Department of
Computer Science and Engineering
计算机科学与工程系

CS215 DISCRETE MATH

Dr. Qi Wang

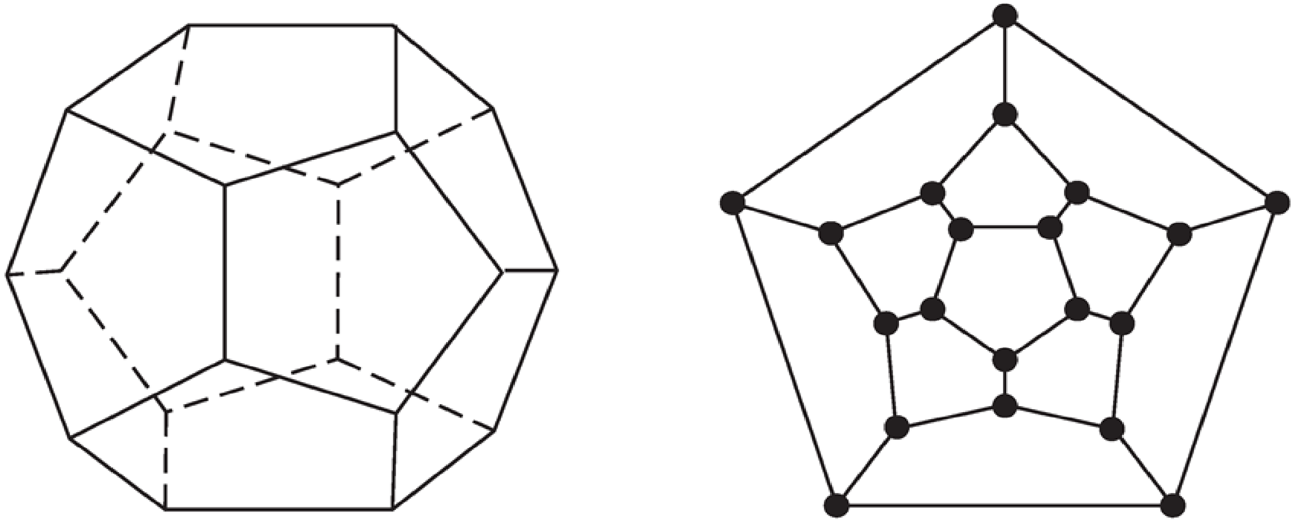
Department of Computer Science and Engineering
Office: Room413, CoE South Tower
Email: wangqi@sustech.edu.cn

Hamilton Paths and Circuits

- Euler paths and circuits contained every edge only once. What about containing every vertex exactly once?

Hamilton Paths and Circuits

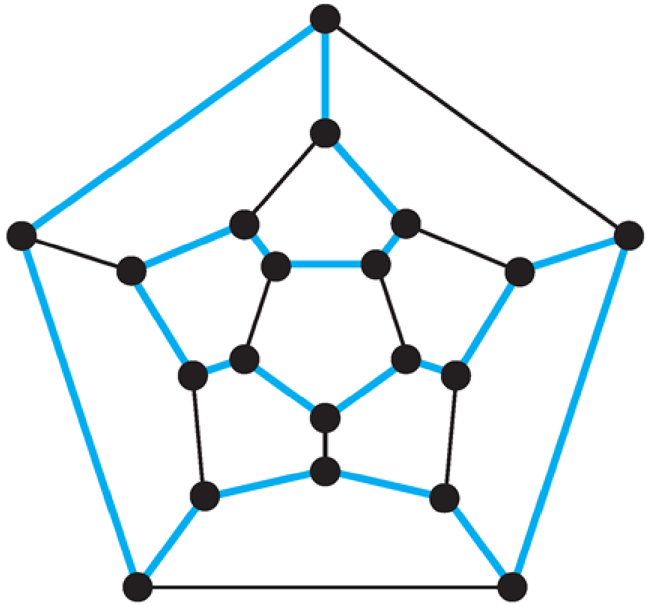
- Euler paths and circuits contained every edge only once. What about containing every vertex exactly once?



(a) (b)

Hamilton Paths and Circuits

- Euler paths and circuits contained every edge only once. What about containing every vertex exactly once?

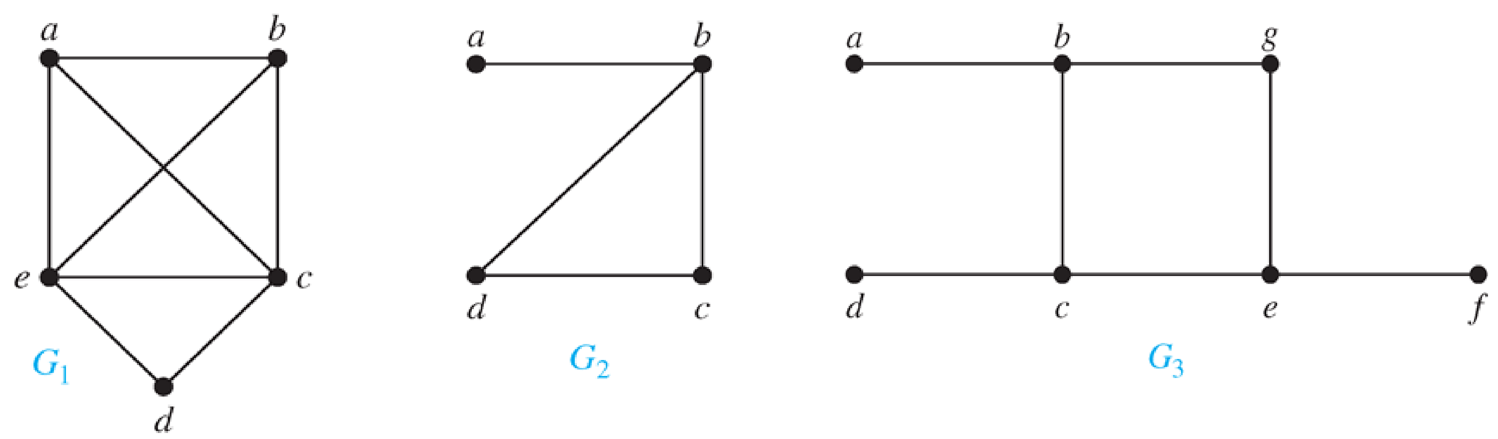


Hamilton Paths and Circuits

- **Definition**: A simple path in a graph G that passes through every vertex exactly once is called a *Hamilton path*, and a simple circuit in a graph G that passes through every vertex exactly once is called a *Hamilton circuit*.

- **Definition**: A simple path in a graph G that passes through every vertex exactly once is called a *Hamilton path*, and a simple circuit in a graph G that passes through every vertex exactly once is called a *Hamilton circuit*.

Example Which of these simple graphs has a Hamilton circuit or, if not, a Hamilton path?



Sufficient Conditions for Hamilton Circuits

- No simple necessary and sufficient conditions are known for the existence of a Hamilton circuit.

Sufficient Conditions for Hamilton Circuits

- No simple necessary and sufficient conditions are known for the existence of a Hamilton circuit.

  But, there are some useful sufficient conditions.

Sufficient Conditions for Hamilton Circuits

- No simple necessary and sufficient conditions are known for the existence of a Hamilton circuit.

  But, there are some useful sufficient conditions.

  Dirac's Theorem If G is a simple graph with $n \geq 3$ vertices such that the degree of every vertex in G is $\geq n/2$, then G has a Hamilton circuit.

每个顶点的degree都足够大，n/2

Sufficient Conditions for Hamilton Circuits

- No simple necessary and sufficient conditions are known for the existence of a Hamilton circuit.

  But, there are some useful sufficient conditions.

  Dirac's Theorem If G is a simple graph with $n \geq 3$ vertices such that the degree of every vertex in G is $\geq n/2$, then G has a Hamilton circuit.

  Ore's Theorem If G is a simple graph with $n \geq 3$ vertices such that $\deg(u) + \deg(v) \geq n$ for every pair of nonadjacent vertices, then G has a Hamilton circuit.

每一对不相邻的顶点的degree和>=n

Sufficient Conditions for Hamilton Circuits

- No simple necessary and sufficient conditions are known for the existence of a Hamilton circuit.

  But, there are some useful sufficient conditions.

  Dirac's Theorem If G is a simple graph with $n \geq 3$ vertices such that the degree of every vertex in G is $\geq n/2$, then G has a Hamilton circuit.

  Ore's Theorem If G is a simple graph with $n \geq 3$ vertices such that $\deg(u) + \deg(v) \geq n$ for every pair of nonadjacent vertices, then G has a Hamilton circuit.

  Hamilton path problem $\in$ NPC

Applications of Hamilton Paths and Circuits

- A path or a circuit that visits each city, or each node in a communication network exactly once, can be solved by finding a Hamilton path.

Applications of Hamilton Paths and Circuits

- A path or a circuit that visits each city, or each node in a communication network exactly once, can be solved by finding a Hamilton path.

  Traveling Salesperson Problem (TSP) asks for the shortest route a traveling salesperson should take to visit a set of cities.

