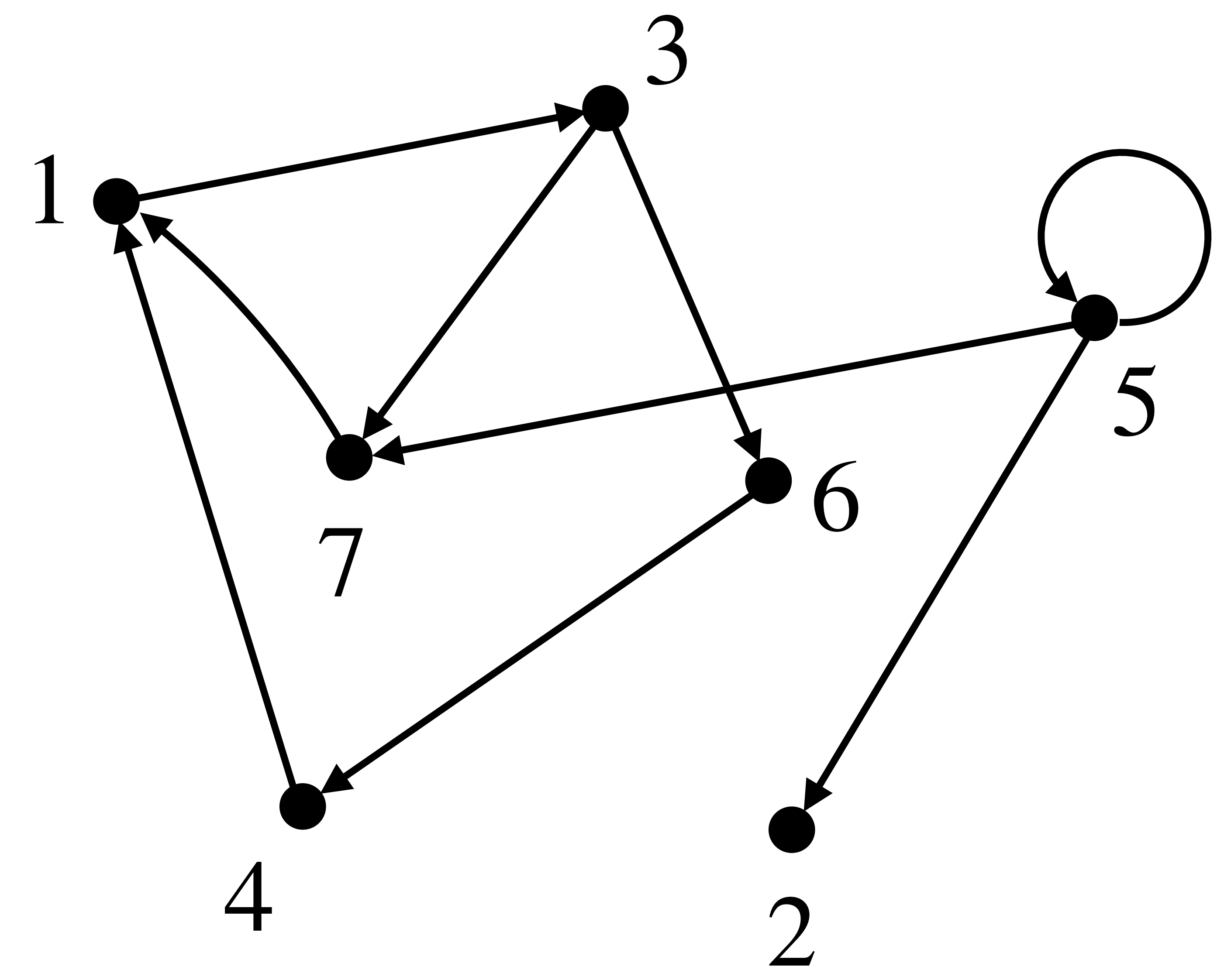
1	→ 3
2	→ /
3	→ 6 − 7
4	→ 1
5	→ 5 − 2 − 7
6	→ 4
7	→ 1

The adjacency-list representation of a graph $G = (V, E)$ consists of an array Adj of $|V|$ lists,

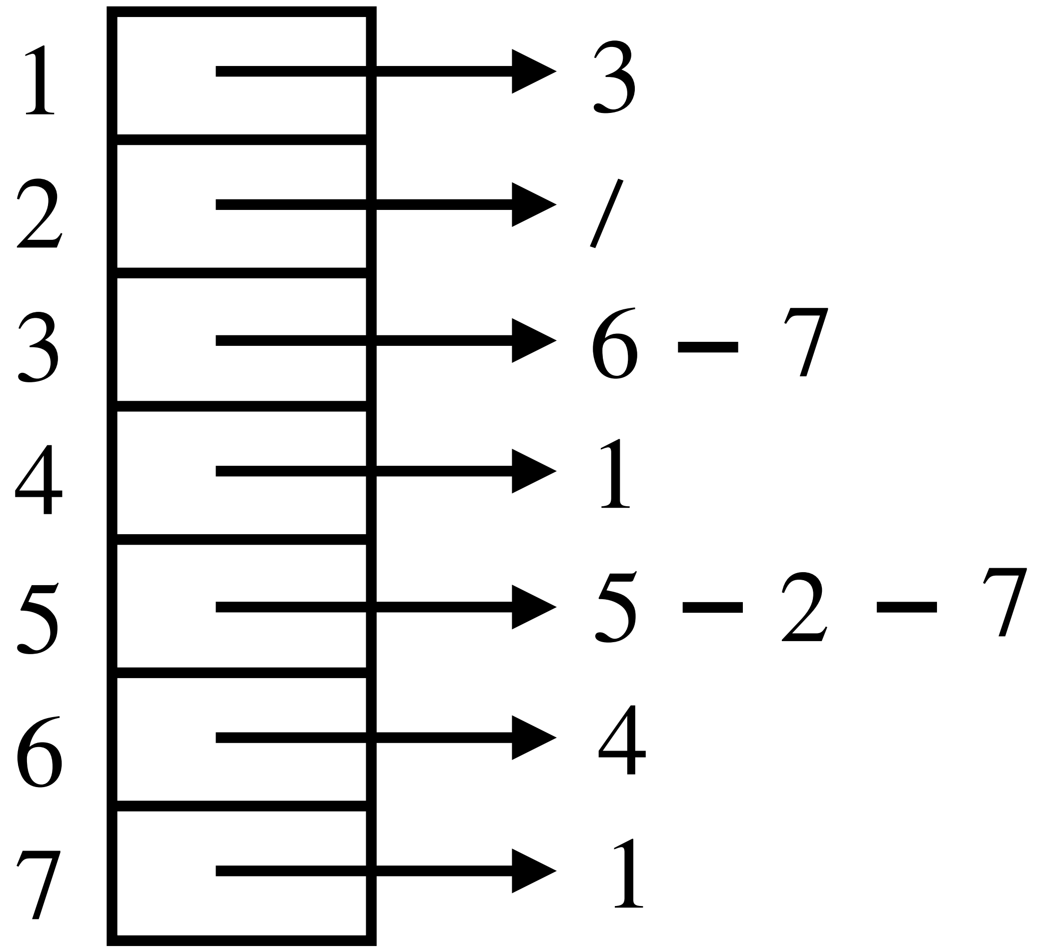
Graphs: Representation

Adjacency-list representation:

Graph G



G's adjacency list

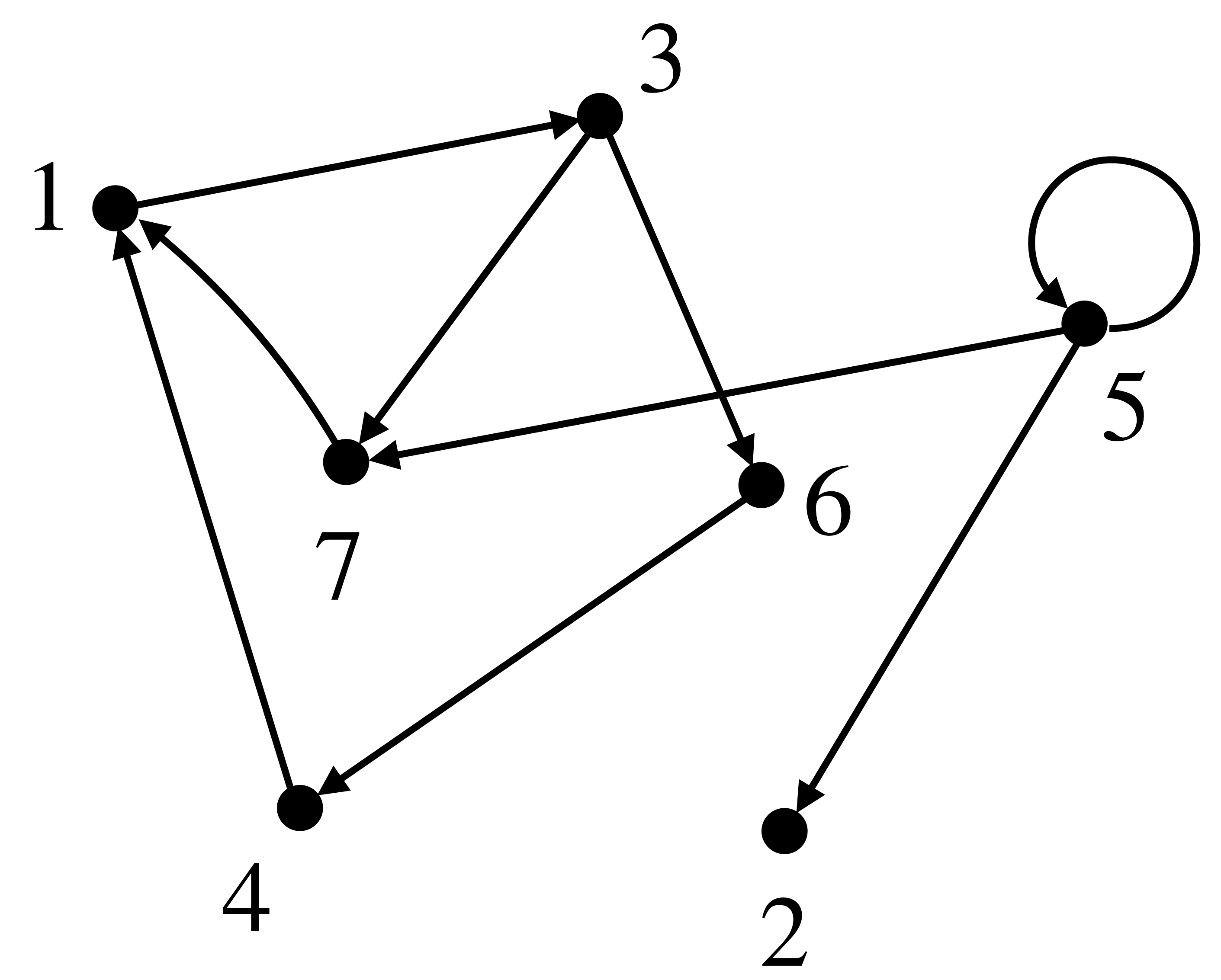


The adjacency-list representation of a graph $G = (V, E)$ consists of an array Adj of $|V|$ lists, one for each vertex in V.

