Eulerian graphs

Graph G is a pair (V, E), where V is the set of *vertices* and E is the set of *edges*. Besides that, we are given the *incidence function* $\mathcal{E}$.

Walk in the graph G is a sequence

$$v_0 \overset{e_1}{\text{———}} v_1 \overset{e_2}{\text{———}} v_2 \overset{e_3}{\text{———}} v_3 \overset{e_4}{\text{———}} \ldots v_{k-1} \overset{e_k}{\text{———}} v_k,$$

where $v_0, \ldots, v_k \in V$, $e_1, \ldots, e_k \in E$ and $\mathcal{E}(e_i) = \{v_{i-1}, v_i\}$.

The walk is *closed*, if its first and last vertices coincide.

Path is a walk where every vertex occurs at most once.

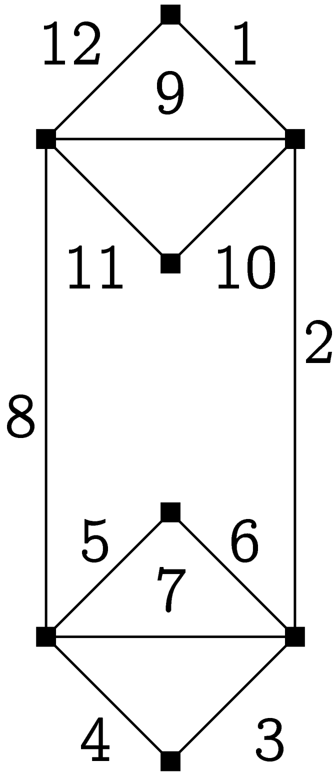
Cycle is a closed path.

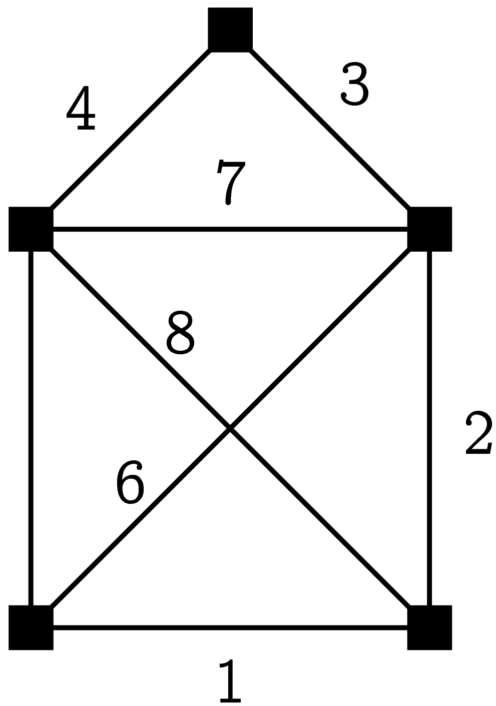
Eulerian walk in the graph $G = (V, E)$ is a closed walk covering each edge exactly once.

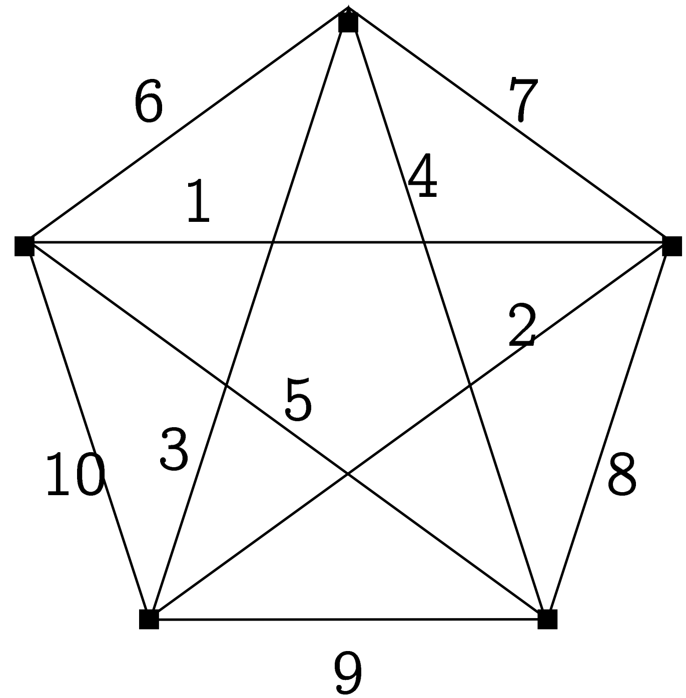
Eulerian graph is a graph with a Eulerian walk.

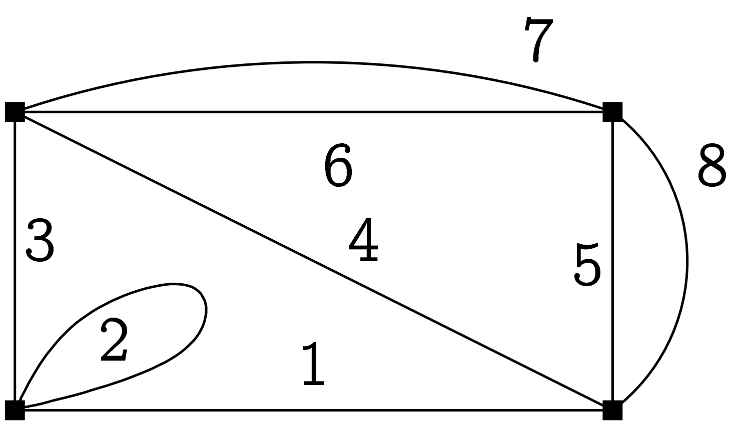
A graph that has a non-closed walk covering each edge exactly once is called *semi-Eulerian*.

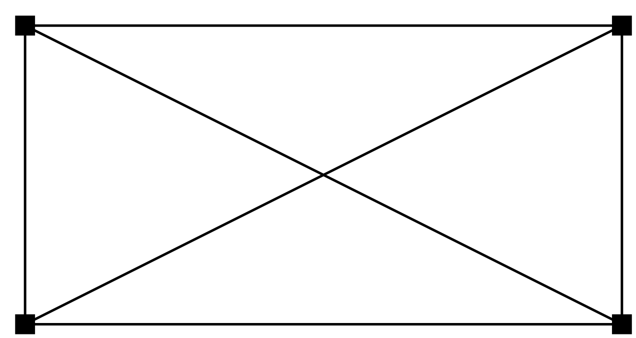
A well-known class of puzzles: draw the figure without raising the pen from the paper and covering each line exactly once.