Applications of Hamilton Paths and Circuits

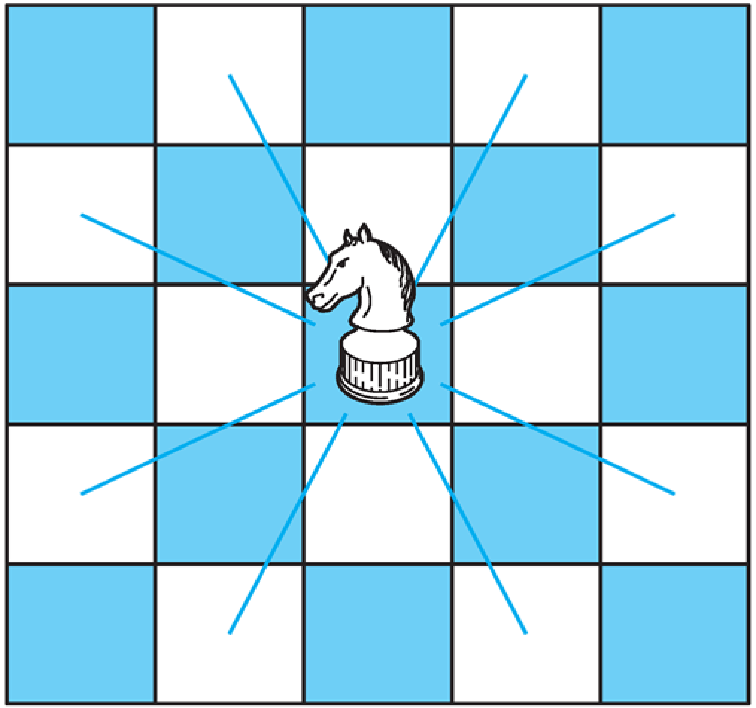
- A path or a circuit that visits each city, or each node in a communication network exactly once, can be solved by finding a Hamilton path.

  Traveling Salesperson Problem (TSP) asks for the shortest route a traveling salesperson should take to visit a set of cities.

  the decision version of the TSP $\in$ NPC

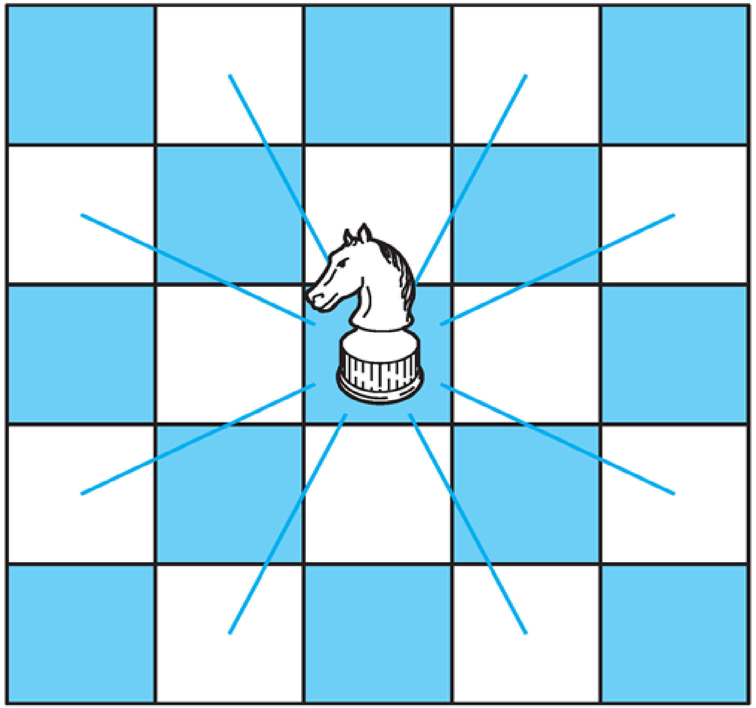
Applications of Hamilton Paths and Circuits

- Can we traverse every space (and come back) in the 5×5 chessboard?



Applications of Hamilton Paths and Circuits

- Can we traverse every space (and come back) in the 5×5 chessboard?



What about in 6×6 chessboard?

Shortest Path Problems

- Using graphs with weights assigned to their edges

Shortest Path Problems

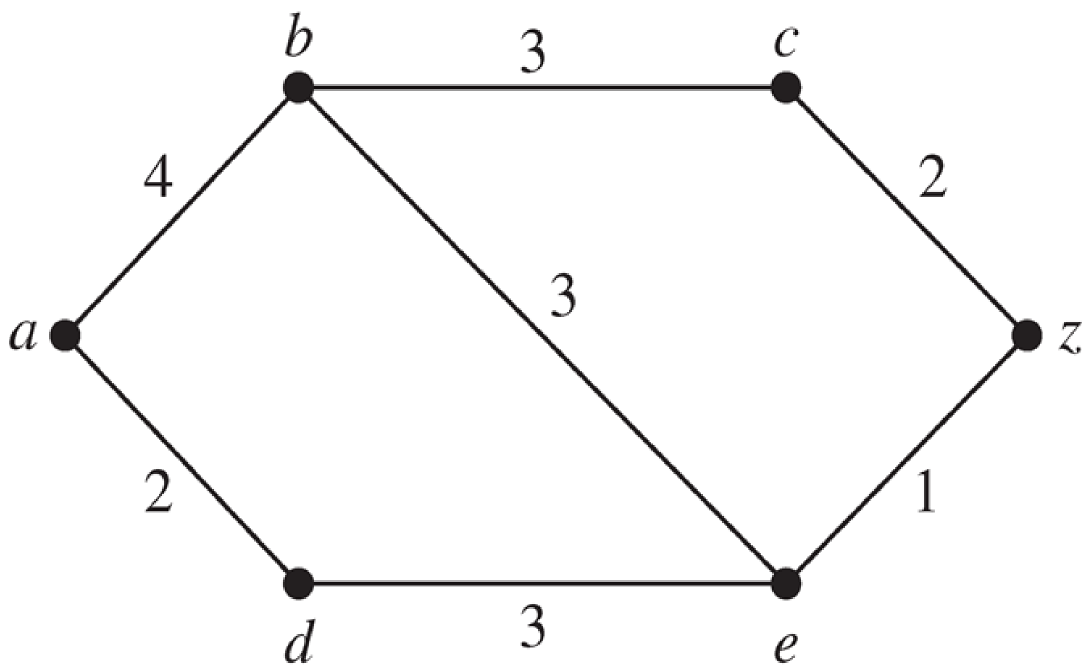
- Using graphs with weights assigned to their edges

  Such graphs are called *weighted graphs* and can model lots of questions involving distance, time consuming, fares, etc.

Shortest Path Problems

- Using graphs with weights assigned to their edges

  Such graphs are called *weighted graphs* and can model lots of questions involving distance, time consuming, fares, etc.



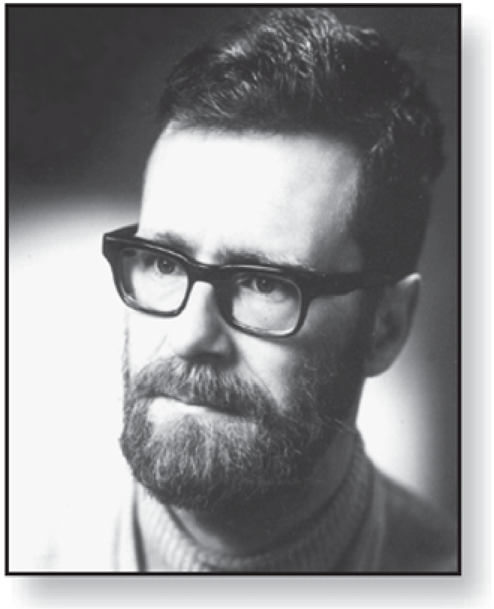
- **Definition** Let G^α be an weighted graph, with a weight function $\alpha : E \to \mathbf{R}^+$ on its edges. If $P = e_1 e_2 \cdots e_k$ is a path, then its weight is $\alpha(P) = \sum_{i=1}^{k} \alpha(e_i)$. The *minimum weighted distance* between two vertices is

$$d(u, v) = \min\{\alpha(P) | P : u \to v\}$$

Shortest Path Problems

- **Definition** Let G^α be an weighted graph, with a weight function $\alpha : E \to \mathbf{R}^+$ on its edges. If $P = e_1 e_2 \cdots e_k$ is a path, then its weight is $\alpha(P) = \sum_{i=1}^{k} \alpha(e_i)$. The *minimum weighted distance* between two vertices is

$$d(u, v) = \min\{\alpha(P) | P : u \to v\}$$



Edsger Wybe Dijkstra

Dijkstra's Algorithm

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

Dijkstra's Algorithm

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $$\min\{d(u), d(v) + \alpha(u, v)\}$$

Dijkstra's Algorithm

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $\min\{d(u), d(v) + \alpha(u, v)\}$

  (iii) return all $d(v)$'s

Dijkstra's Algorithm

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

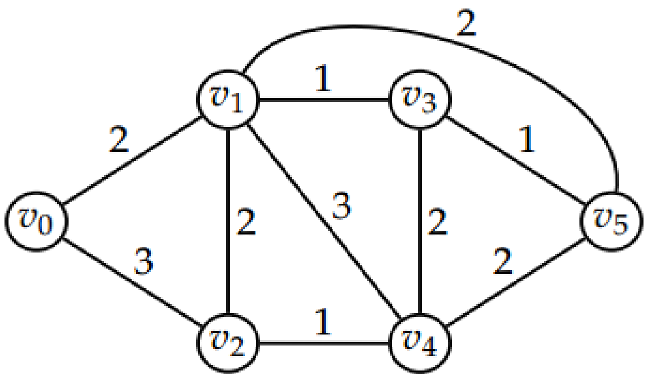
  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $$\min\{d(u), d(v) + \alpha(u, v)\}$$