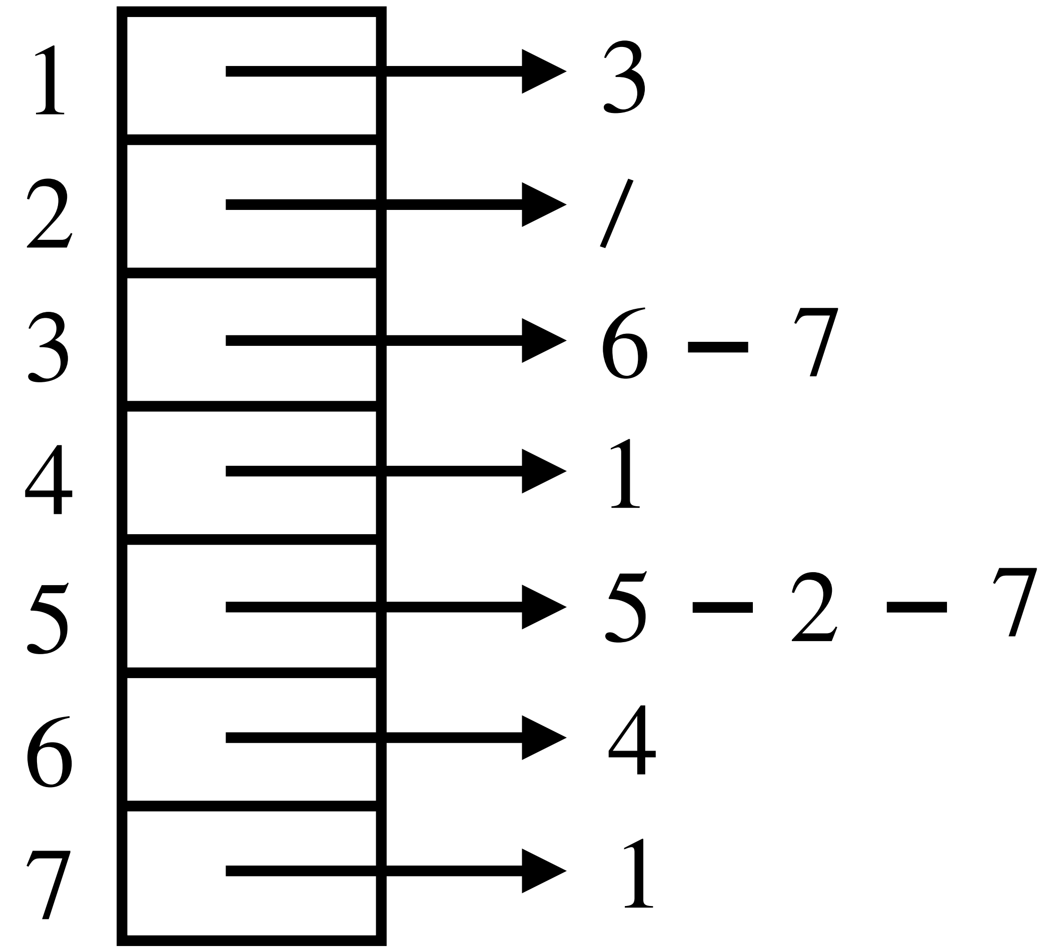
Graphs: Representation

Adjacency-list representation:

Graph G



G's adjacency list

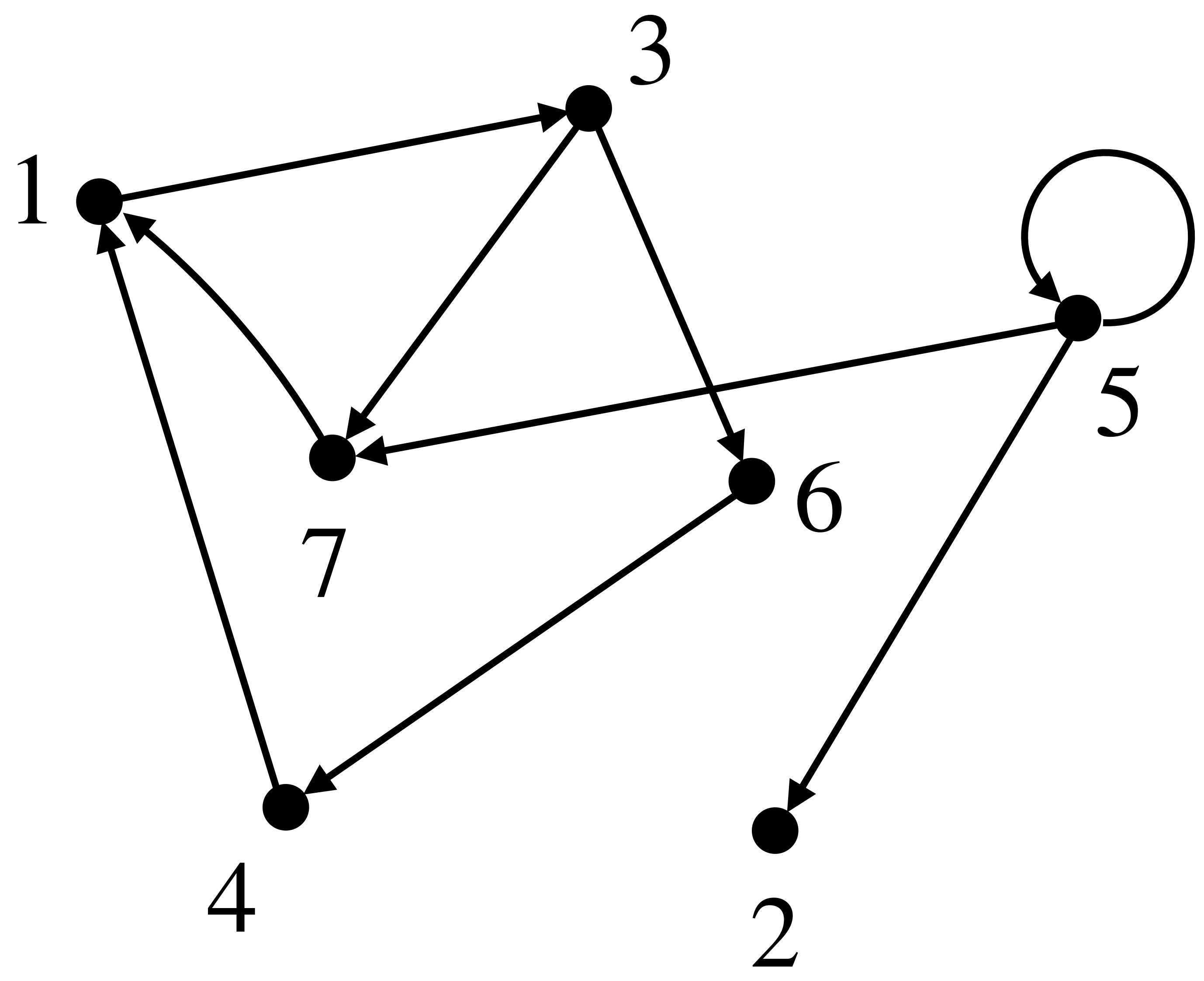


The adjacency-list representation of a graph $G = (V, E)$ consists of an array Adj of $|V|$ lists, one for each vertex in V. For each $u \in V$, the adjacency-list $Adj[u]$ contains all the vertices v

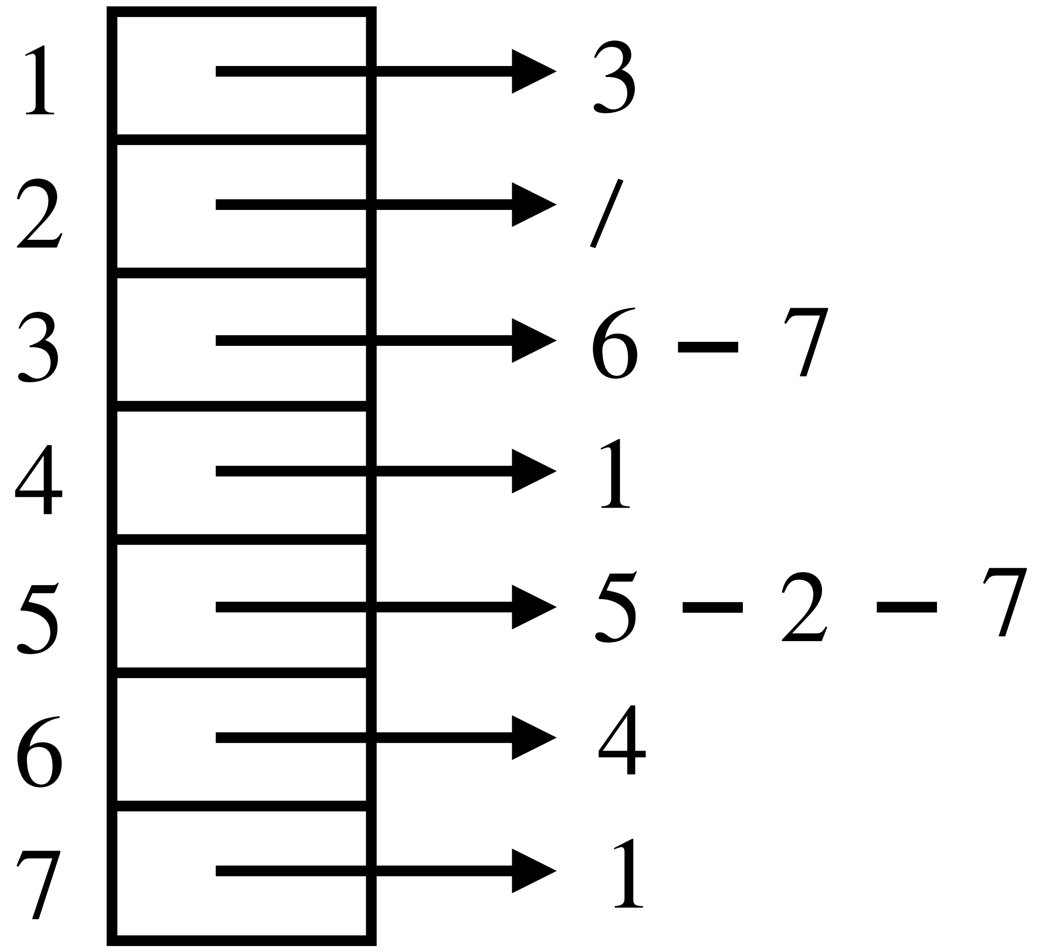
Graphs: Representation

Adjacency-list representation:

Graph G



G's adjacency list

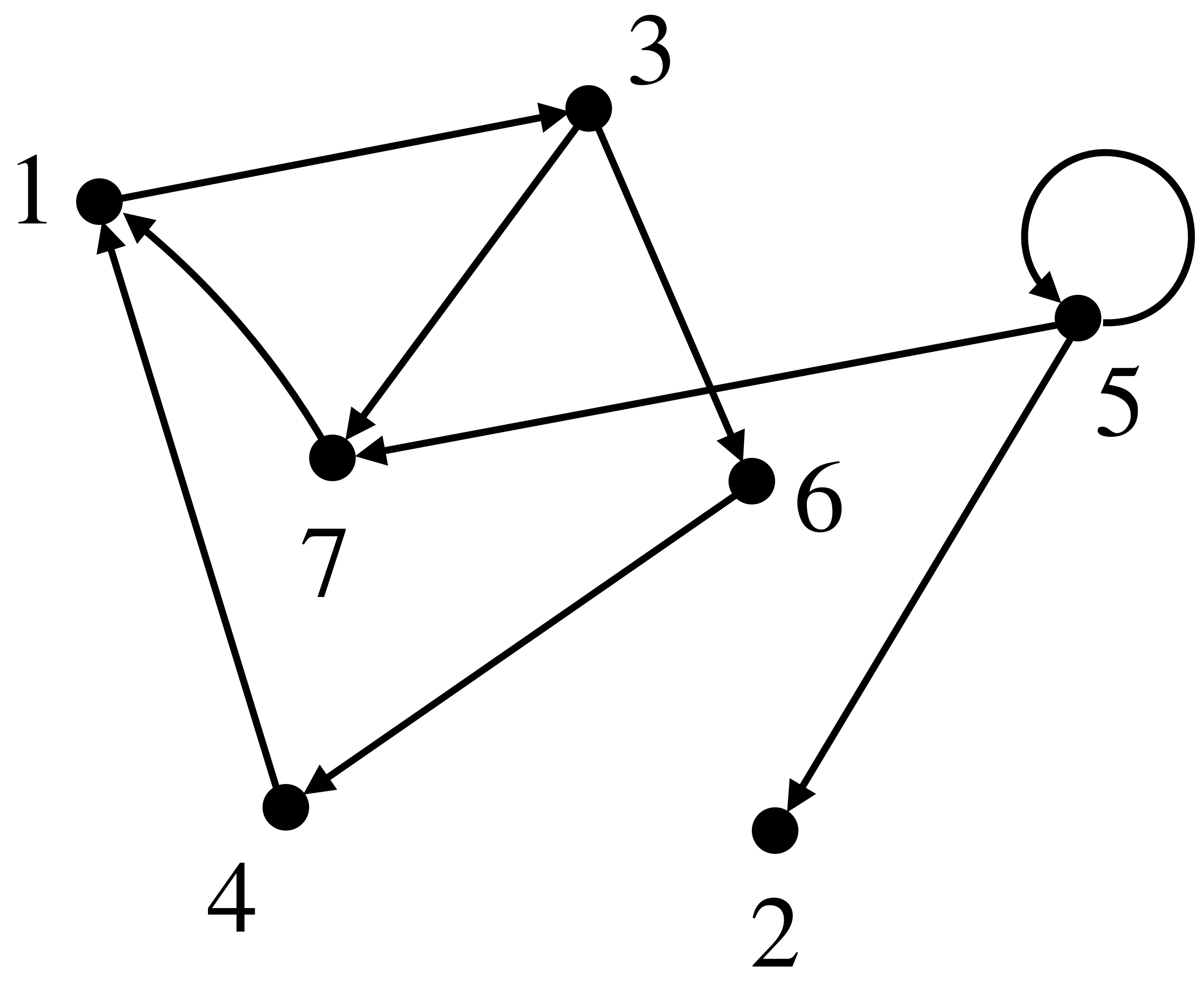


The adjacency-list representation of a graph $G = (V, E)$ consists of an array Adj of $|V|$ lists, one for each vertex in V. For each $u \in V$, the adjacency-list $Adj[u]$ contains all the vertices v such that there is an edge $(u, v) \in E$.

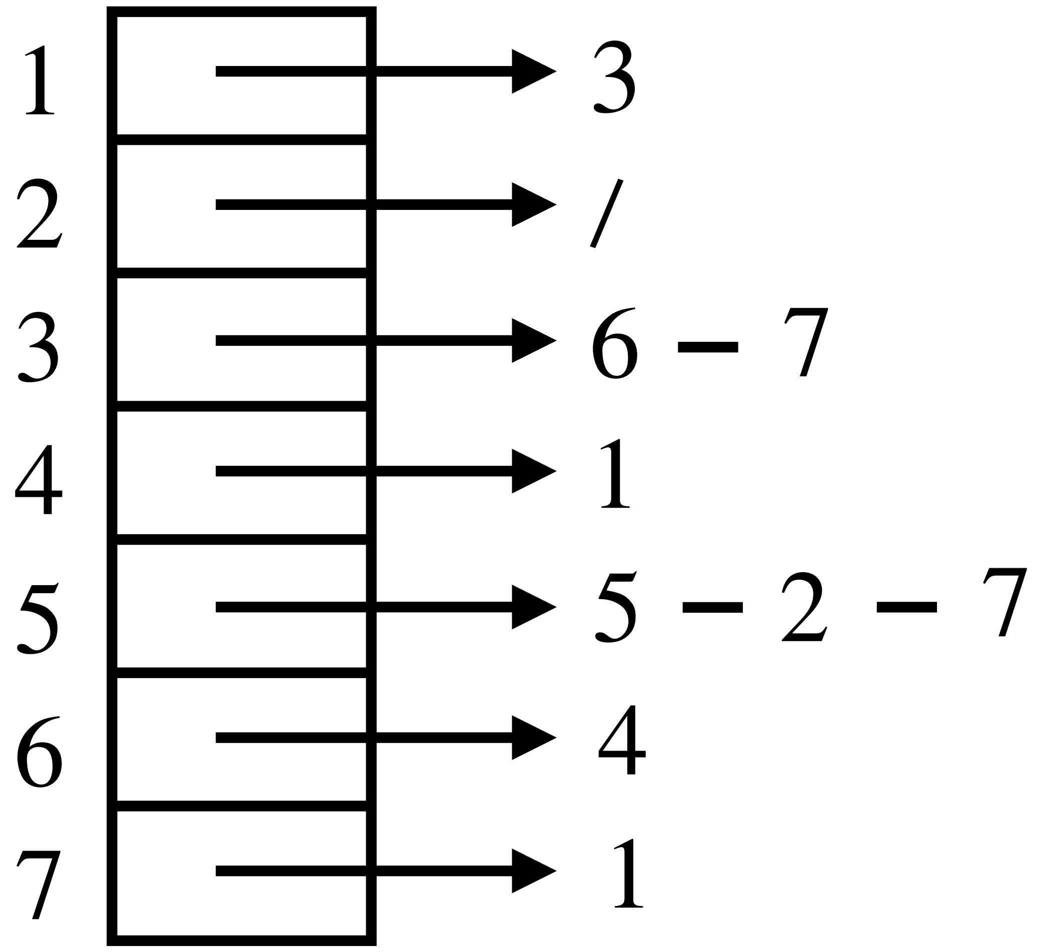
Graphs: Representation

Adjacency-list representation:

Graph G



G's adjacency list



The adjacency-list representation of a graph $G = (V, E)$ consists of an array Adj of $|V|$ lists, one for each vertex in V. For each $u \in V$, the adjacency-list $Adj[u]$ contains all the vertices v such that there is an edge $(u, v) \in E$.

Note: In case of undirected graphs, for an edge $\{i, j\}$, $i \in Adj[j]$ and $j \in Adj[i]$.

Graphs: Representation

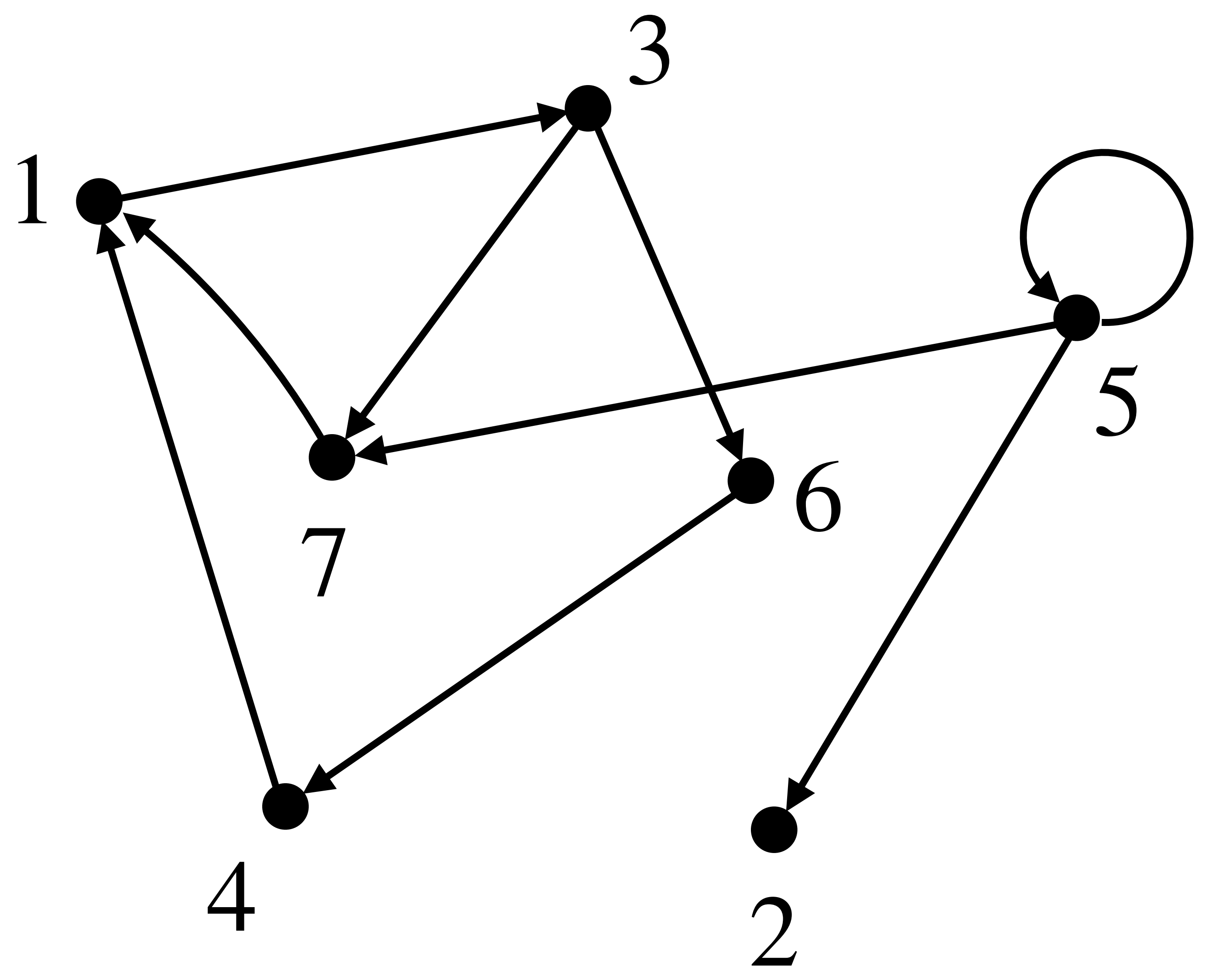
Graphs: Representation

Adjacency-matrix representation:

Graphs: Representation

Adjacency-matrix representation:

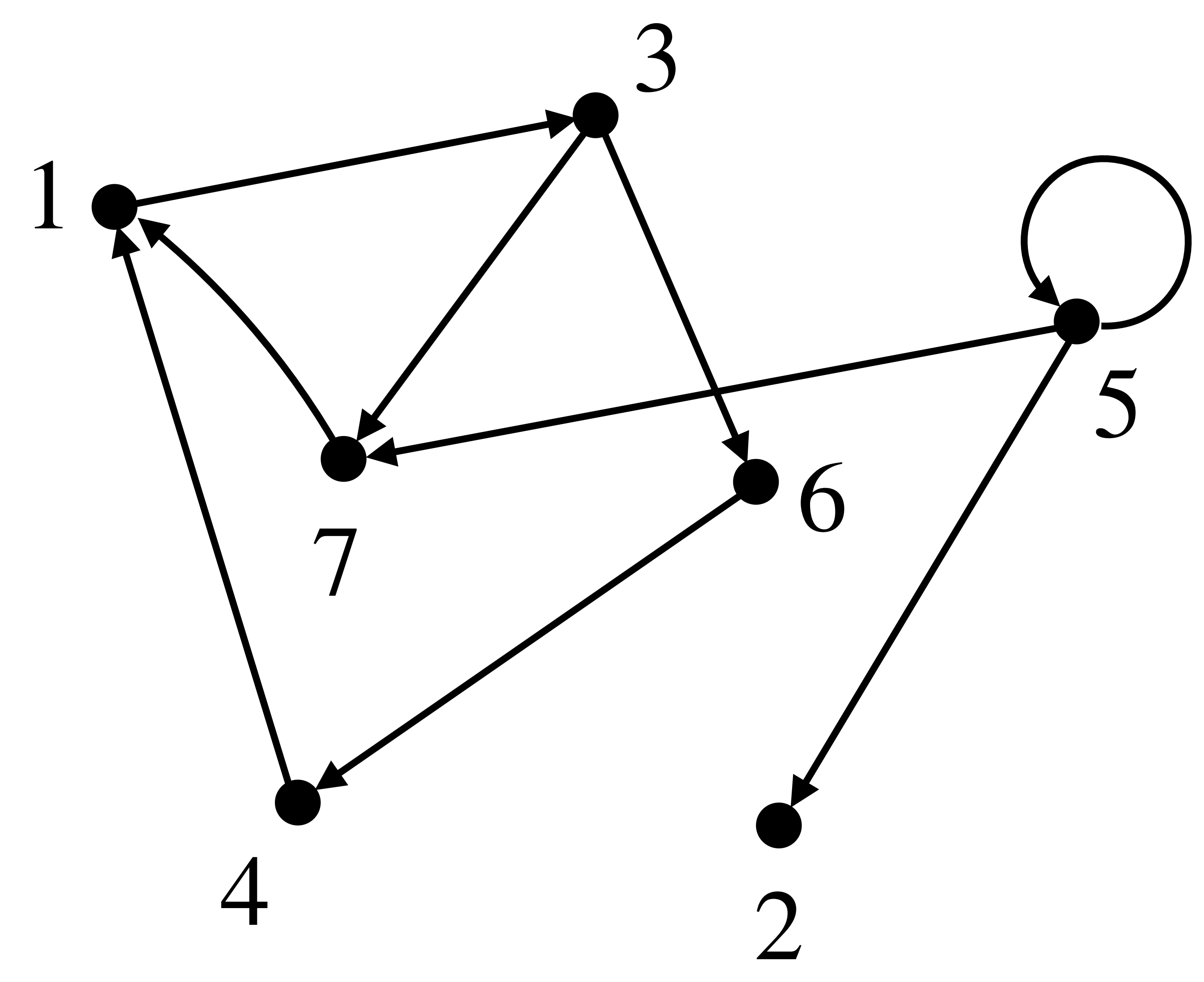
Graph G



Graphs: Representation

Adjacency-matrix representation:

Graph G



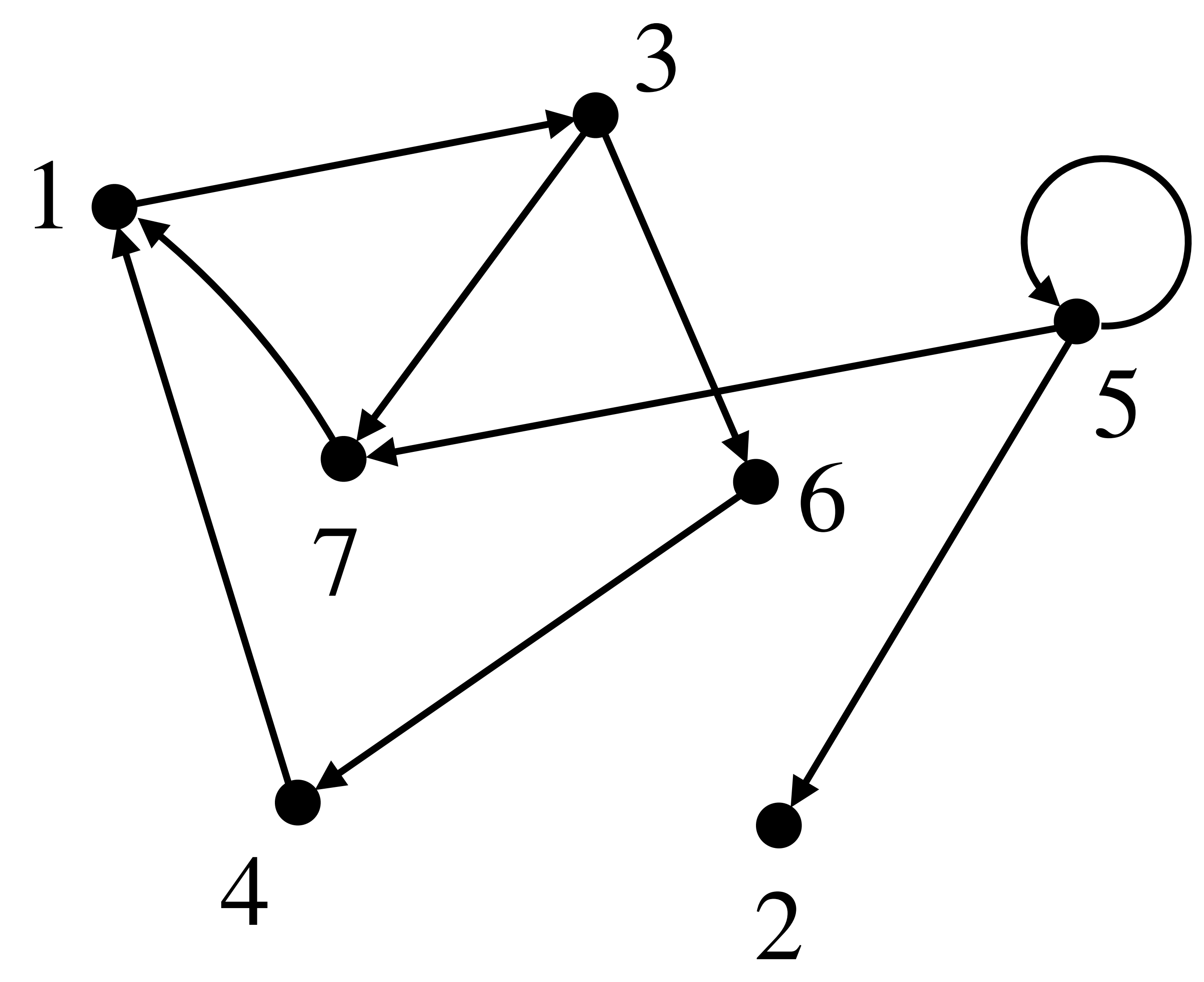
G's adjacency matrix

	1	2	3	4	5	6	7
1							
2							
3							
4							
5							
6							
7							

Graphs: Representation

Adjacency-matrix representation:

Graph G



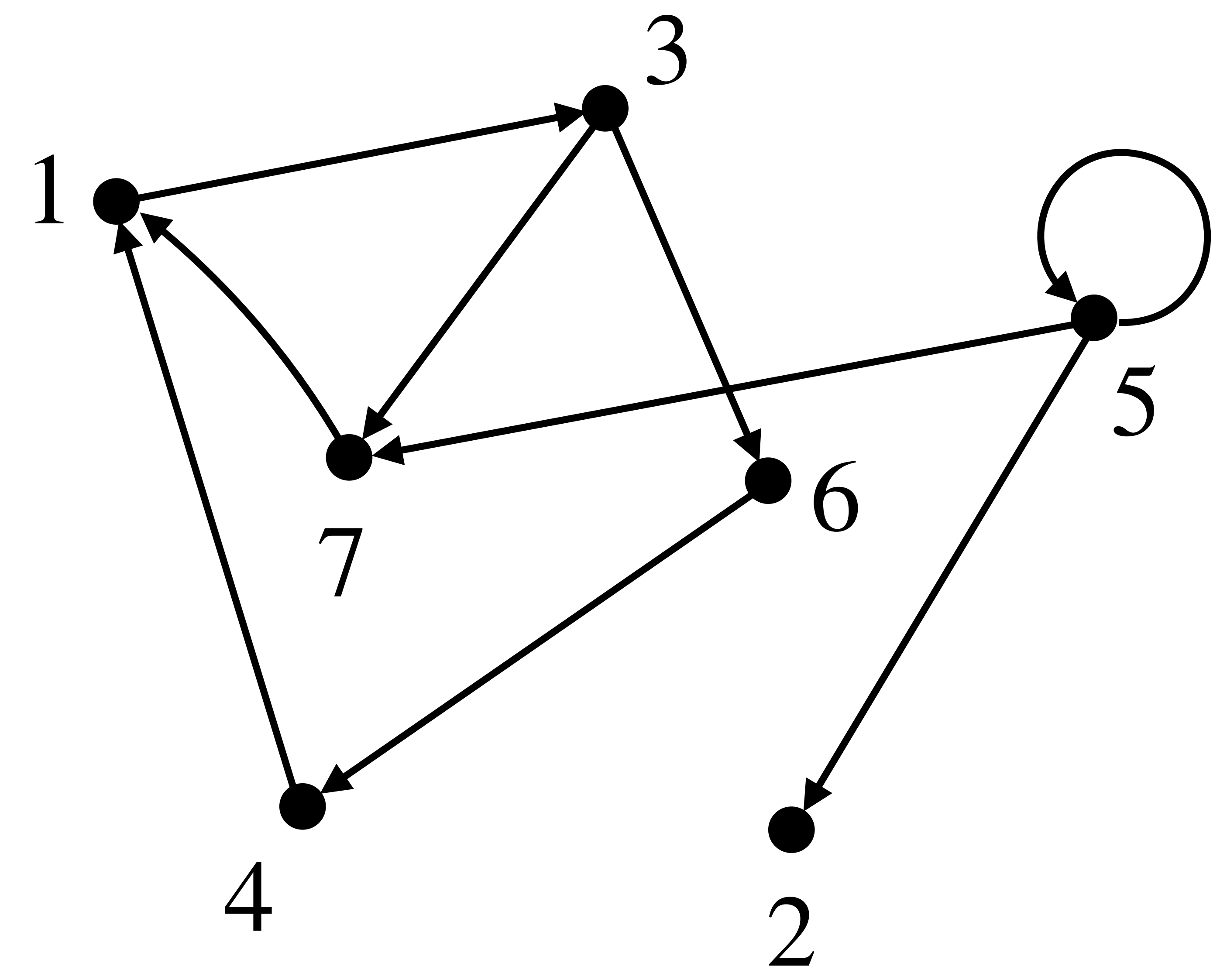
G's adjacency matrix

	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2							
3							
4							
5							
6							
7							