12
1
9
11
10
2
8
5
6
7
4
3

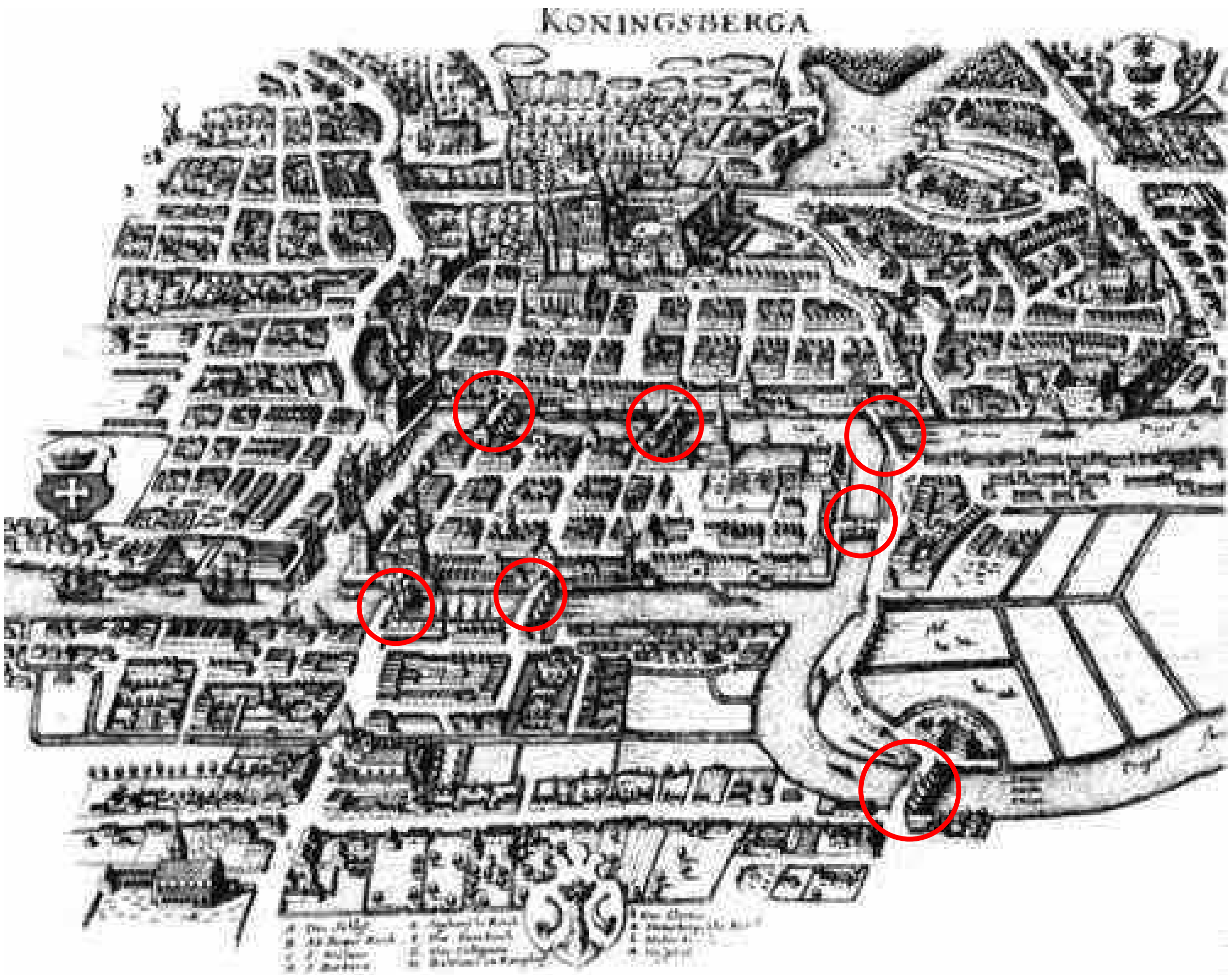

4
3
7
8
5
2
6
1

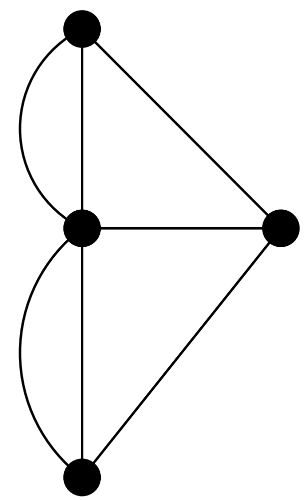

6
7
4
1
2
5
3
10
8
9


7
6
8
3
4
5
2
1



“Original problem”:





Theorem. Let $G = (V, E)$ be a connected graph. The following are equivalent:

(i). G is a Eulerian graph.

(ii). All vertex degrees of G are even.

(iii). E can be represented as a union of edge-wise non-intersecting cycles.

Proof (i)$\Rightarrow$(ii). Let P be some Eulerian walk of G and let $v \in V$.

The walk P enters v some number of times and also exits it the same number of times. Thus the number of edges of P incident with v is even (again, loops are counted twice).

On the other hand, P is a Eulerian walk, thus the edges of P incident with v are exactly all the edges of G incident with v.

Proof (ii)$\Rightarrow$(iii). Induction over $|E|$.

Base. $|E| = 0$. Then E is a union of 0 pieces, each one of them is $\ldots$.

Step. $|E| > 0$. Since G is connected, all the vertex degrees must be positive.

According to (ii), all the vertex degrees are ≥ 2.

Using a theorem from the previous lecture, there is a cycle C in G.

Theorem. If all the vertex degrees in a graph are at least 2, then there is a cycle in this graph.

Delete all the edges of C from grapg G; let the remaining graph be G'.

G' has less edges than G and all its vertex degrees are still even.

Let $H_1, \ldots, H_k$ be the connected components of graph G'. Induction hypothesis implies that each of them can be represented as a union of edge-wise non-intersecting cycles. Adding the cycle C to the union of these representations, we have obtained the required representation for E.

Proof (iii)$\Rightarrow$(i). Let $E = C_1 \dot\cup C_2 \dot\cup \cdots \dot\cup C_n$, where $C_1, \ldots, C_n$ are cycles.

If $n = 1$, the claim is clear. Assume $n \geq 2$.

W.l.o.g assume that every cycle C_i $(i > 1)$ has a common vertex with some cycle C_j $(j < i)$.

We will now construct closed walks $P_1, \ldots, P_n$ so that each P_i covers each edge of the cycles $C_1, \ldots, C_i$ exactly once and does not cover any other edges.

Let the closed walk P_1 be the cycle C_1.

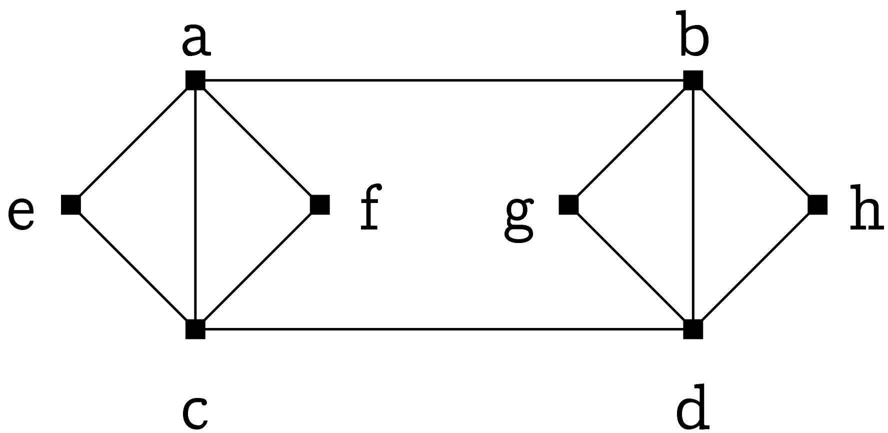
Construct the walk P_i based on the walk P_{i-1} as follows.

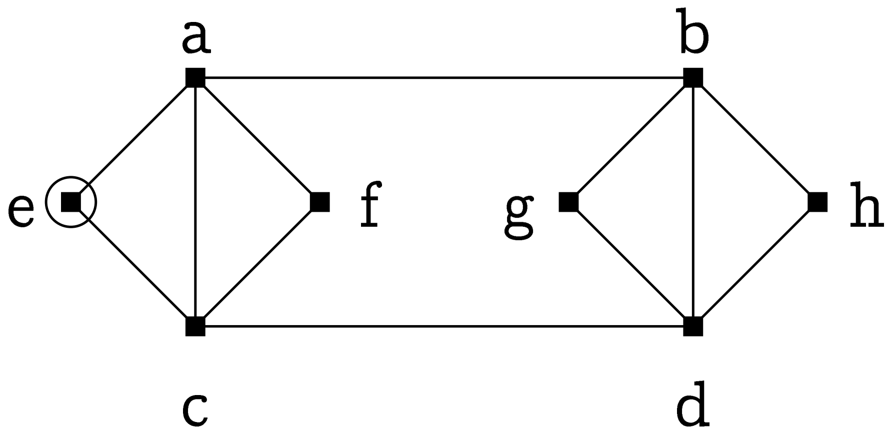
- Move along the walk P_{i-1} until we hit a vertex also present in the cycle C_i.
- Follow the cycle C_i starting and finishing in vertex v.
- Move along the rest of the walk P_{i-1}.