  (iii) return all $d(v)$'s

  Example



- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

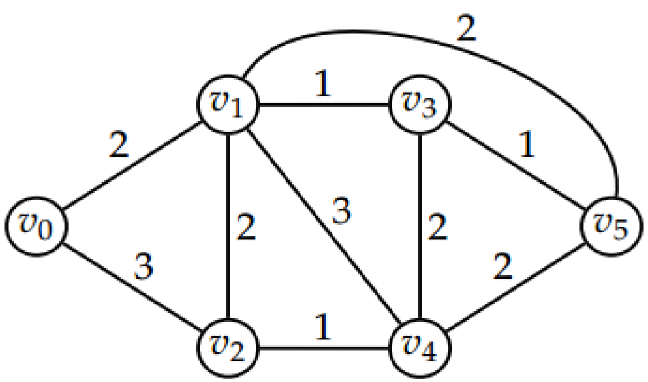
  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $$\min\{d(u), d(v) + \alpha(u, v)\}$$

  (iii) return all $d(v)$'s

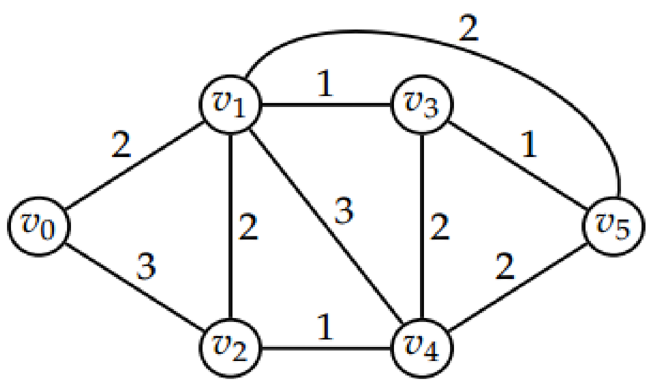
  Example



$d(v_0) = 0$, all other $d(v) = \infty$

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$
  (ii) while $S \neq V$
  let $v \notin S$ be the vertex with the least value $d(v)$,
  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by
  $$\min\{d(u), d(v) + \alpha(u, v)\}$$
  (iii) return all $d(v)$'s

Example



v_0	v_1	v_2	v_3	v_4	v_5
0	∞	∞	∞	∞	∞

$d(v_0) = 0$, all other $d(v) = \infty$

Dijkstra's Algorithm

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

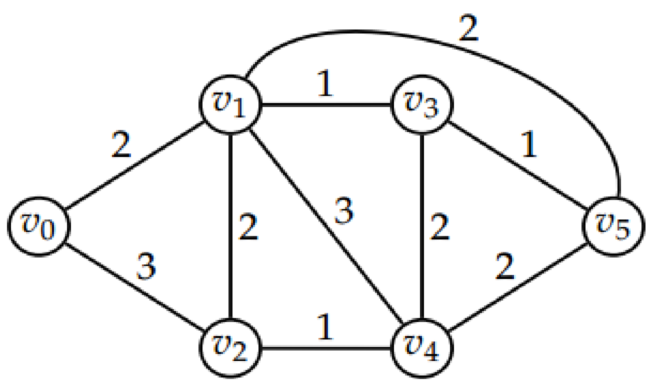
  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $$\min\{d(u), d(v) + \alpha(u, v)\}$$

  (iii) return all $d(v)$'s

Example



v_0	v_1	v_2	v_3	v_4	v_5
0	∞	∞	∞	∞	∞

$i = 0$

$d(v_1) = \min\{\infty, 2\} = 2$, $d(v_2) = \min\{\infty, 3\} = 3$

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

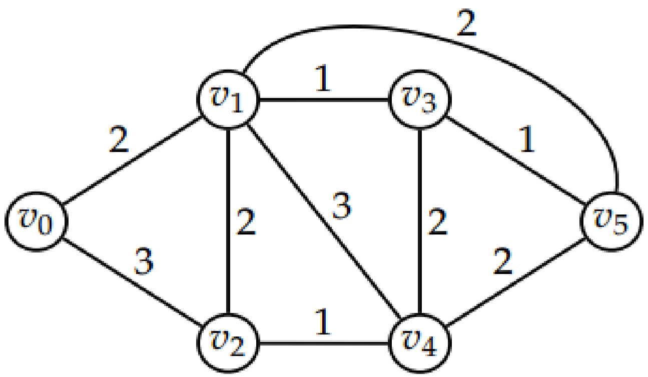
  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $$\min\{d(u), d(v) + \alpha(u, v)\}$$

  (iii) return all $d(v)$'s

  Example



v_0	v_1	v_2	v_3	v_4	v_5
0	2	3	∞	∞	∞

$i = 0$

$d(v_1) = \min\{\infty, 2\} = 2$, $d(v_2) = \min\{\infty, 3\} = 3$

Dijkstra's Algorithm

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

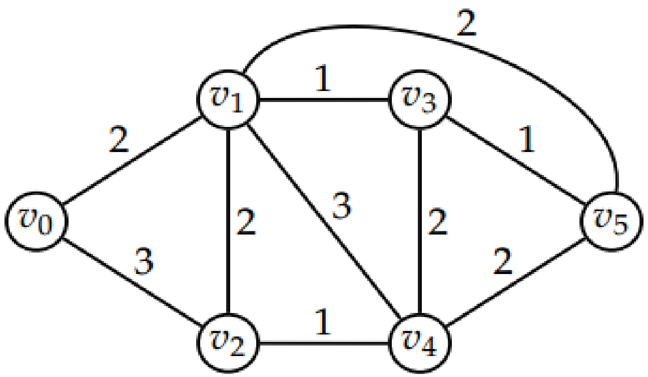
  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $$\min\{d(u), d(v) + \alpha(u, v)\}$$

  (iii) return all $d(v)$'s

Example



v_0	v_1	v_2	v_3	v_4	v_5
0	2	3	∞	∞	∞

$i = 1$

$d(v_2) = \min\{3, d(v_1) + \alpha(v_1 v_2)\} = \min\{3, 4\} = 3$,

$d(v_3) = 2 + 1 = 3$, $d(v_4) = 2 + 3 = 5$,

$d(v_5) = 2 + 2 = 4$

Dijkstra's Algorithm

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

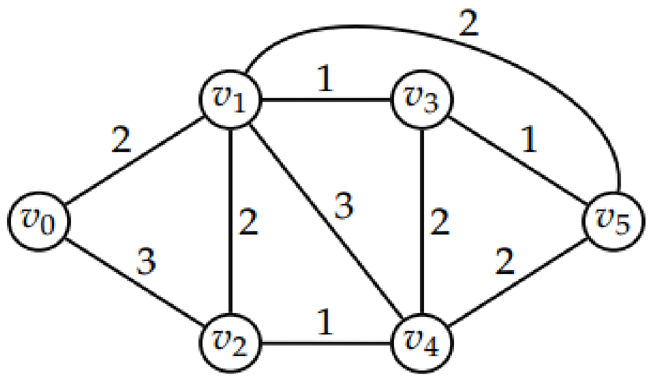
  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $$\min\{d(u), d(v) + \alpha(u, v)\}$$

  (iii) return all $d(v)$'s

Example



v_0	v_1	v_2	v_3	v_4	v_5
0	2	3	3	5	4

$i = 1$

$d(v_2) = \min\{3, d(v_1) + \alpha(v_1 v_2)\} = \min\{3, 4\} = 3$,

$d(v_3) = 2 + 1 = 3$, $d(v_4) = 2 + 3 = 5$,

$d(v_5) = 2 + 2 = 4$

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

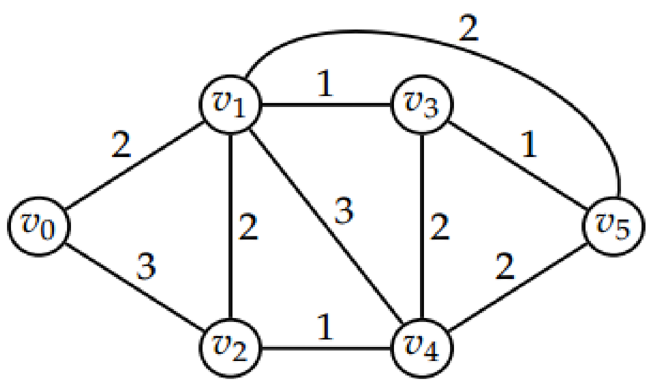
  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $$\min\{d(u), d(v) + \alpha(u, v)\}$$

  (iii) return all $d(v)$'s

Example



v_0	v_1	v_2	v_3	v_4	v_5
0	2	3	3	5	4

$i = 2$

$d(v_3) = \min\{3, \infty\} = 3,$

$d(v_4) = \min\{5, 3 + 1\} = 4,$

$d(v_5) = \min\{4, \infty\} = 4$

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

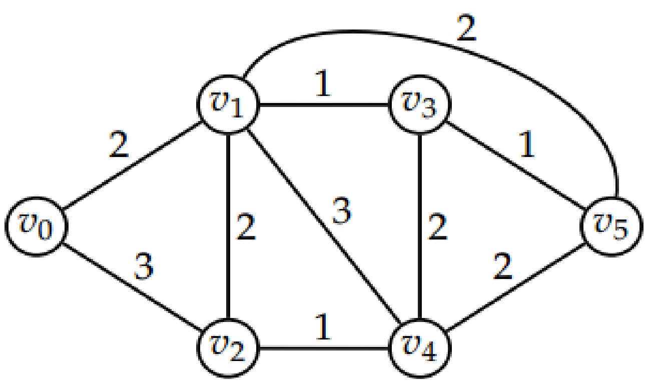
  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $$\min\{d(u), d(v) + \alpha(u, v)\}$$