Graphs: Representation

Adjacency-matrix representation:

Graph G



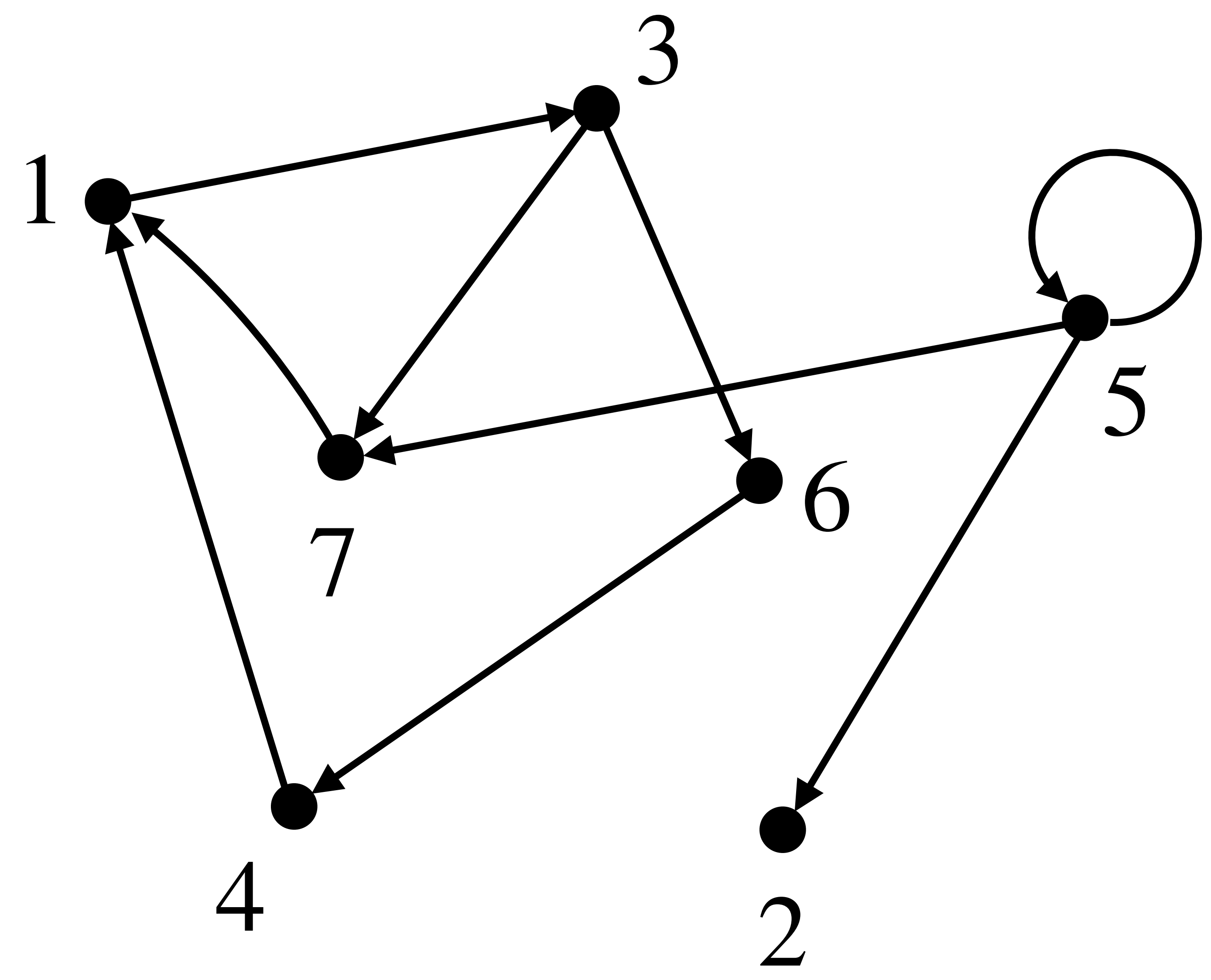
G's adjacency matrix

	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0
3							
4							
5							
6							
7							

Graphs: Representation

Adjacency-matrix representation:

Graph G



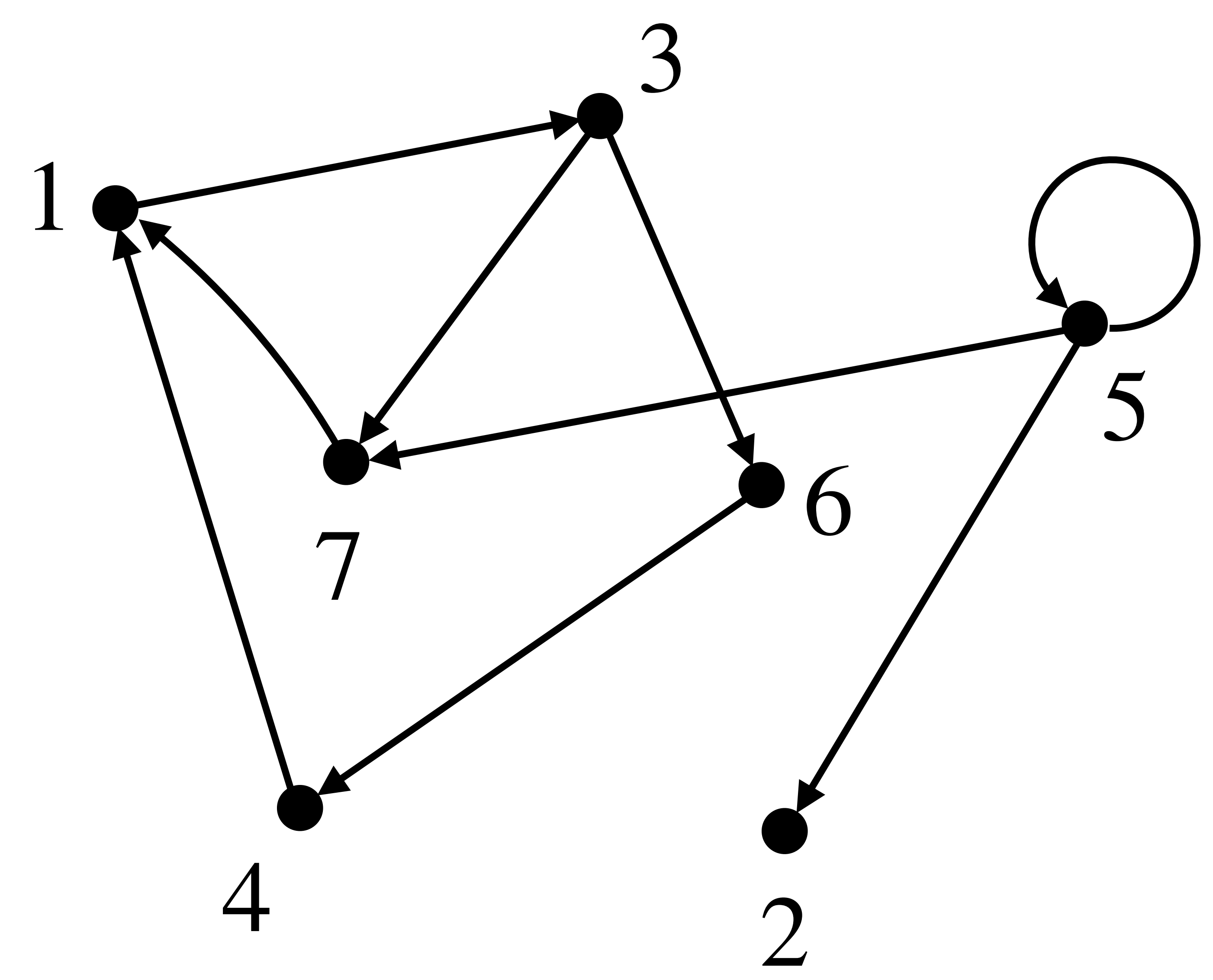
G's adjacency matrix

	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	1	1
4							
5							
6							
7							

Graphs: Representation

Adjacency-matrix representation:

Graph G



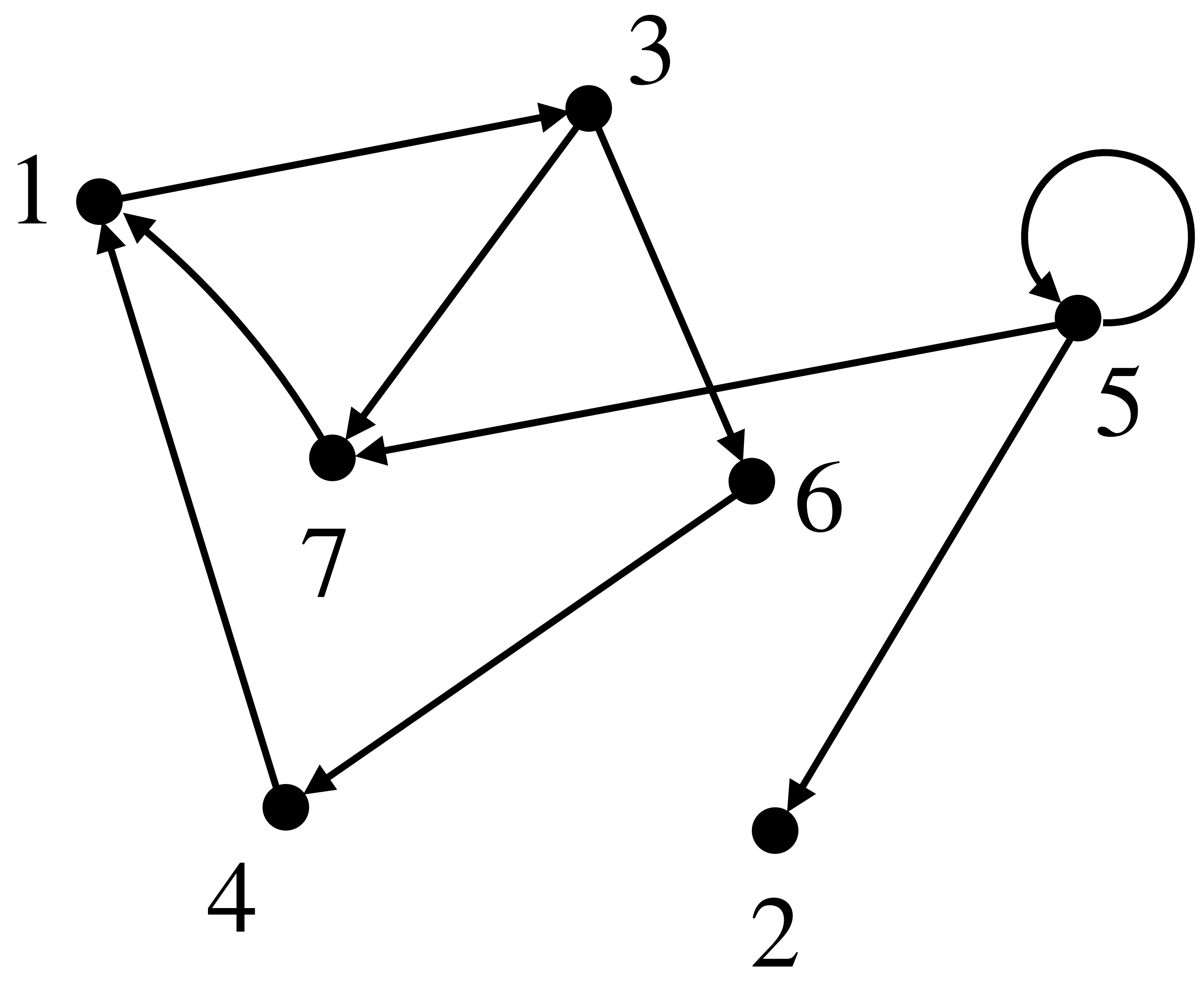
G's adjacency matrix

	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	1	1
4	1	0	0	0	0	0	0
5							
6							
7							