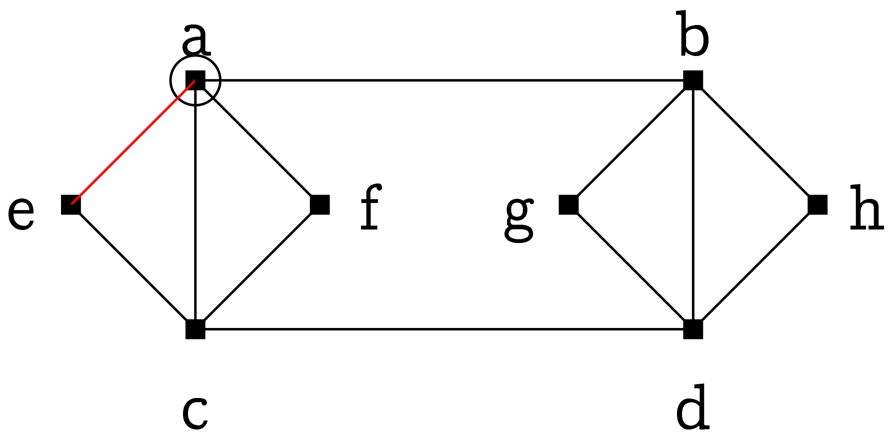
The walk P_n is a Eulerian one in graph G. $\square$

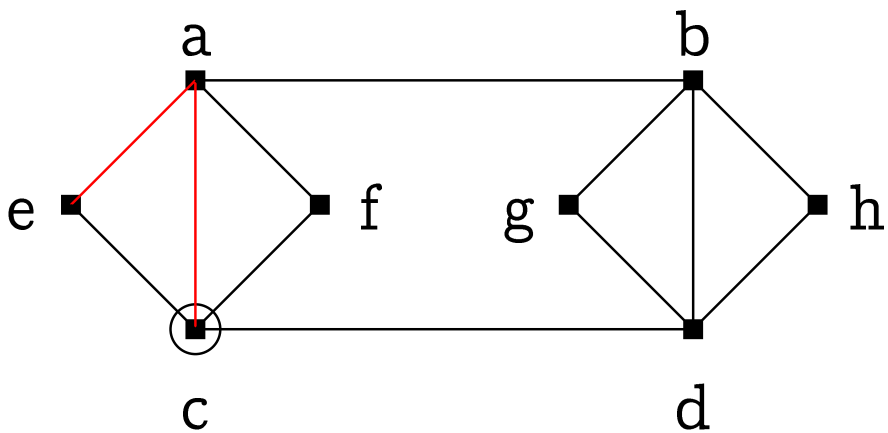
The proof gives an algorithm for finding a Eulerian cycle in a Eulerian graph G:

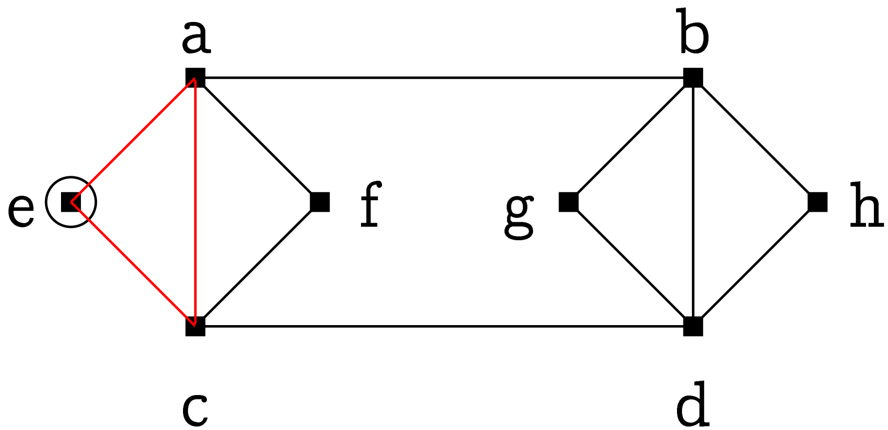
- Partition $E(G)$ into cycles.
  - Construct one of these cycles, say, C.
    - Move along the edges of G until we reach some vertex for the second time.
  - Remove the edges of C from graph G.
  - Partition the edges of the connected components of G (without C) to cycles.
  - Output these cycles and the cycle C.
- Construct a Eulerian walk as shown in the previous slide.