  (iii) return all $d(v)$'s

  Example



v_0	v_1	v_2	v_3	v_4	v_5
0	2	3	3	4	4

$i = 2$

$d(v_3) = \min\{3, \infty\} = 3$,

$d(v_4) = \min\{5, 3 + 1\} = 4$,

$d(v_5) = \min\{4, \infty\} = 4$

Dijkstra's Algorithm

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

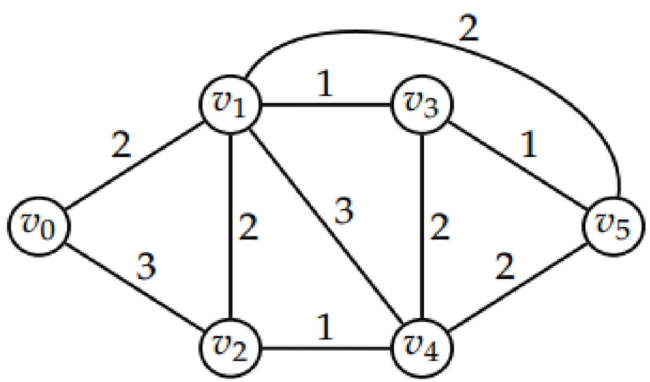
  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $$\min\{d(u), d(v) + \alpha(u, v)\}$$

  (iii) return all $d(v)$'s

Example



v_0	v_1	v_2	v_3	v_4	v_5
0	2	3	3	4	4

$i = 3$

$d(v_4) = \min\{4, 3 + 2\} = 4,$

$d(v_5) = \min\{4, 3 + 1\} = 4$

- (i) Set $d(v_0) = 0$ and $d(v) = \infty$ for all $v \neq v_0$, $S = \emptyset$

  (ii) while $S \neq V$

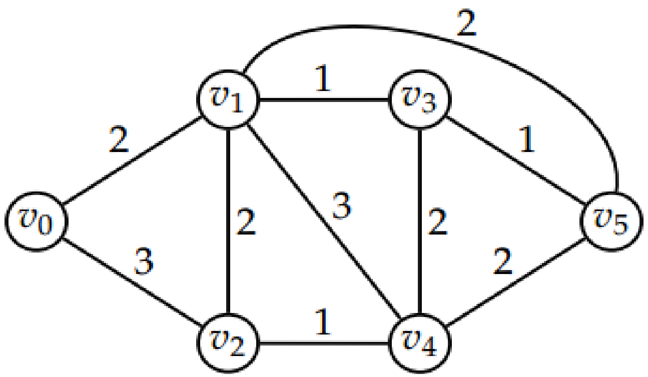
  let $v \notin S$ be the vertex with the least value $d(v)$,

  $S = S \cup \{v\}$ for each $u \notin S$, replace $d(u)$ by

  $$\min\{d(u), d(v) + \alpha(u, v)\}$$

  (iii) return all $d(v)$'s

Example



v_0	v_1	v_2	v_3	v_4	v_5
0	2	3	3	4	4

$i = 4$

$d(v_5) = \min\{4, 4 + 2\} = 4$

Dijkstra's Algorithm

- **Theorem** Dijkstra's algorithm finds the length of a shortest path between two vertices in a connnected simple undirected weighted graph.

- **Theorem** Dijkstra's algorithm finds the length of a shortest path between two vertices in a connnected simple undirected weighted graph.

  Correctness

Dijkstra's Algorithm

- **Theorem** Dijkstra's algorithm finds the length of a shortest path between two vertices in a connnected simple undirected weighted graph.

  Correctness

  Theorem Dijkstra's algorithm uses $O(n^2)$ operations (additions and comparisons) in a connected simple undirected weighted graph with n vertices.

Dijkstra's Algorithm

- **Theorem** Dijkstra's algorithm finds the length of a shortest path between two vertices in a connnected simple undirected weighted graph.

  Correctness

  Theorem Dijkstra's algorithm uses $O(n^2)$ operations (additions and comparisons) in a connected simple undirected weighted graph with n vertices.

  Complexity

Dijkstra's Algorithm

- **Theorem** Dijkstra's algorithm finds the length of a shortest path between two vertices in a connnected simple undirected weighted graph.

  Correctness

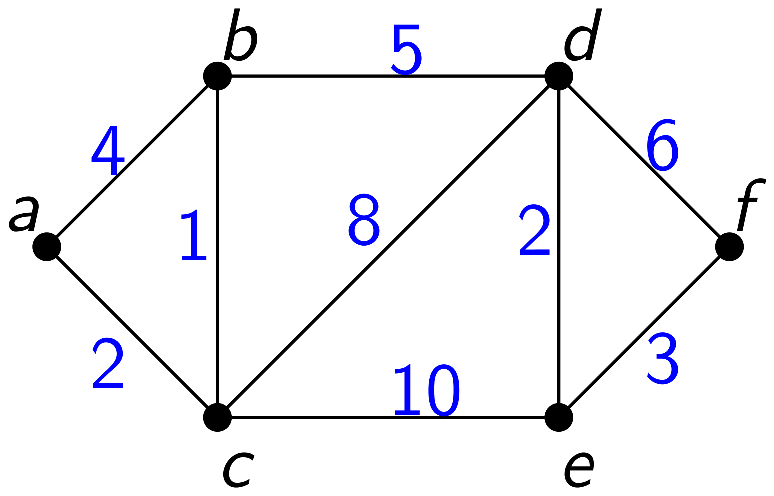
  Theorem Dijkstra's algorithm uses $O(n^2)$ operations (additions and comparisons) in a connected simple undirected weighted graph with n vertices.

  Complexity

  read the Textbook p.712 – p.714

Department of Computer Science and Engineering
南方科技大学计算机科学与工程系

Another Example



The Single-Source Shortest Paths (SSSP) Problem

- Dijkstra's algorithm

  $O(v^2)$ using an *array* [Dijkstra 1956]

  $O(e + v \log v)$ using a *Fibonacci heap min-priority queue* [Fredman & Tarjan 1984]

The Single-Source Shortest Paths (SSSP) Problem

- Dijkstra's algorithm

  $O(v^2)$ using an *array* [Dijkstra 1956]

  $O(e + v \log v)$ using a *Fibonacci heap min-priority queue* [Fredman & Tarjan 1984]

  Only for *nonnegative* edge weights!

The Single-Source Shortest Paths (SSSP) Problem

- Dijkstra's algorithm

  $O(v^2)$ using an *array* [Dijkstra 1956]

  $O(e + v \log v)$ using a *Fibonacci heap min-priority queue*

  [Fredman & Tarjan 1984]

  Only for *nonnegative* edge weights!

- Bellman-Ford algorithm [Ford 1956] [Bellman 1958]

  Works for negative weights; no negative-weight cycle reachable from s

  $O(ev)$

The Single-Source Shortest Paths (SSSP) Problem

- Dijkstra's algorithm
  $O(v^2)$ using an *array* [Dijkstra 1956]
  $O(e + v \log v)$ using a *Fibonacci heap min-priority queue*
  [Fredman & Tarjan 1984]

  Only for *nonnegative* edge weights!

- Bellman-Ford algorithm [Ford 1956] [Bellman 1958]
  Works for negative weights; no negative-weight cycle reachable from s
  $O(ev)$

- New result

Negative-Weight Single-Source Shortest Paths in Near-linear Time

Aaron Bernstein* Danupon Nanongkai† Christian Wulff-Nilsen‡

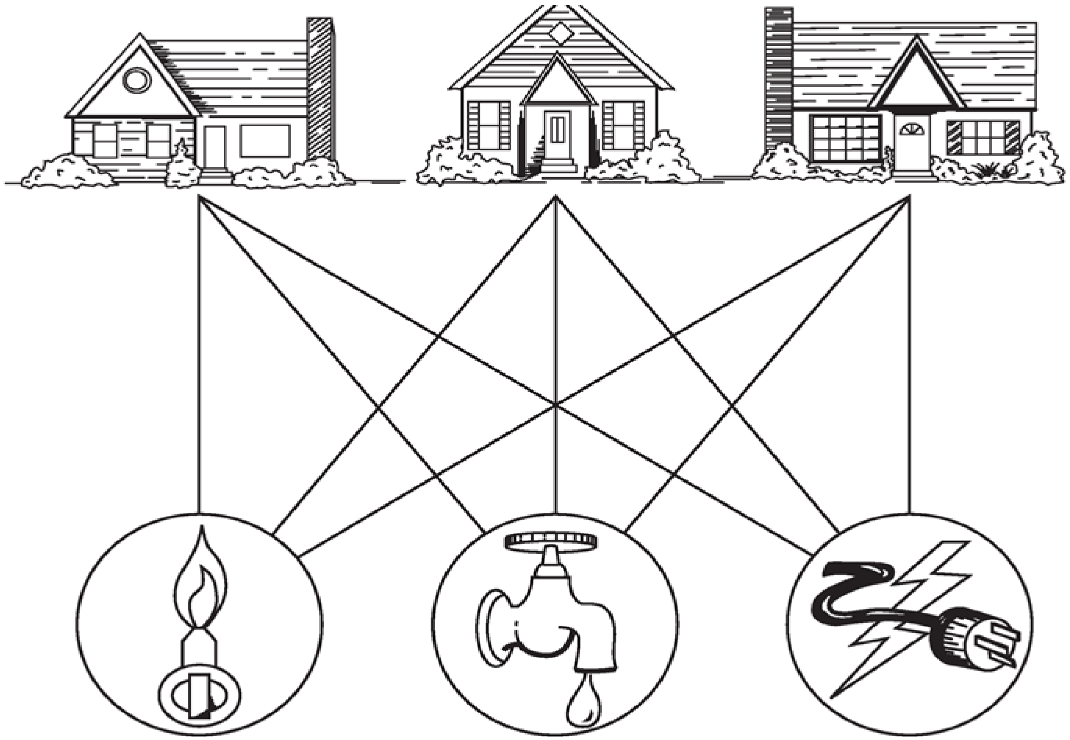
Abstract

We present a randomized algorithm that computes single-source shortest paths (SSSP) in $O(m\log^8(n)\log W)$ time when edge weights are integral and can be negative.[1] This essentially resolves the classic negative-weight SSSP problem. The previous bounds are $\tilde{O}((m+n^{1.5})\log W)$ [BLNPSSSW FOCS'20] and $m^{4/3+o(1)}\log W$ [AMV FOCS'20]. Near-linear time algorithms were known previously only for the special case of planar directed graphs [Fakcharoenphol and Rao FOCS'01].