Graphs: Representation

Adjacency-matrix representation:

Graph G



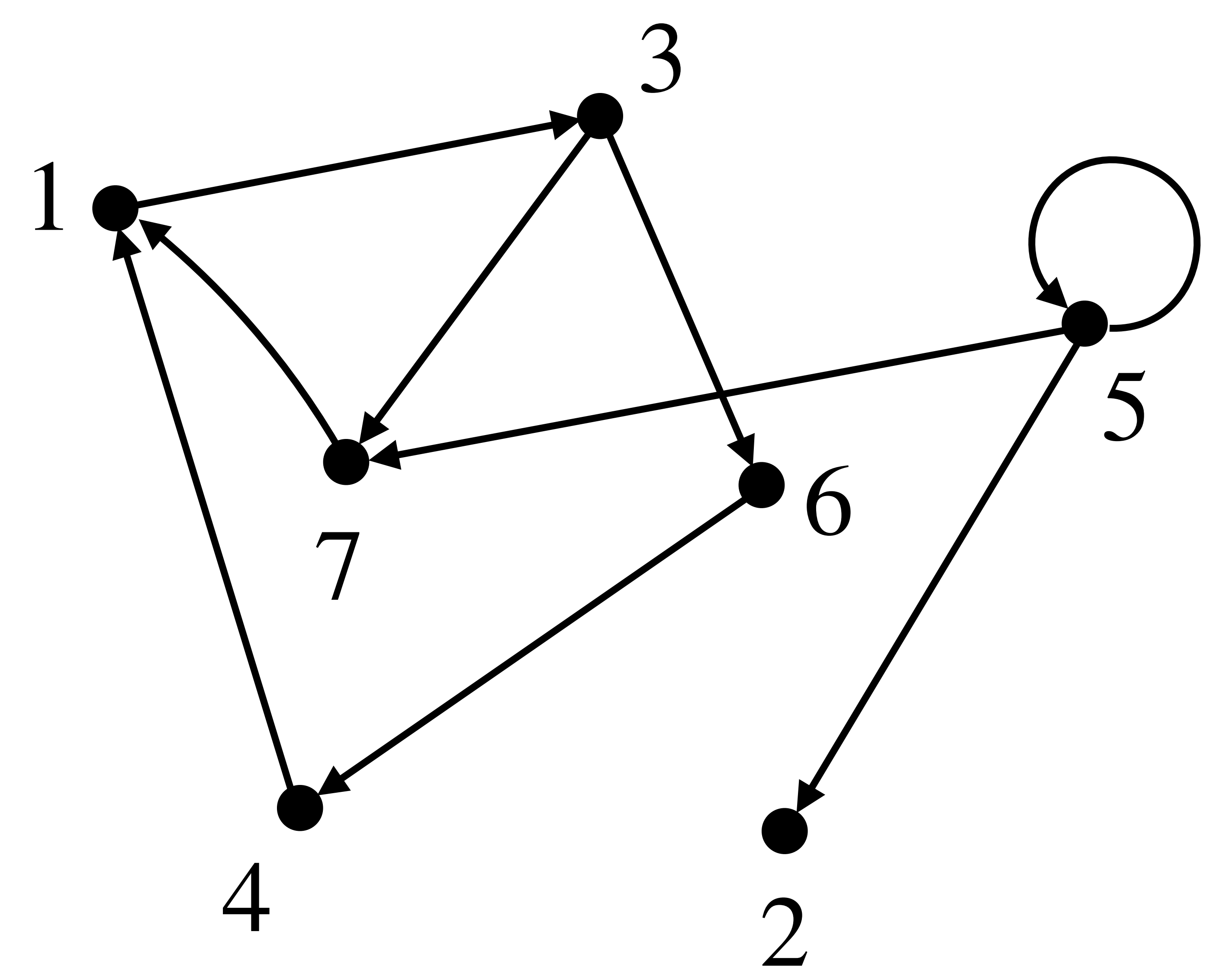
G's adjacency matrix

	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	1	1
4	1	0	0	0	0	0	0
5	0	1	0	0	1	0	1
6							
7							

Graphs: Representation

Adjacency-matrix representation:

Graph G



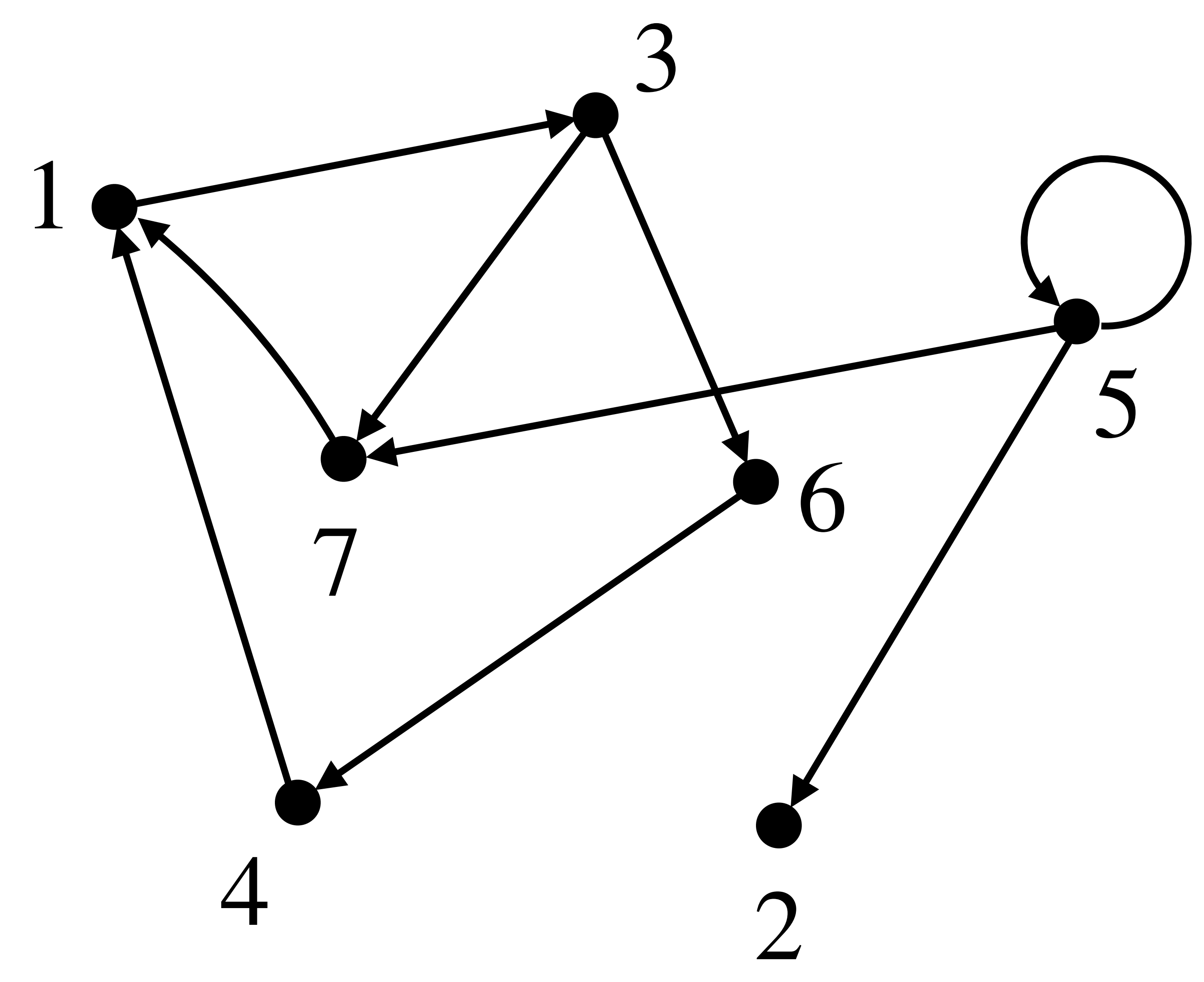
G's adjacency matrix

	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	1	1
4	1	0	0	0	0	0	0
5	0	1	0	0	1	0	1
6	0	0	0	1	0	0	0
7							

Graphs: Representation

Adjacency-matrix representation:

Graph G



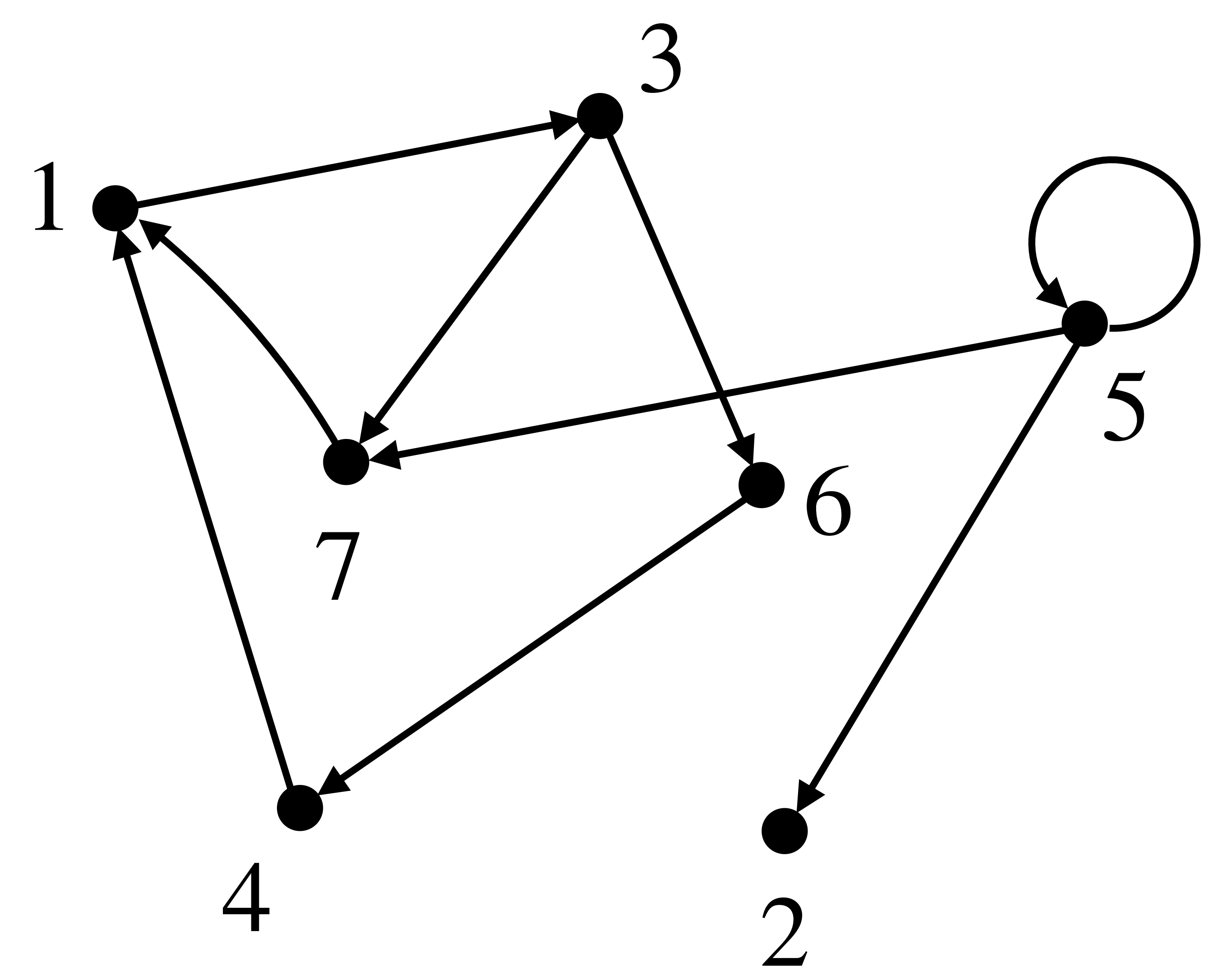
G's adjacency matrix

	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	1	1
4	1	0	0	0	0	0	0
5	0	1	0	0	1	0	1
6	0	0	0	1	0	0	0
7	1	0	0	0	0	0	0

Graphs: Representation

Adjacency-matrix representation:

Graph G



G's adjacency matrix

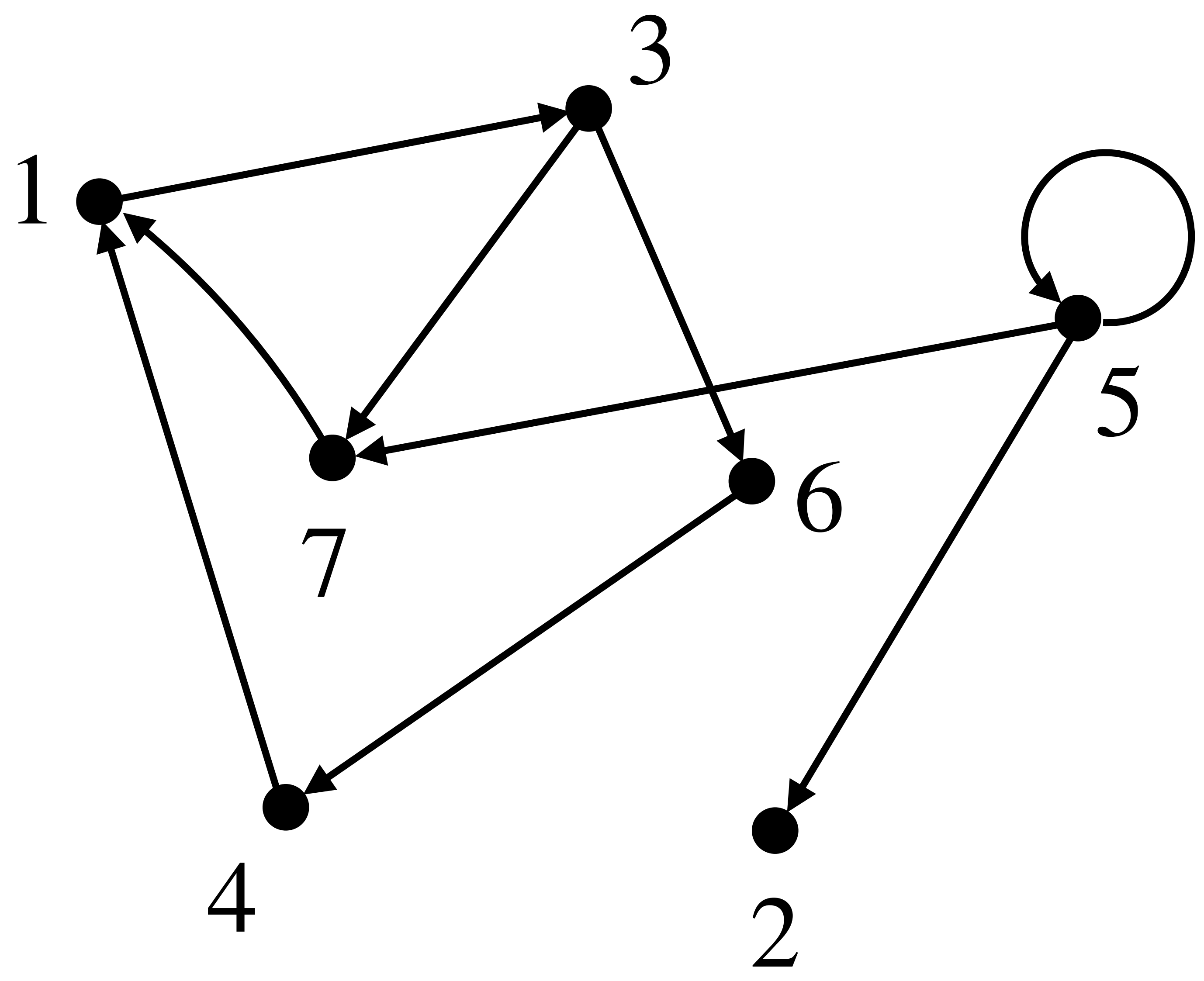
	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	1	1
4	1	0	0	0	0	0	0
5	0	1	0	0	1	0	1
6	0	0	0	1	0	0	0
7	1	0	0	0	0	0	0

The adjacency-matrix representation of a graph $G = (V, E)$ assumes that the vertices are

Graphs: Representation

Adjacency-matrix representation:

Graph G



G's adjacency matrix

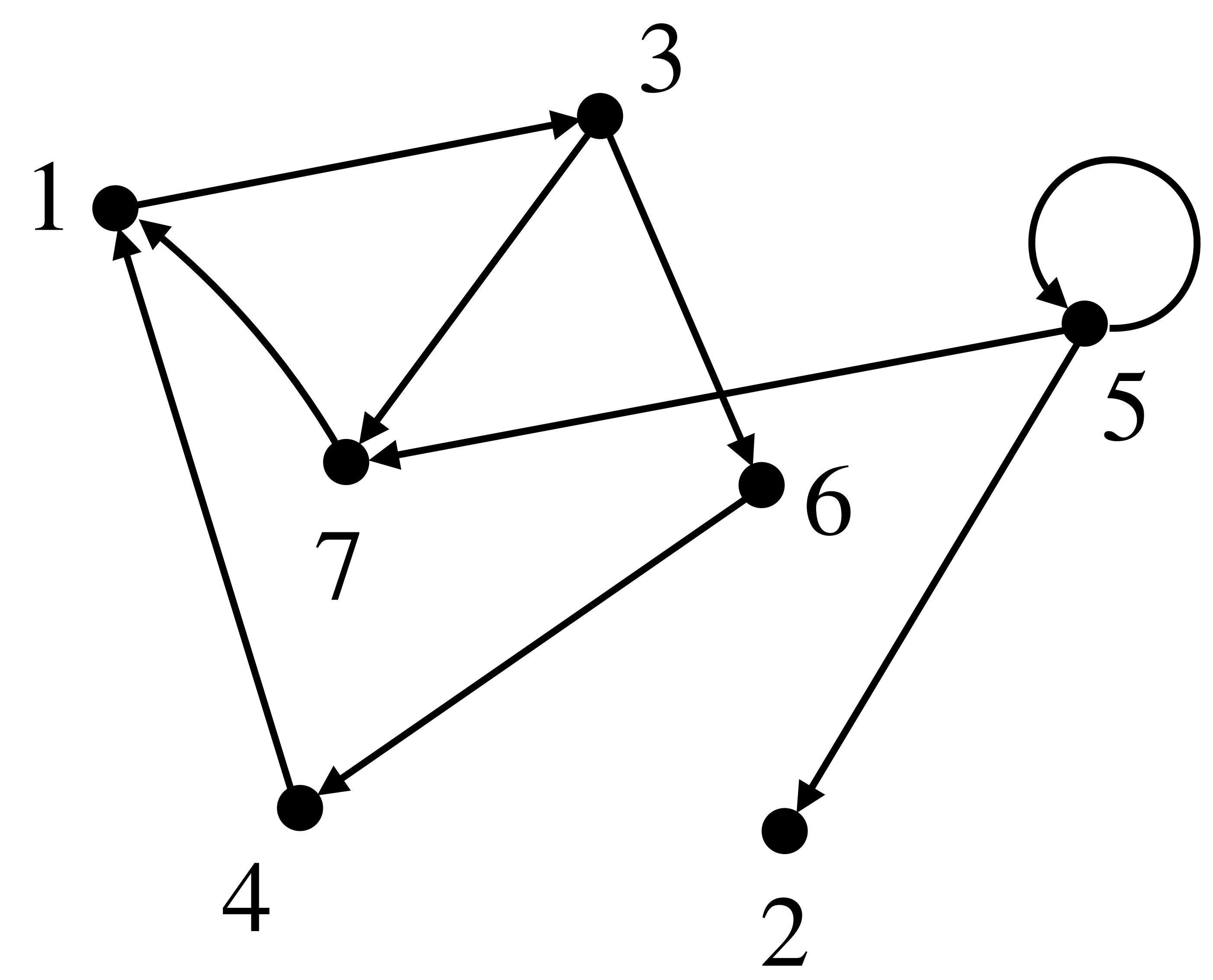
	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	1	1
4	1	0	0	0	0	0	0
5	0	1	0	0	1	0	1
6	0	0	0	1	0	0	0
7	1	0	0	0	0	0	0

The adjacency-matrix representation of a graph $G = (V, E)$ assumes that the vertices are numbered $1,2,\ldots,|V|$ in an arbitrary manner.