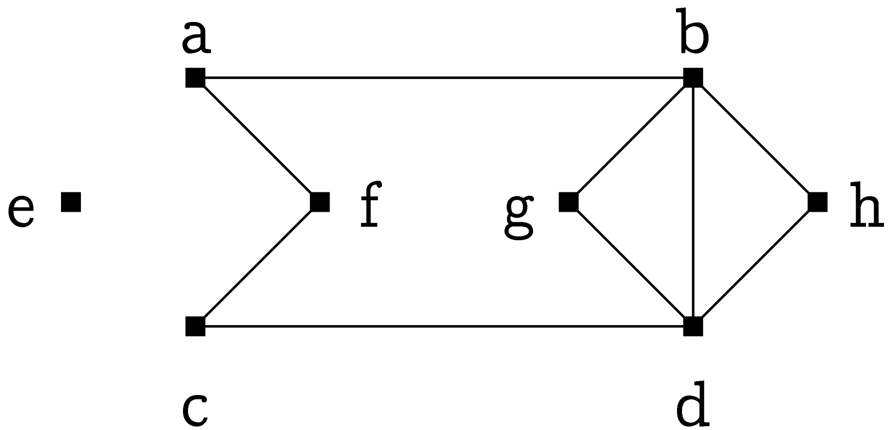

a
b
e
f
g
h
c
d

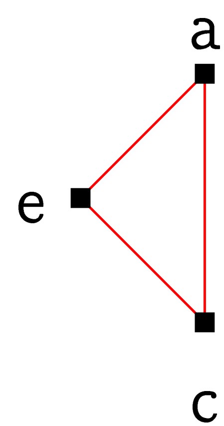

a
b
e
f
g
h
c
d

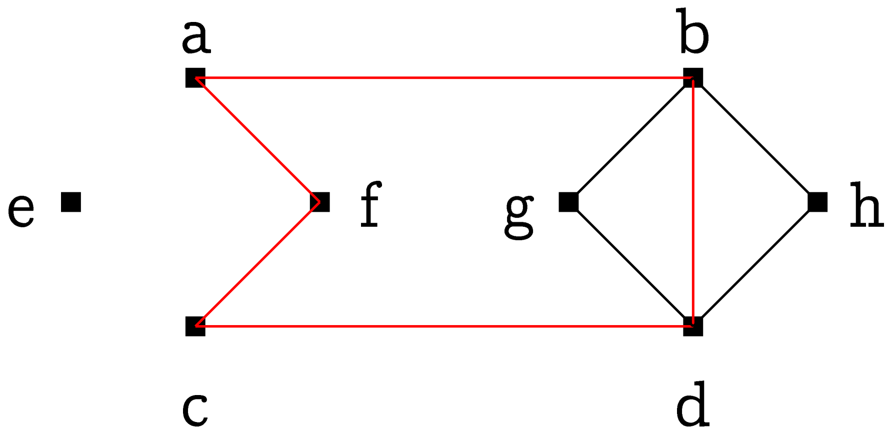

a
b
e
f
g
h
c
d

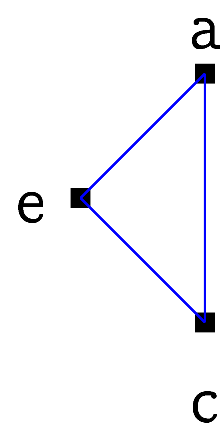

a
b
e
f
g
h
c
d

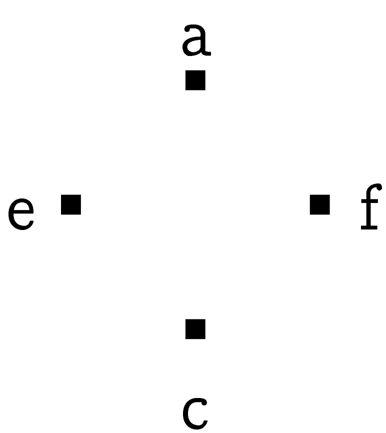
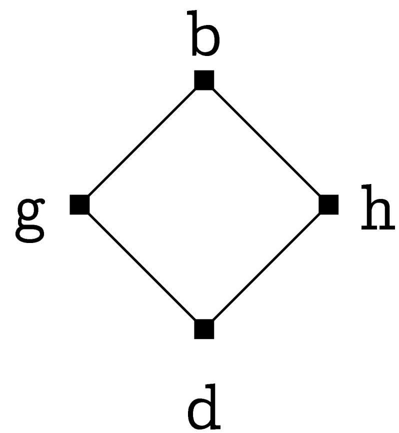
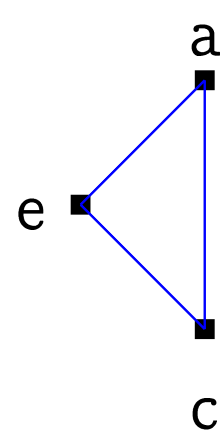
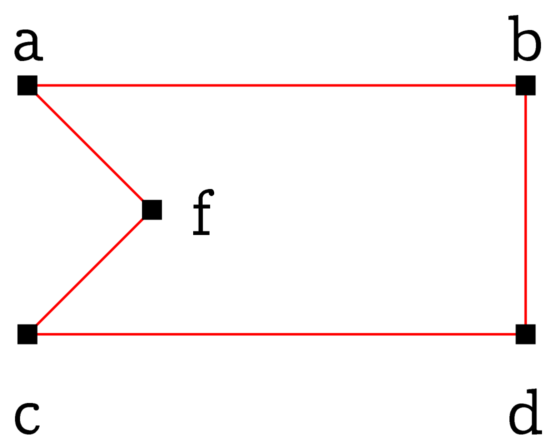