In contrast to all recent developments that rely on sophisticated continuous optimization methods and dynamic algorithms, our algorithm is simple: it requires only a simple graph decomposition and elementary combinatorial tools. In fact, ours is the first combinatorial algorithm for negative-weight SSSP to break through the classic $\tilde{O}(m\sqrt{n}\log W)$ bound from over three decades ago [Gabow and Tarjan SICOMP'89].

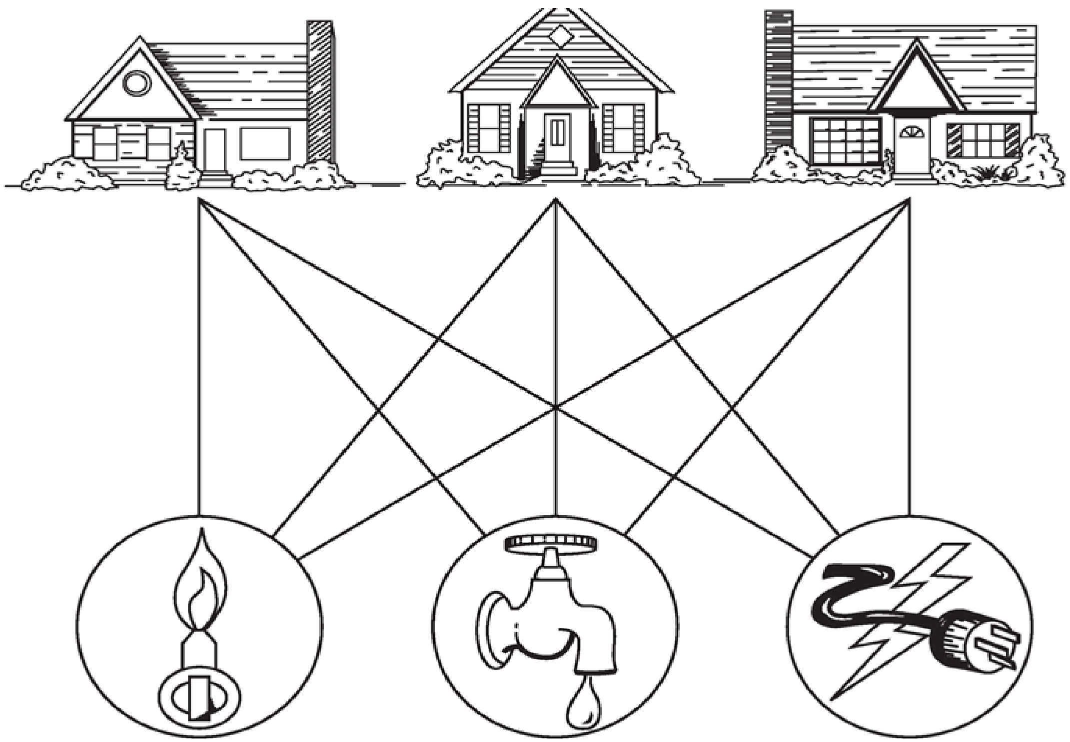
Planar Graphs

- Join three houses to each of three seperate utilities.



Planar Graphs

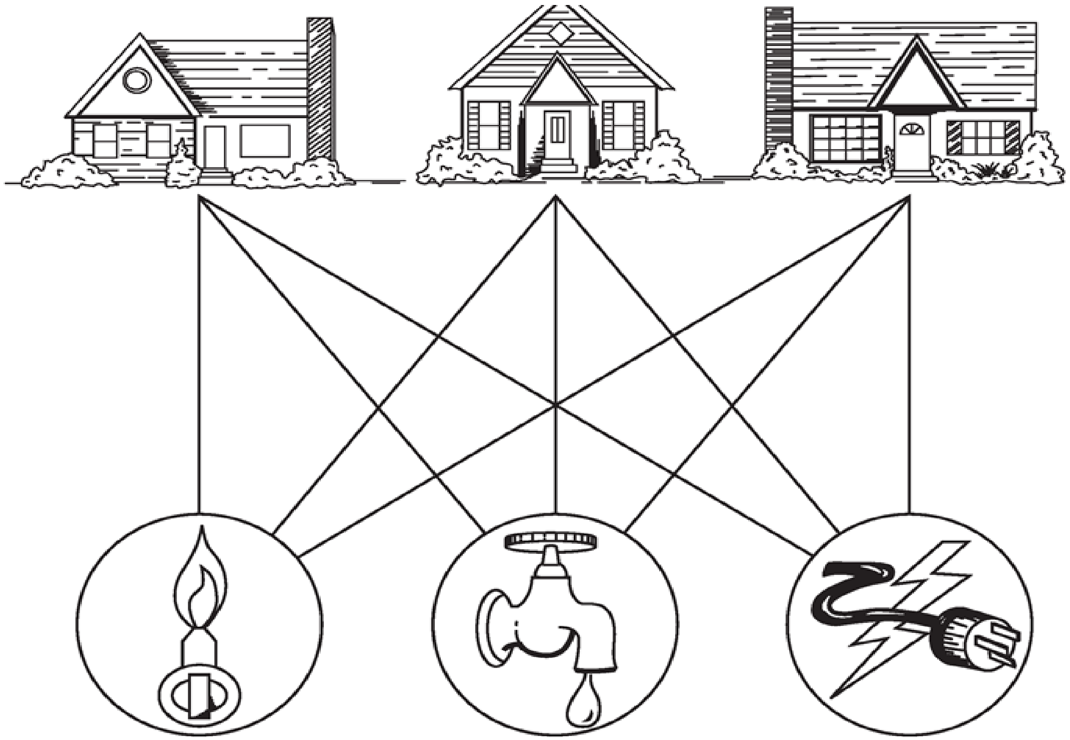
- Join three houses to each of three seperate utilities.



Can this graph be drawn in the plane s.t. no two of its edges cross?

Planar Graphs

- Join three houses to each of three seperate utilities.



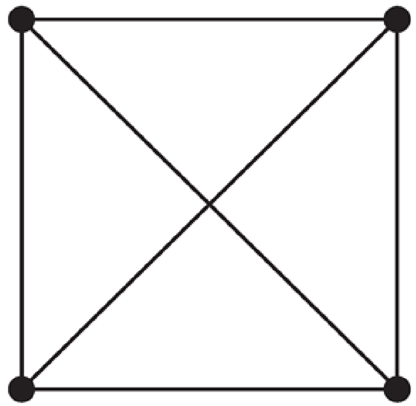
Can this graph be drawn in the plane s.t. no two of its edges cross? $K_{3,3}$

Planar Graphs

- **Definition** A graph is called *planar* if it can be drawn in the plane without any edges crossing. Such a drawing is called a *planar representation* of the graph.

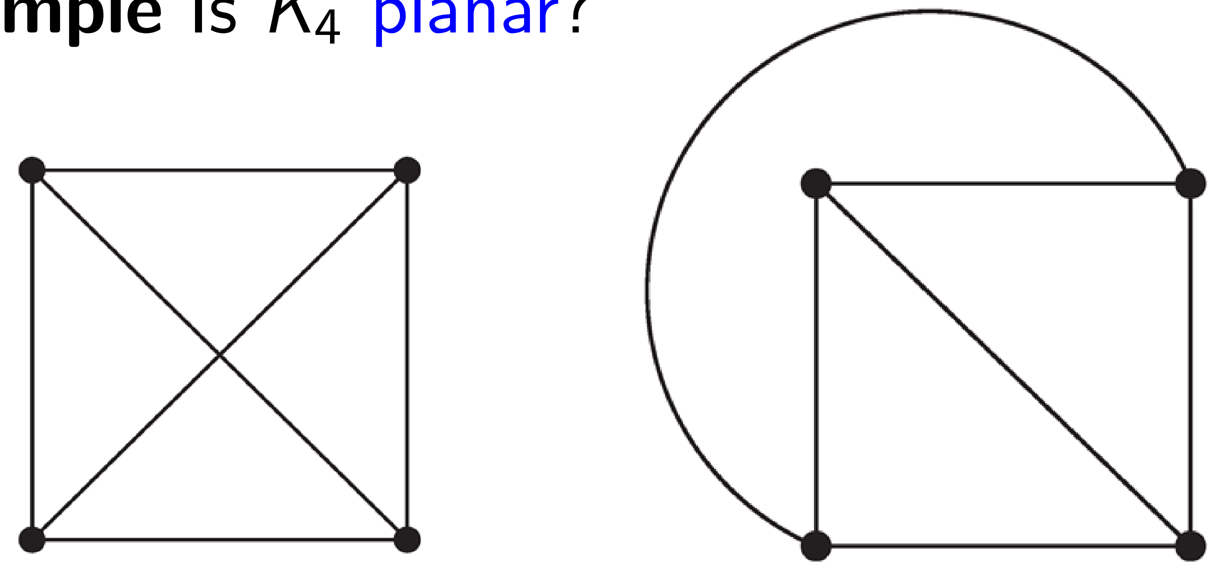
- **Definition** A graph is called *planar* if it can be drawn in the plane without any edges crossing. Such a drawing is called a *planar representation* of the graph.