Graphs: Representation

Adjacency-matrix representation:

Graph G



G's adjacency matrix

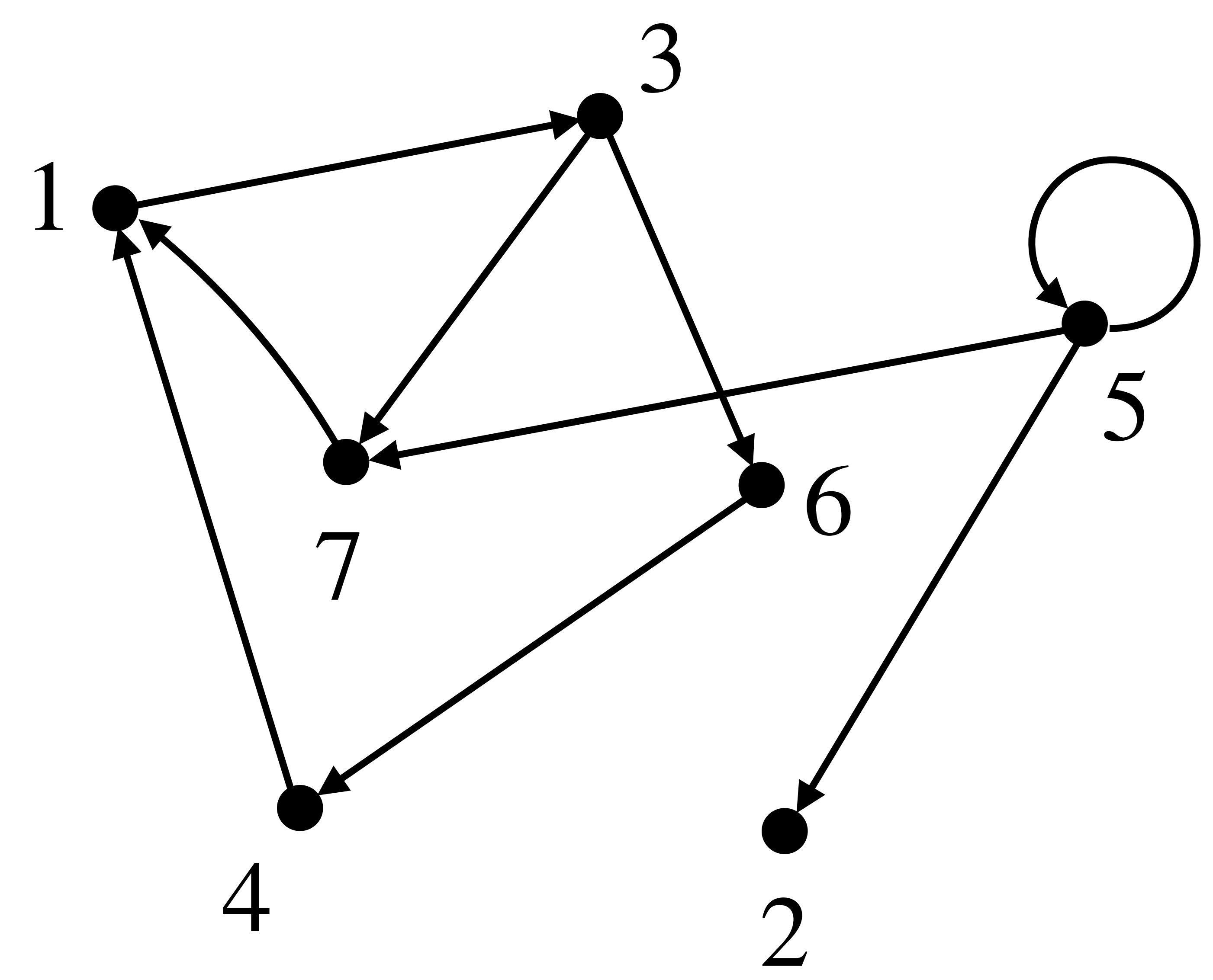
	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	1	1
4	1	0	0	0	0	0	0
5	0	1	0	0	1	0	1
6	0	0	0	1	0	0	0
7	1	0	0	0	0	0	0

The adjacency-matrix representation of a graph $G = (V, E)$ assumes that the vertices are numbered $1,2,\ldots,|V|$ in an arbitrary manner. Then, the adjacency matrix representation of a

Graphs: Representation

Adjacency-matrix representation:

Graph G



G's adjacency matrix

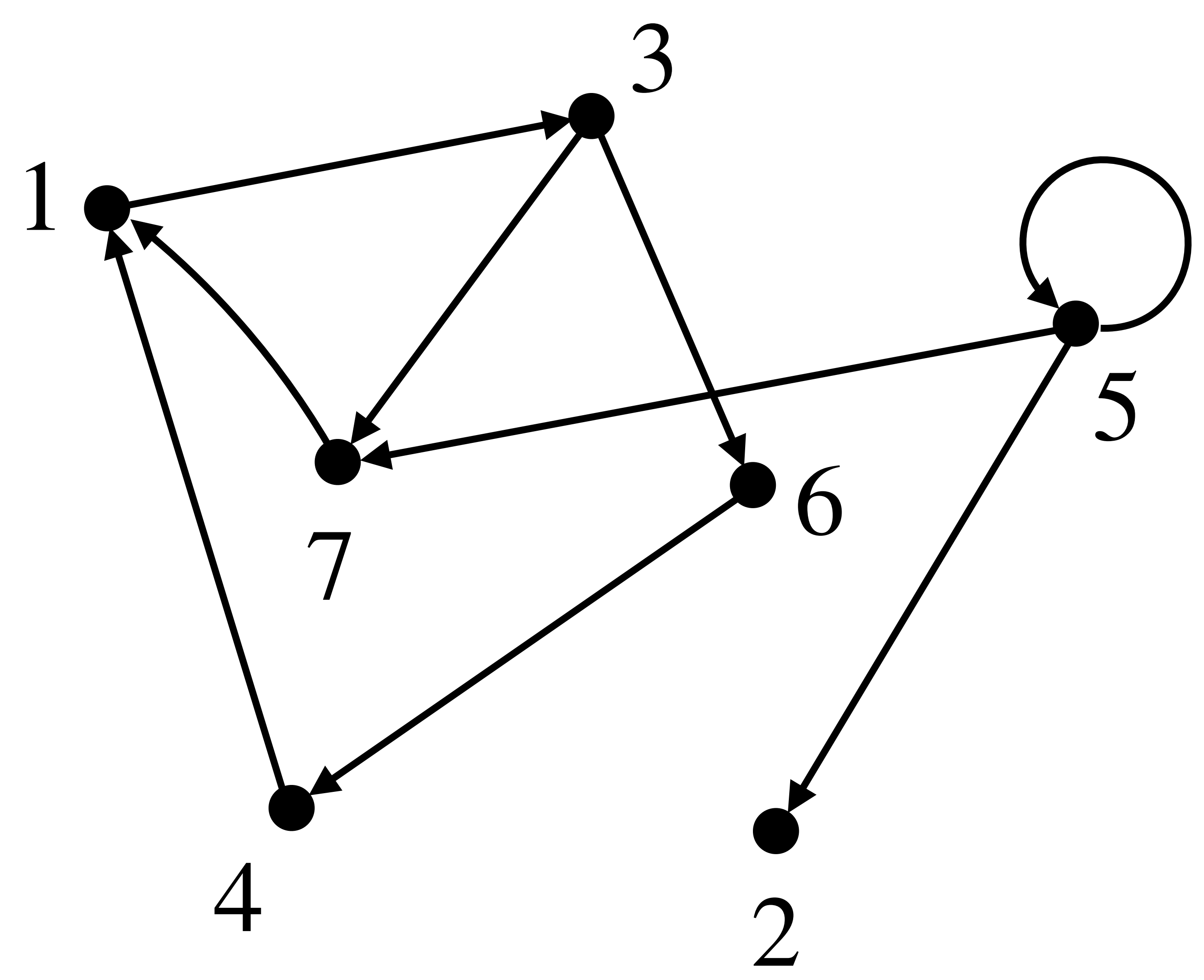
	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	1	1
4	1	0	0	0	0	0	0
5	0	1	0	0	1	0	1
6	0	0	0	1	0	0	0
7	1	0	0	0	0	0	0

The adjacency-matrix representation of a graph $G = (V, E)$ assumes that the vertices are numbered $1,2,\ldots,|V|$ in an arbitrary manner. Then, the adjacency matrix representation of a graph G consists of a $|V| \times |V|$ matrix A such that $A[i][j] = 1$, if $(i, j) \in E$, else $A[i][j] = 0$.

Graphs: Representation

Adjacency-matrix representation:

Graph G



G's adjacency matrix

	1	2	3	4	5	6	7
1	0	0	1	0	0	0	0
2	0	0	0	0	0	0	0
3	0	0	0	0	0	1	1
4	1	0	0	0	0	0	0
5	0	1	0	0	1	0	1
6	0	0	0	1	0	0	0
7	1	0	0	0	0	0	0

The adjacency-matrix representation of a graph $G = (V, E)$ assumes that the vertices are numbered $1,2,\ldots,|V|$ in an arbitrary manner. Then, the adjacency matrix representation of a graph G consists of a $|V| \times |V|$ matrix A such that $A[i][j] = 1$, if $(i, j) \in E$, else $A[i][j] = 0$.

Note: In case of undirected graphs, for an edge (i, j), $Adj[i][j] = Adj[j][i]$.

Graphs: Representation

Graphs: Representation

Some observations on adjacency-list and adjacency-matrix representation:

Graphs: Representation

Some observations on adjacency-list and adjacency-matrix representation:

- Adjacency-list requires $\Theta(|V| + |E|)$ space and adjacency-matrix representation requires $\Theta(|V|^2)$ space.

Graphs: Representation

Some observations on adjacency-list and adjacency-matrix representation:

- Adjacency-list requires $\Theta(|V| + |E|)$ space and adjacency-matrix representation requires $\Theta(|V|^2)$ space.
- Adjacency-list is preferred when you need to go over all the neighbours of a vertex.

Graphs: Representation

Some observations on adjacency-list and adjacency-matrix representation:

- Adjacency-list requires $\Theta(|V| + |E|)$ space and adjacency-matrix representation requires $\Theta(|V|^2)$ space.
- Adjacency-list is preferred when you need to go over all the neighbours of a vertex.
- Adjacency-matrix is preferred when you need to quickly determine the existence of an edge between two vertices i and j.

Graphs: Representation

Some observations on adjacency-list and adjacency-matrix representation:

- Adjacency-list requires $\Theta(|V| + |E|)$ space and adjacency-matrix representation requires $\Theta(|V|^2)$ space.
- Adjacency-list is preferred when you need to go over all the neighbours of a vertex.
- Adjacency-matrix is preferred when you need to quickly determine the existence of an edge between two vertices i and j.
- In adjacency-list of a directed graph, sum of size of all the lists $= |E|$.

Graphs: Representation

Some observations on adjacency-list and adjacency-matrix representation:

- Adjacency-list requires $\Theta(|V| + |E|)$ space and adjacency-matrix representation requires $\Theta(|V|^2)$ space.
- Adjacency-list is preferred when you need to go over all the neighbours of a vertex.
- Adjacency-matrix is preferred when you need to quickly determine the existence of an edge between two vertices i and j.
- In adjacency-list of a directed graph, sum of size of all the lists $= |E|$.
- If A is an adjacency-matrix of an undirected graph, then $A = A^T$.