a
b
e
f
g
h
c
d

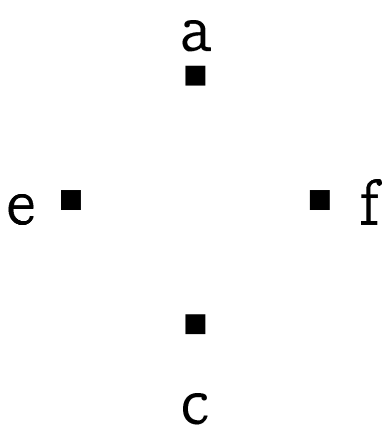

a
b
e
f
g
h
c
d

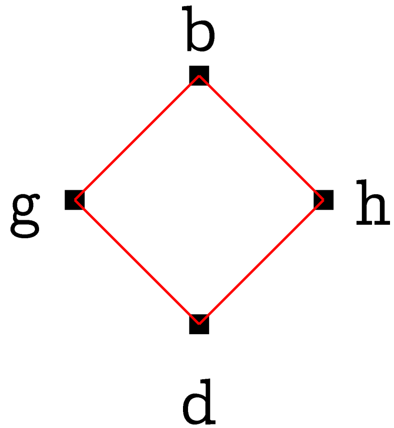

a
e
c

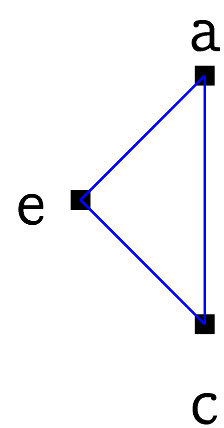

a
b
e
f
g
h
c
d

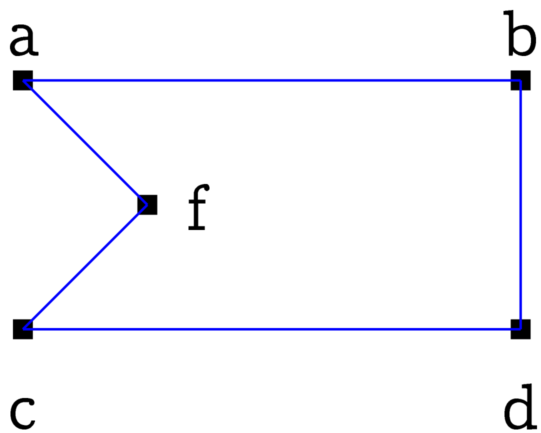

a
e
c

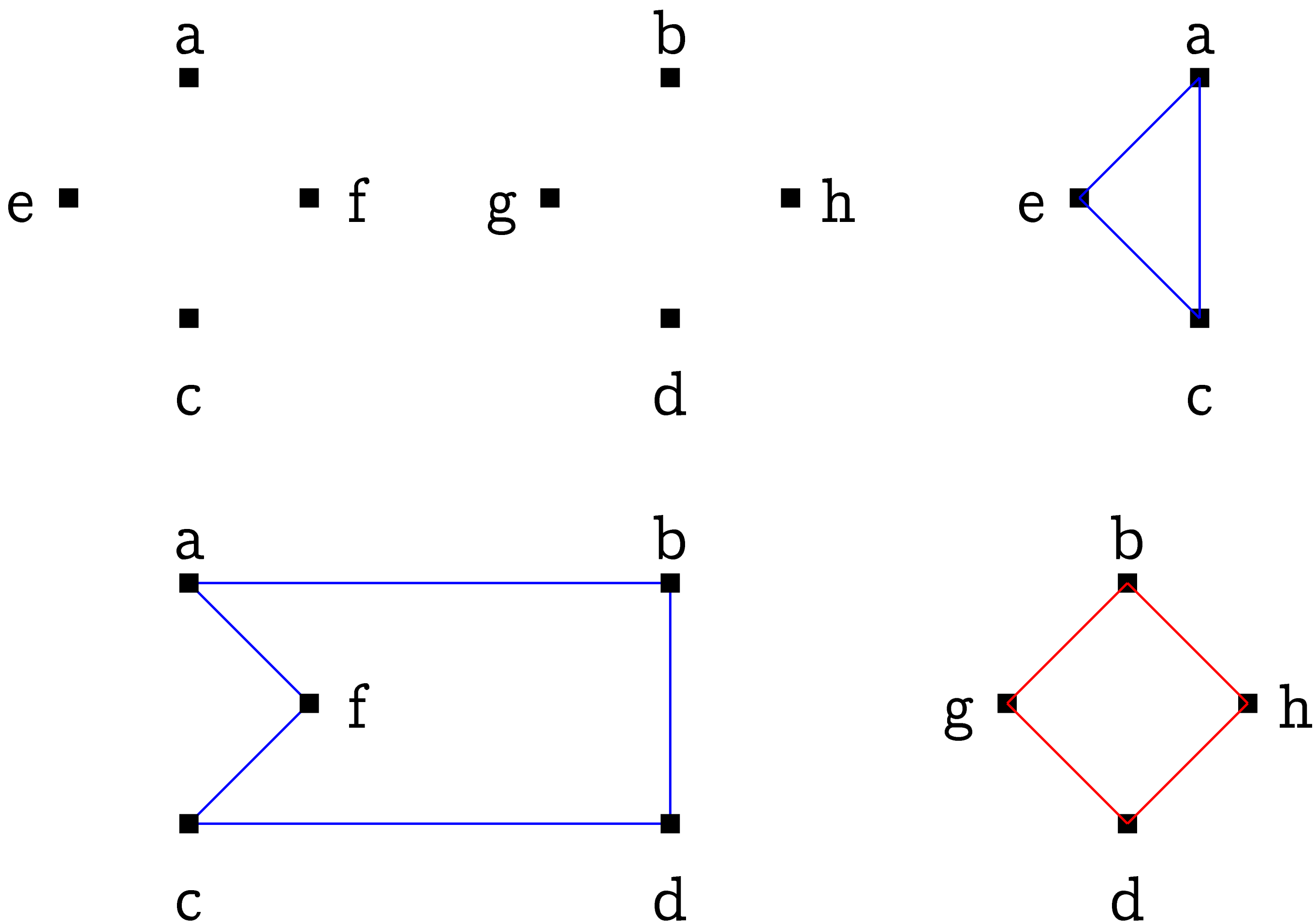

a
e
f
c

b
g
h
d

a
e
c

a
b
f
c
d

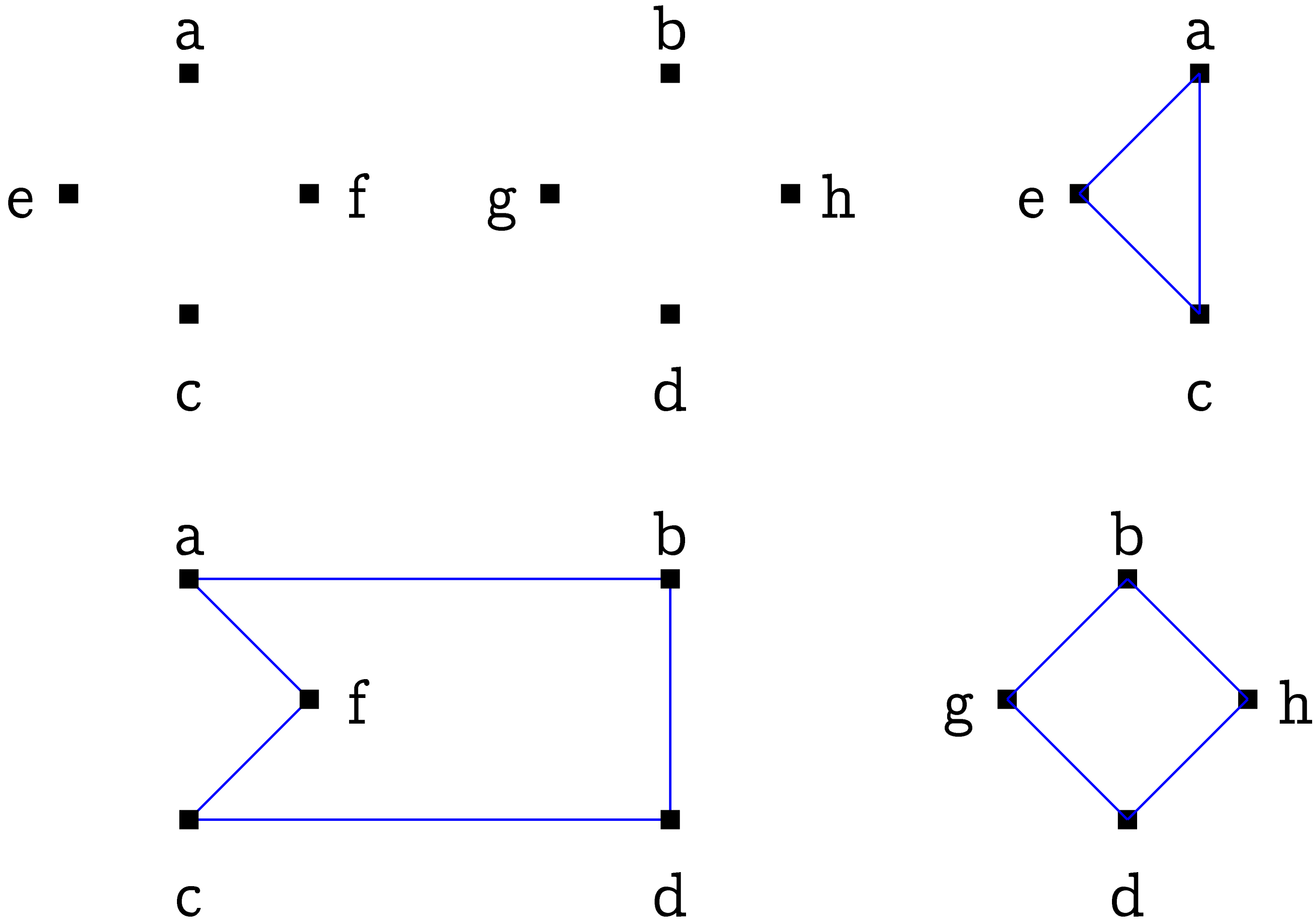

a
e
f
c


b
g
h
d

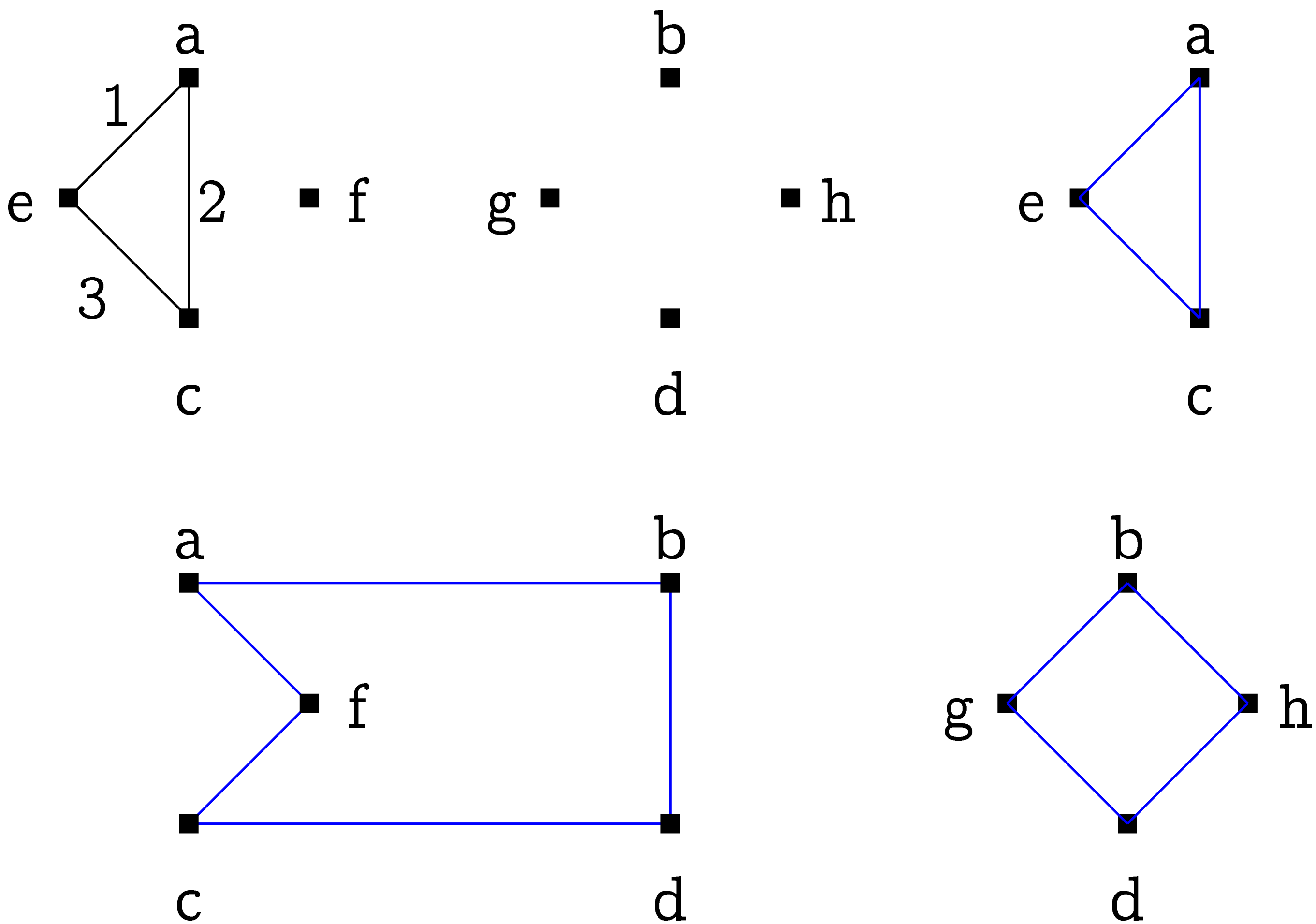

a
e
c

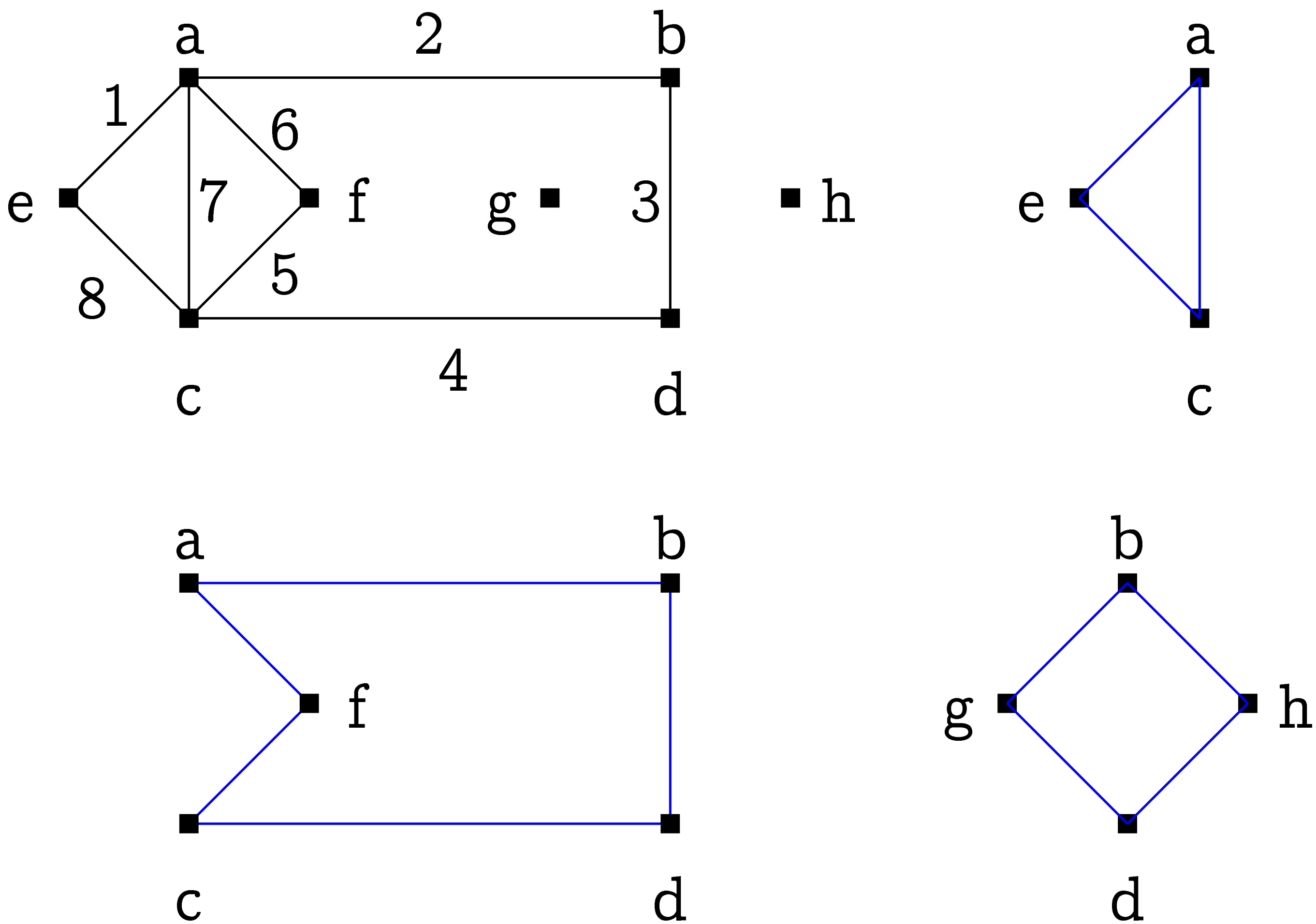