Example Is K_4 planar?

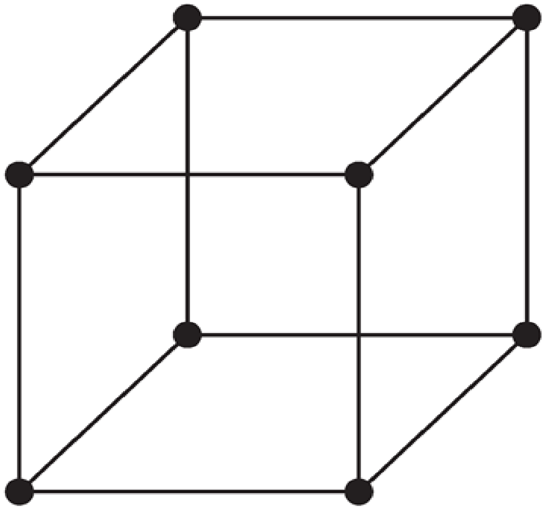


- **Definition** A graph (平面图) is called *planar* if it can be drawn in the plane without any edges crossing. Such a drawing is called a *planar representation* of the graph.

Example Is K_4 planar?

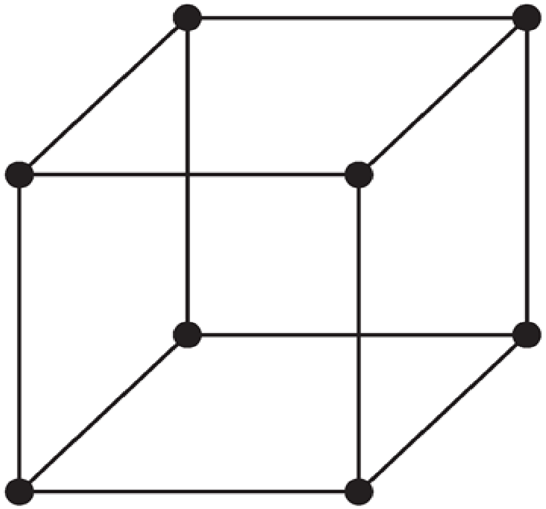


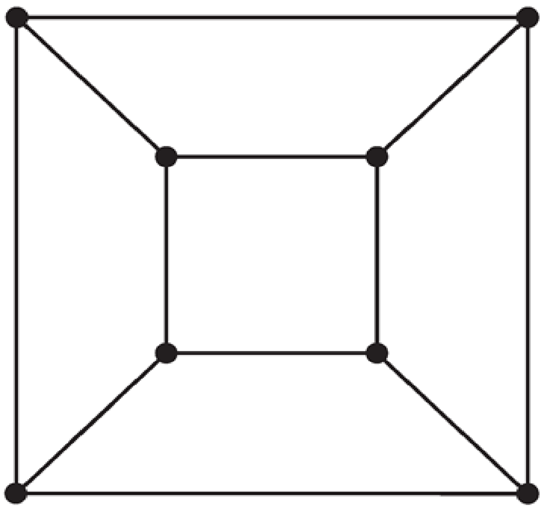
- **Example**



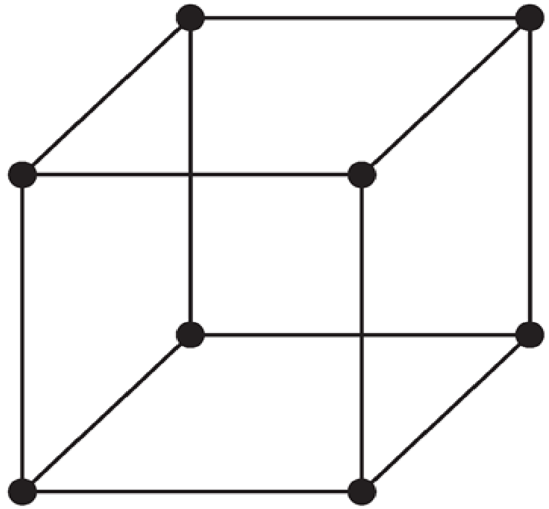
Planar Graphs

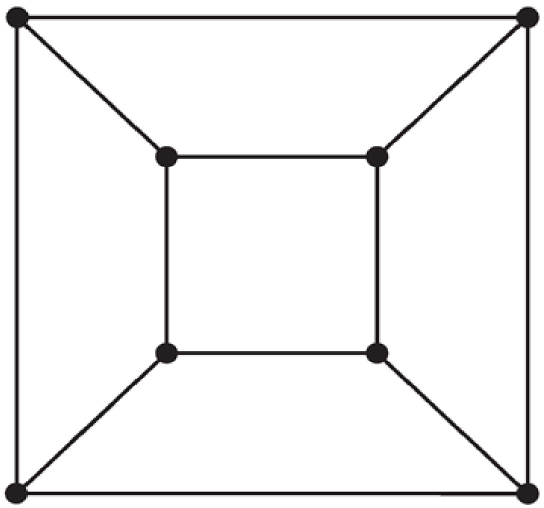
- **Example**

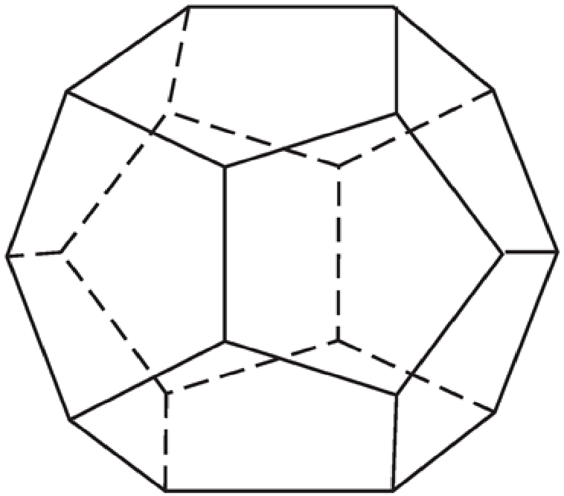




- **Example**

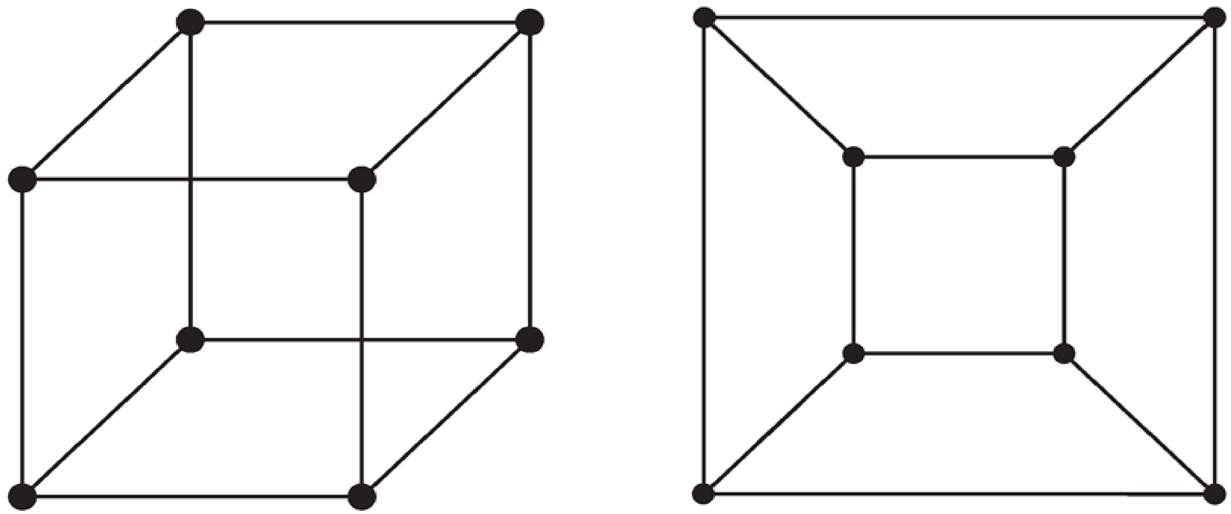


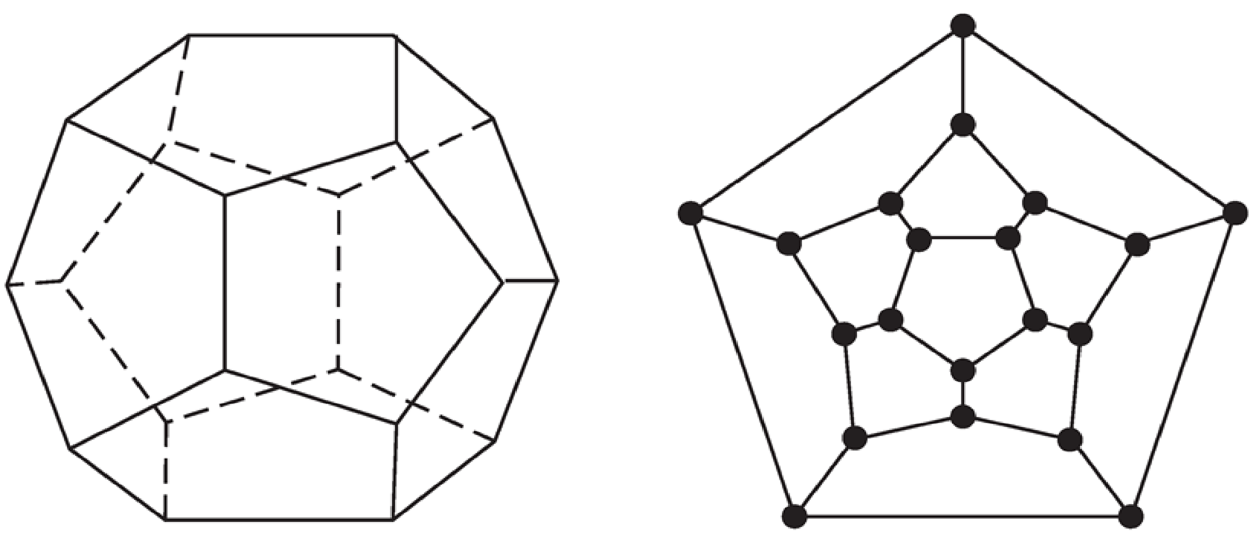




(a)

- **Example**

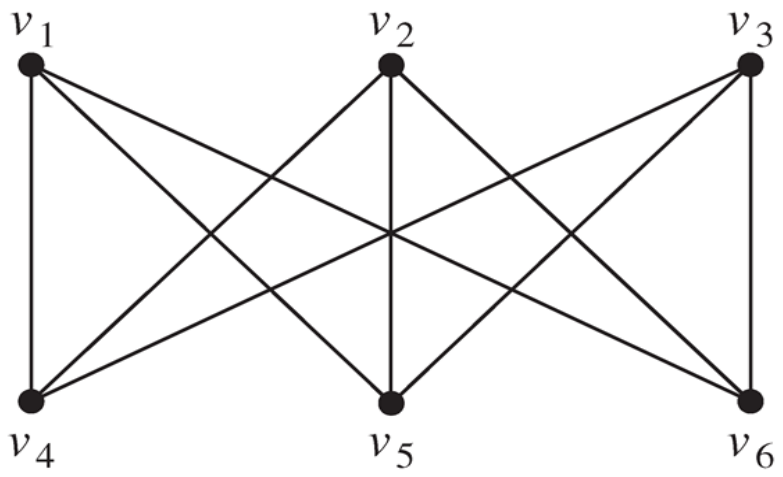




(a) (b)

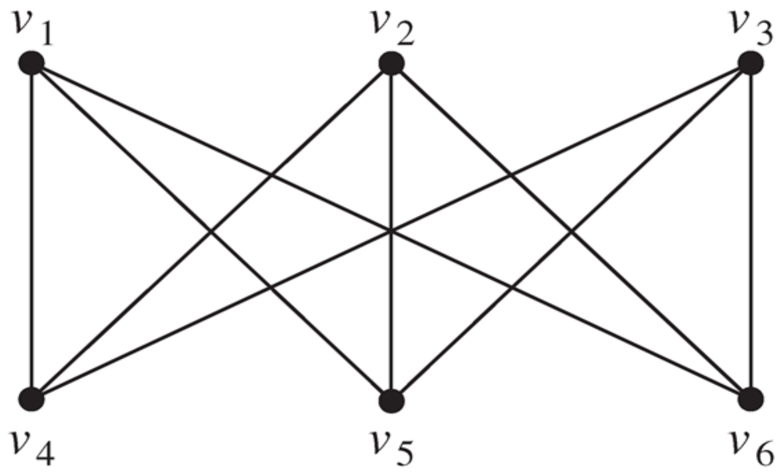
Planar Graphs

- **Example**



Planar Graphs

- **Example**



Applications

- ◇ IC design
- ◇ design of road networks

Euler's Formula

- **Theorem** (Euler's Formula) Let G be a connected planar simple graph with e edges and v vertices. Let r be the number of regions in a planar representation of G. Then $r = e - v + 2$.

  Proof (by induction)

- **Theorem** (Euler's Formula) Let G be a connected planar simple graph with e edges and v vertices. Let r be the number of regions in a planar representation of G. Then $r = e - v + 2$.

  Proof (by induction)

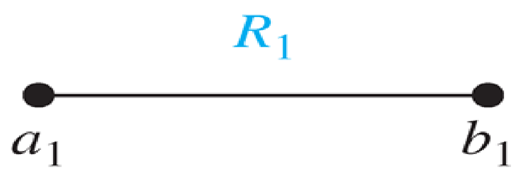
  Basic step:

Euler's Formula

- **Theorem** (Euler's Formula) Let G be a connected planar simple graph with e edges and v vertices. Let r be the number of regions in a planar representation of G. Then $r = e - v + 2$.

  Proof (by induction)

  Basic step:



Euler's Formula