a
b
f
c
d

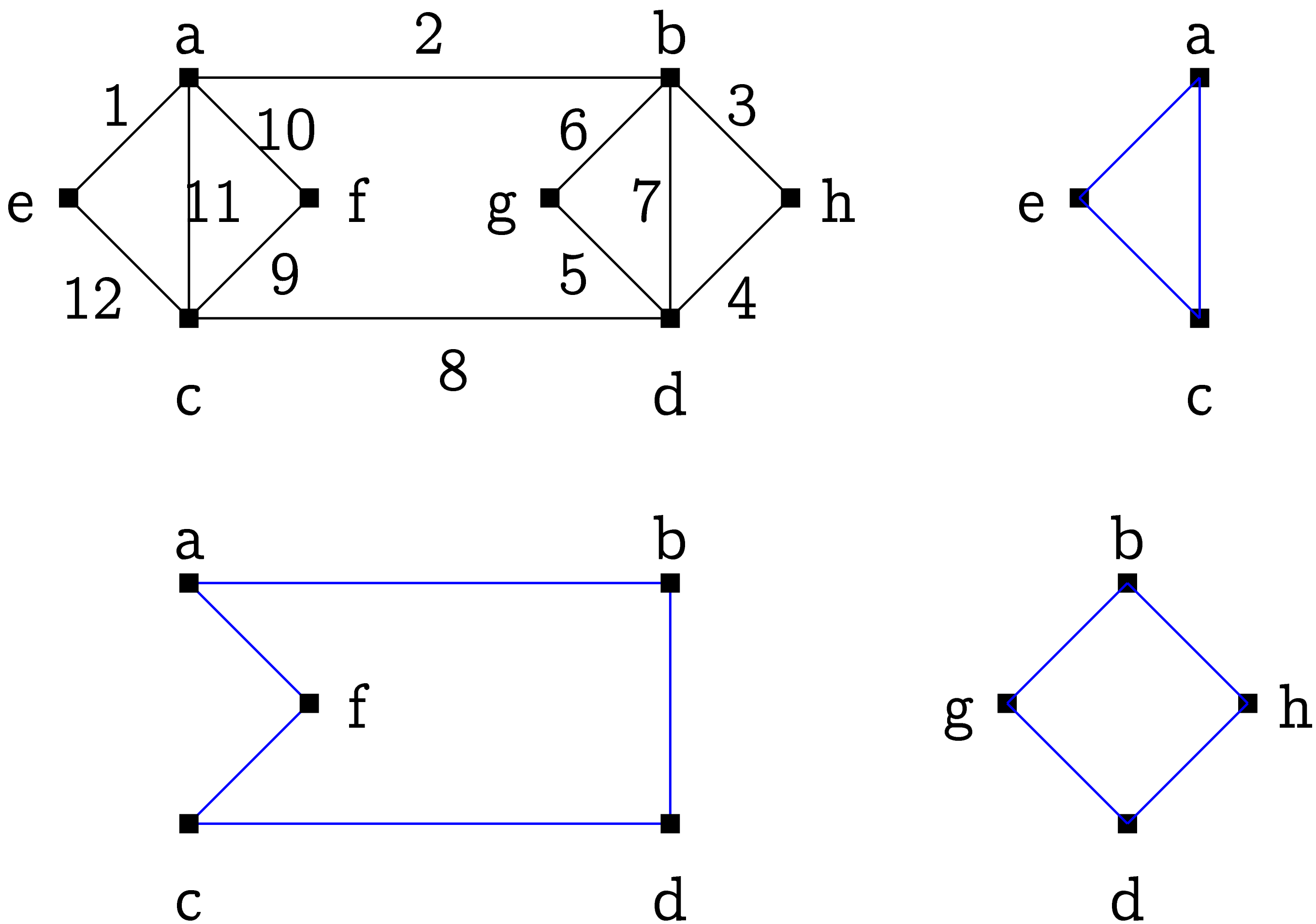

a
b
a
e
f
g
h
e
c
d
c
a
b
b
f
g
h
c
d
d



a
b
a
e
f
g
h
e
c
d
c
a
b
b
f
g
h
c
d
d


a
1
e
2
f
3
c
b
g
h
d
a
e
c
a
b
f
c
d
b
g
h
d


a
2
b
1
6
e
7
f
g
3
h
8
5
c
4
d
a
e
c
a
b
f
c
d
b
g
h
d



a
b
e
f
g
h
c
d
1
2
3
4
5
6
7
8
9
10
11
12
a
e
c
a
b
f
c
d
b
g
h
d

Corollary. Connected graph G is semi-Eulerian $\Leftrightarrow$ the graph G has exactly two vertices with odd degree.

Proof $\Rightarrow$. Let $x \overset{P}{\rightsquigarrow} y$ be a walk in G covering each of the edges of G exactly once.

Add an edge e to G so that $\mathcal{E}(e) = \{x, y\}$.

The graph we obtain is Eulerian ($x \overset{P}{\rightsquigarrow} y \overset{e}{\text{---}} x$ is a Eulerian walk), thus all the vertex degrees are even.

Hence in the original graph x and y have odd degree and all the other vertices have even degrees.

Proof $\Leftarrow$. Let x and y be the two vertices of G having odd degree.

Add an edge e to G so that $\mathcal{E}(e) = \{x, y\}$.

As a result, all the vertex degrees become even, thus there exists a Eulerian walk P.

W.l.o.g assume that the last edge in this walk is e. Removing it from P we obtain the required walk. $\square$

The proof gives an algorithm for finding such a walk:

> Add an additional edge e, find the Eulerian walk and then drop e from it.

Fleury's algorithm for finding a Eulerian walk in Eulerian graph $G = (V, E)$:

1. Pick any vertex $u \in V$ as the first one in the walk. Let $i := 0$ and $v_0 := u$.

2. Pick an edge e incident with vertex v_i, add it to the walk and delete it from the graph G. Let v_{i+1} be the other endpoint of e and let $i := i + 1$.

   - If e is a bridge, pick it *only* if there is no other alternative.

3. Repeat the last step until all the edges are deleted.

Theorem. Fleury's algorithm is correct (i.e. it will always run successfully and produce a Eulerian walk).

Proof. The algorithm produces some walk P starting from u. At some point it stops, because it reaches a vertex v_n, that has all the incident edges deleted. Considering the vertex degrees, it is obvious that $v_n = u$.

We have to show that at that moment all the edges are deleted.

Let G_i be the graph remaining of G after step i. Then $G_0 = G$ and G_{i+1} contains one edge less than the graph G_i. Let H_i be the connected component of G_i containing the vertex u.

Note that the degrees of all the vertices of G_i (except for, possibly, u and v_i) are even. If $u = v_i$ then also $\deg(u)$ is even. If $u \neq v_i$ then $\deg(u)$ and $\deg(v_i)$ are odd.

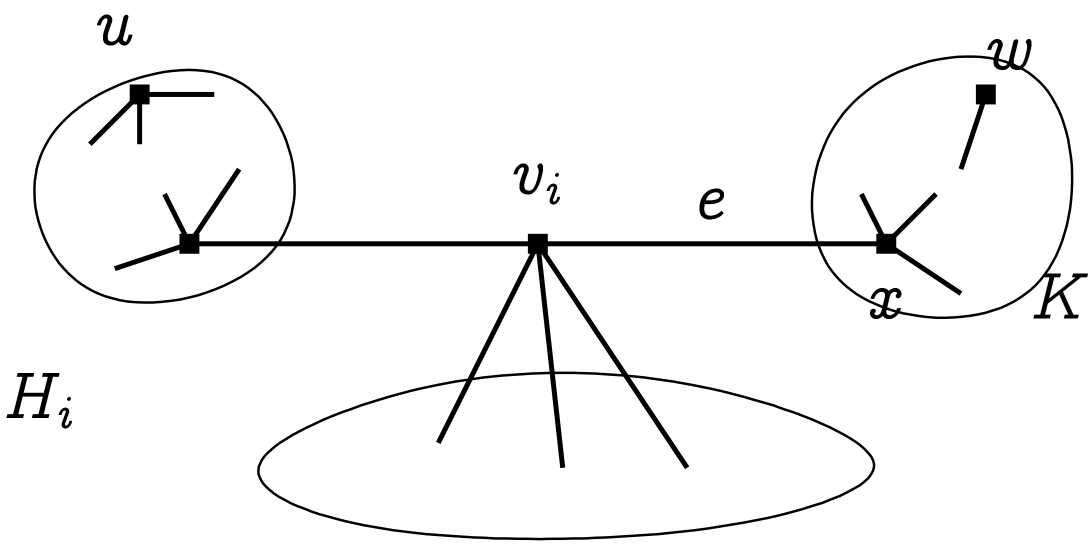
We will show that all the remaining connected components of G_i are isolated vertices.

We will use induction over i. If $i = 0$ then $G_0 = G = H_0$, and G_0 has only one connected component, thus the claim holds.

Let the claim hold for G_i. Consider first the case $u \neq v_i$. In order to give the proof for G_{i+1}, it is enough to prove that there is at most one bridge incident with v_i in the graph G_i.

- If so, then we are done, because the connected components of G_{i+1} are the following.
  - If we deleted a non-bridge, the connected components did not change.
  - If we deleted a bridge, it was the last edge incident with v_i. The component H_i is divided into two new components – v and $H_{i+1} = H_i \backslash v$. The first one is an isolated vertex, the second one contains vertex u.

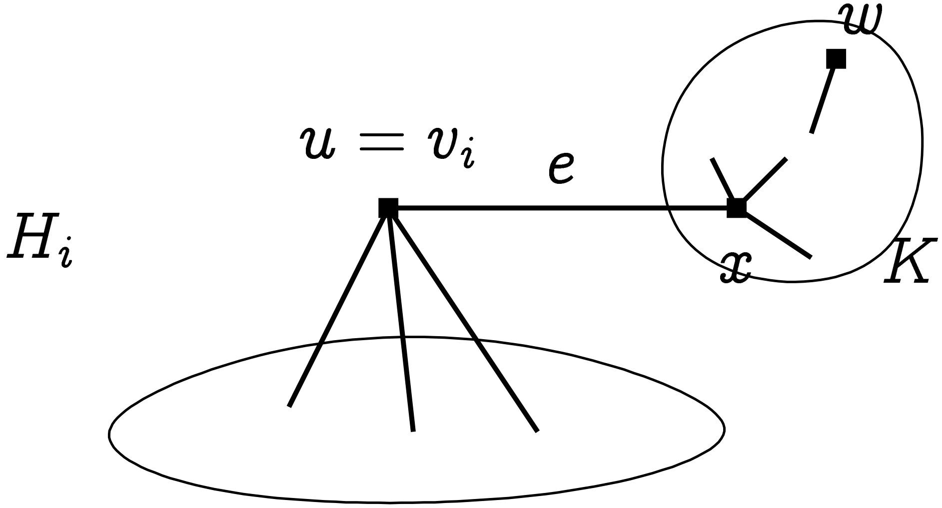
If at least two bridges were incident to v_i then:



- There exists an edge e incident to v_i such that the connected component of $H_i - e$ not containing v_i does not contain u either.
- $\deg_{H_i}(x)$ is even. Thus $\deg_K(x)$ is odd.
- There has to exist another vertex w of K so that $\deg_K(w)$ is odd. At the same time, $\deg_K(w) = \deg_{H_i}(w)$ an this had to be even.

If $u = v_i$, it is enough to show that there are no bridges incident with u, i.e. G_i and G_{i+1} have the same connected components.