- **Theorem** (Euler's Formula) Let G be a connected planar simple graph with e edges and v vertices. Let r be the number of regions in a planar representation of G. Then $r = e - v + 2$.

  连通平面图

  Proof (by induction)

  Basic step:

  Inductive Hypothesis:

- **Theorem** (Euler's Formula) Let G be a connected planar simple graph with e edges and v vertices. Let r be the number of regions in a planar representation of G. Then $r = e - v + 2$.

  Proof (by induction)

  Basic step:

  Inductive Hypothesis:

$$r_k = e_k - v_k + 2$$

- **Theorem** (Euler's Formula) Let G be a connected planar simple graph with e edges and v vertices. Let r be the number of regions in a planar representation of G. Then $r = e - v + 2$.

  Proof (by induction)

  Basic step:

  Inductive Hypothesis:

  Inductive step:

Euler's Formula

- **Theorem** (Euler's Formula) Let G be a connected planar simple graph with e edges and v vertices. Let r be the number of regions in a planar representation of G. Then $r = e - v + 2$.

  Proof (by induction)

  Basic step:

  Inductive Hypothesis:

  Inductive step:
  Let $\{a_{k+1}, b_{k+1}\}$ be the edge that is added to G_k to obtain G_{k+1}.

Euler's Formula

- **Theorem** (Euler's Formula) Let G be a connected planar simple graph with e edges and v vertices. Let r be the number of regions in a planar representation of G. Then $r = e - v + 2$.

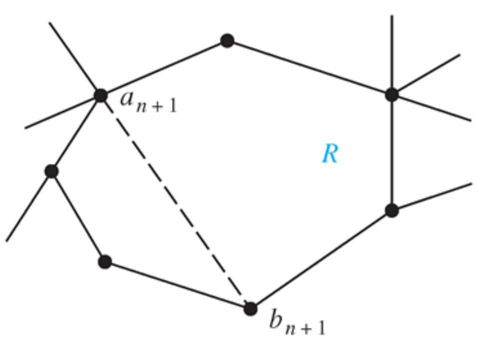
  Proof (by induction)

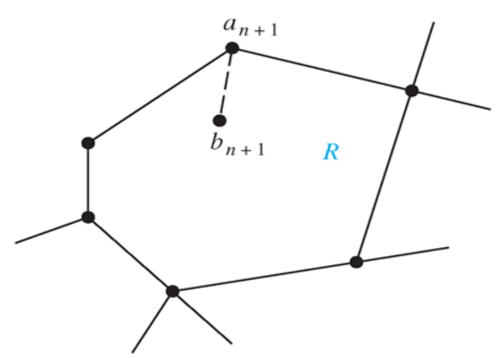
  Basic step:

  Inductive Hypothesis:

  Inductive step:

  Let $\{a_{k+1}, b_{k+1}\}$ be the edge that is added to G_k to obtain G_{k+1}.