If u would have an incident bridge,



there would again exist a vertex w with odd degree. $\square$

Let the edges of the graph $G = (V, E, \mathcal{E})$ have non-negative weights ("lengths").

Let the function $w : E \longrightarrow \mathbb{R}^+$ give the lengths.

If $P = \cdot \xrightarrow{e_1} \cdot \xrightarrow{e_2} \cdots \xrightarrow{e_k} \cdot$ is a walk then let $w(P) := \sum_{i=1}^{k} w(k)$ be its length.

Chinese postman problem (Hiina postiljoniprobleem) (CPP): find the closed walk of minimum length that passes each edge *at least* once.

Obviously, if G is Eulerian the the solution to CPP is any Eulerian walk.

Tasks that reduce to CPP (or its variants):

- Routing postmen, garbage trucks, snowplows, etc.
- Checking the transportation routes (highways, railways, power lines, etc.)
- Optimizing the testing strategies of state automata (e.g. UIs)
  - A test: does the system in state A go to state B after the action s?

Let a *pseudo-Eulerian walk* be a closed walk that passes through all edges of a graph at least once.

CPP is looking for a pseudo-Eulerian walk of minimum length.

Let P be a pseudo-Eulerian walk in the graph G. Define the graph $G_P = (V, E_P, \mathcal{E}_P)$ as follows:

- $E_P = \{e^{(i)} \mid 1 \le i \le |P|_e\}$,
- $\mathcal{E}_P(e^{(i)}) = \mathcal{E}(e)$,

where $|P|_e$ is the number of occurrences of e in P.

Proposition. G_P is an Eulerian graph for any graph G and pseudo-Eulerian walk P.

Proof. Replace the i-th occurrence of an edge e in P with $e^{(i)}$. This gives an Eulerian walk in G_P. $\square$

In the other direction, let $c : E \longrightarrow \mathbb{N}$. Define $G_c = (V, E_c, \mathcal{E}_c)$, as follows:

- $E_c = \{e^{(i)} \mid 1 \leq i \leq c(e)\}$,
- $\mathcal{E}_P(e^{(i)}) = \mathcal{E}(e)$,

If $c(e) > 0$ for all $e \in E$ and G_c is an Eulerian graph then each Eulerian walk in G_c defines a pseudo-Eulerian walk in G.

The lengths of all pseudo-Eulerian walks resulting from G_c are equal.

- they equal $\sum_{e \in E} c(e)w(e)$.

Proposition. In the solution to CPP, no edge occurs more than twice.

Proof. Let P be the solution to CPP in $G = (V, E, \mathcal{E})$. Assume the opposite: $\exists e \in E$, such that $n = |P|_e \geq 3$.

Consider the graph G_P. It is an Eulerian graph.

Remove $e^{(n-1)}$ and $e^{(n)}$ from G_P, giving G_c. It is still an Eulerian graph and $e^{(1)} \in E(G_c)$.

For all $e \in G$, G_c contains at least one copy of G. Hence an Eulerian walk in G_c is a pseudo-Eulerian walk in G. The cost of such a walk is $w(P) - 2w(e) \leq w(P)$. $\square$

A generalization:...

Proposition. Let P be a solution to CPP in $G = (V, E, \mathcal{E})$. Let $c(e) = |P|_e - 1$. Then G_c does not contain cycles.

Proof. Assume that the graph G_c contains a cycle C. Let $c'(e) = |P|_e - |C|_e$. Then $c'(e) > 0$ for any $e \in E$.

$G_{c'}$ is an Eulerian graph, giving pseudo-Eulerian walks in G with the cost $w(P) - w(C)$. $\square$

Theorem. Let $G = (V, E, \mathcal{E})$ a graph and let $V^- \subseteq V$ be the set of vertices of odd degree in G. The set V^- can be partitioned to pairs $V^- = \{u_1, v_1\} \dot\cup \{u_2, v_2\} \dot\cup \cdots \dot\cup \{u_n, v_n\}$;

- (let P_i be the shortest path from u_i to v_i)

such that an edge occurs twice in a CPP solution P for G iff this edge belongs to one of $P_1, \ldots, P_n$.

In other words, the edges of G_c (from the previous proposition) are made up of $P_1, \ldots, P_n$.

Proof. Consider this graph G_c.

Then $\deg_G(v) \equiv \deg_{G_c}(v) \pmod 2$ for any $v \in V$, because $\deg_{G_c}(v) = \deg_{G_P}(v) - \deg_G(v)$ and G_P is Eulerian.

Let $G_0 = G_c$ and $n = |V^-|/2$. For all $i \in \{1, \ldots, n\}$ define

- let $u_i, v_i \in V$ be two vertices of odd degree in the same connected component of G_{i-1};
- let P_i be a path from u_i to v_i in G_{i-1};
- let G_i be a graph obtained from G_{i-1} be removing from it the edges of P_i.

In G_i, the degrees of u_i and v_i are even and the parity of degrees of other vertices did not change from G_{i-1}.

In G_n, all vertices have even degree.

Consider a connected component of G_n. If it is not an isolated vertex, then it contains a cycle. The same cycle exists in G_c. This contradicts the last proposition. Hence G_n contains no edges.

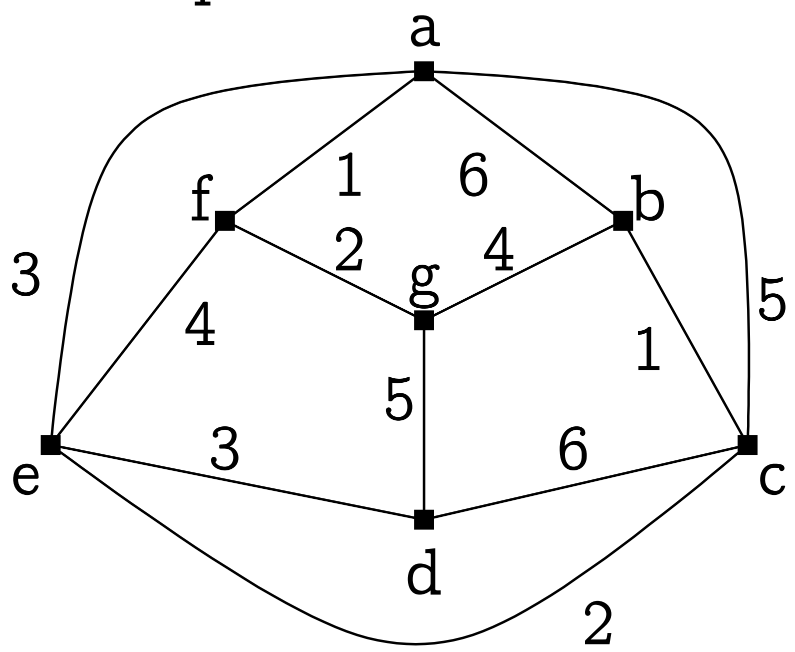
We have partitioned the edges of G_c to n paths.

P is the solution to CPP, hence these paths must be of minimal length between their endpoints. $\square$

Algorithm for solving CPP in the graph $G = (V, E, \mathcal{E})$:

1. Find the pairwise distances between all vertices in $V^- \subseteq V$.
   - It makes sense to use e.g. Floyd-Warshall algorithm to find the pairwise distances between all vertices.
   - Find the corresponding shortest paths, too.
2. Partition V^- to pairs $\{u_i, v_i\}$ in such a way, that the summary length of distances between u_i and v_i is as small as possible.
   - This can be done in polynomial time.
   - We might see an algorithm in one of the following lectures.
3. Augment G with a copy of edges on some of the shortest paths between u_i and v_i. Find an Eulerian walk in the resulting graph.

Example: Distances between vertices of odd degree:



	b	d	f	g
b	X	6	6	4
d	6	X	7	5
f	6	7	X	2
g	4	5	2	X

The pairs $\{b, d\}$ and $\{f, g\}$ give the minimum summary length.

The solution to CPP is an Eulerian walk in the graph