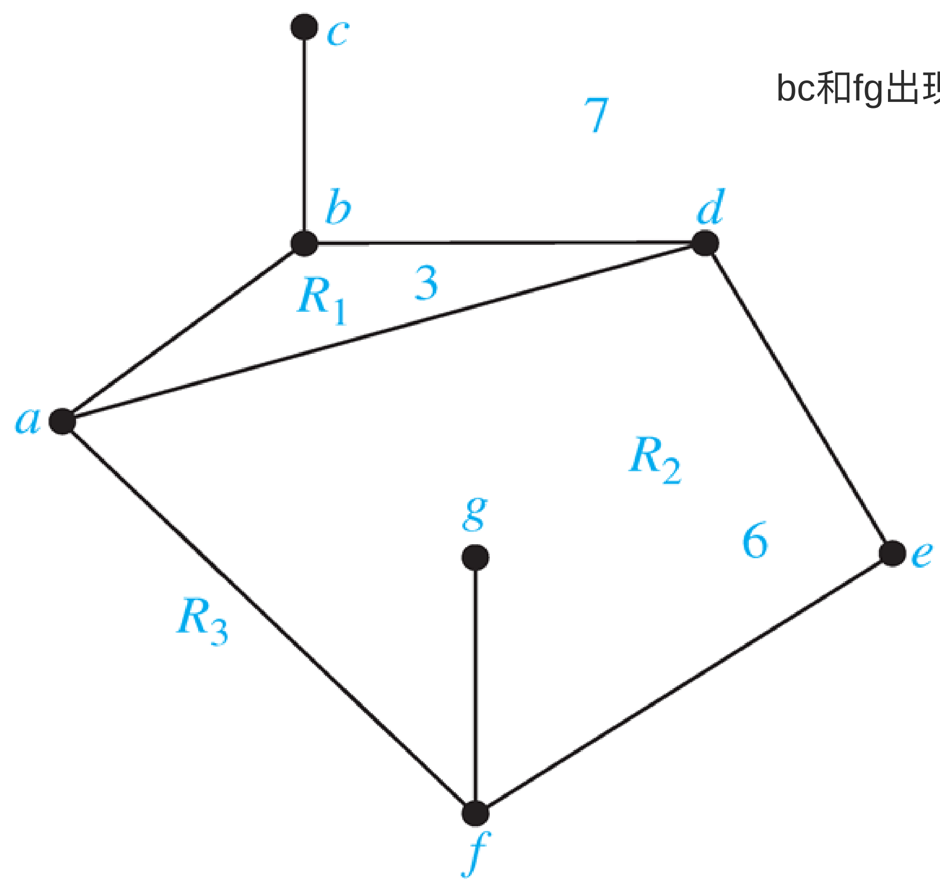




The Degree of Regions

- **Definition** The *degree* of a region is defined to be the number of edges on the boundary of this region. When an edge occurs twice on the boundary, it contributes two to the degree.

- **Definition** The *degree* of a region is defined to be the number of edges on the boundary of this region. When an edge occurs twice on the boundary, it contributes two to the degree.



bc和fg出现了两次

Corollaries

- **Corollary 1** If G is a connected planar simple graph with e edges and v vertices, where $v \geq 3$, then $e \leq 3v - 6$.

Corollaries

- **Corollary 1** If G is a connected planar simple graph with e edges and v vertices, where $v \geq 3$, then $e \leq 3v - 6$.

  Proof The degree of every region is at least 3.

Corollaries

- **Corollary 1** If G is a connected planar simple graph with e edges and v vertices, where $v \geq 3$, then $e \leq 3v - 6$.

  Proof The degree of every region is at least 3.

  - G is simple
  - $v \geq 3$

Corollaries

- **Corollary 1** If G is a connected planar simple graph with e edges and v vertices, where $v \geq 3$, then $e \leq 3v - 6$.

  Proof The degree of every region is at least 3.

  - ◇ G is simple
  - ◇ $v \geq 3$

  The sum of the degrees of the regions is exactly twice the number of edges in the graph.

Corollaries

- **Corollary 1** If G is a connected planar simple graph with e edges and v vertices, where $v \geq 3$, then $e \leq 3v - 6$.

Proof The degree of every region is at least 3.

- ◇ G is simple
- ◇ $v \geq 3$

每条边会给region一共贡献两个degree

The sum of the degrees of the regions is exactly twice the number of edges in the graph.

$$2e = \sum_{\text{all regions } R} \deg(R) \geq 3r$$

Corollaries

- **Corollary 1** If G is a connected planar simple graph with e edges and v vertices, where $v \geq 3$, then $e \leq 3v - 6$.

  Proof The degree of every region is at least 3.

  - G is simple
  - $v \geq 3$

  The sum of the degrees of the regions is exactly twice the number of edges in the graph.

  $$2e = \sum_{\text{all regions } R} \deg(R) \geq 3r$$

  By Euler's formula, the proof is completed.

Corollaries

- **Corollary 2** If G is a connected planar simple graph, then G has a vertex of degree not exceeding 5.

Corollaries

- **Corollary 2** If G is a connected planar simple graph, then G has a vertex of degree not exceeding 5.

  Proof

- **Corollary 2** If G is a connected planar simple graph, then G has a vertex of degree not exceeding 5.

Proof

(By contradiction)

By Corollary 1 and the Handshaking Theorem.

Corollaries

- **Corollary 2** If G is a connected planar simple graph, then G has a vertex of degree not exceeding 5.

  Proof

  图中所有顶点度数之和等于边数的两倍
  （region 的度数也类似）

  (By contradiction)

  By Corollary 1 and the Handshaking Theorem.

  Corollary 3 In a connected planar simple graph has e edges and v vertices with $v \geq 3$ and no circuits of length three, then $e \leq 2v - 4$.

Corollaries

- **Corollary 2** If G is a connected planar simple graph, then G has a vertex of degree not exceeding 5.

  Proof

  (By contradiction)

  By Corollary 1 and the Handshaking Theorem.

  Corollary 3 In a connected planar simple graph has e edges and v vertices with $v \geq 3$ and no circuits of length three, then $e \leq 2v - 4$.

  Proof similar to that of Corollary 1.

Examples

- Show that K_5 is nonplanar.

Examples

- Show that K_5 is nonplanar.

  Using Corollary 1

Examples

- Show that K_5 is nonplanar.

  Using Corollary 1

  Show that $K_{3,3}$ is nonplanar.

Examples

- Show that K_5 is nonplanar.

  Using Corollary 1

  Show that $K_{3,3}$ is nonplanar.

  Using Corollary 3

Examples

- Show that K_5 is nonplanar.

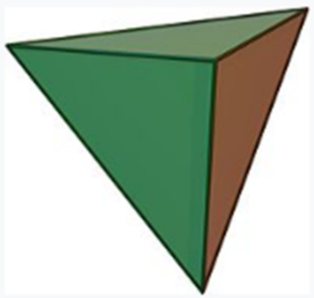
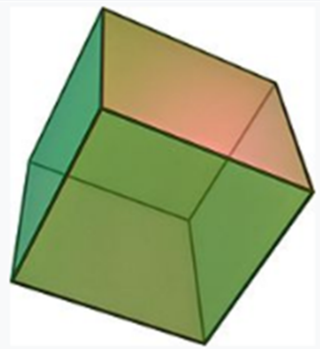
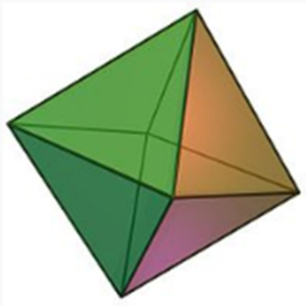
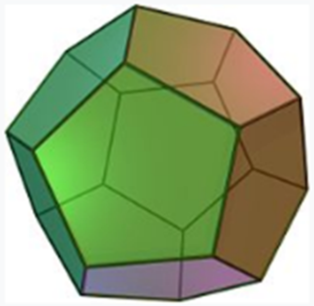
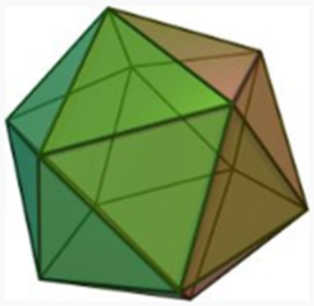
  Using Corollary 1

  Show that $K_{3,3}$ is nonplanar.

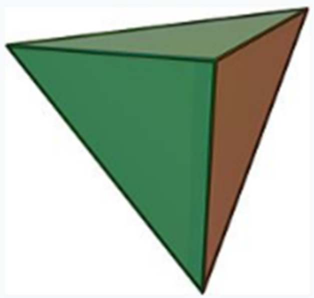
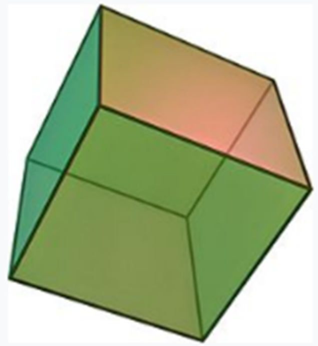
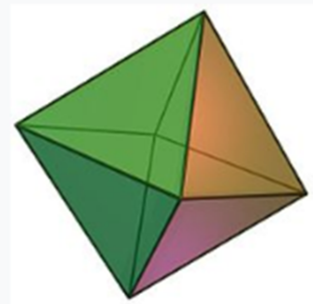
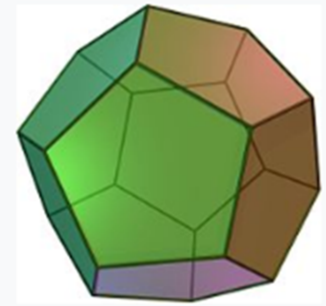
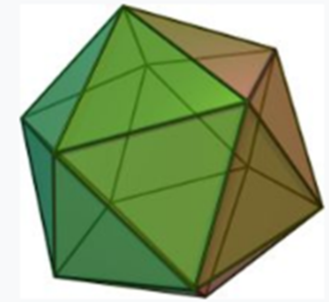
  Using Corollary 3

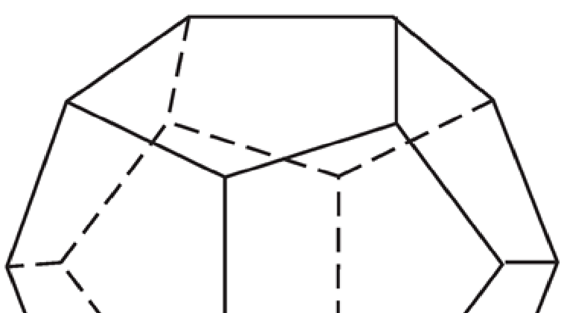
  Corollary 2 is used in the proof of Five Color Theorem.

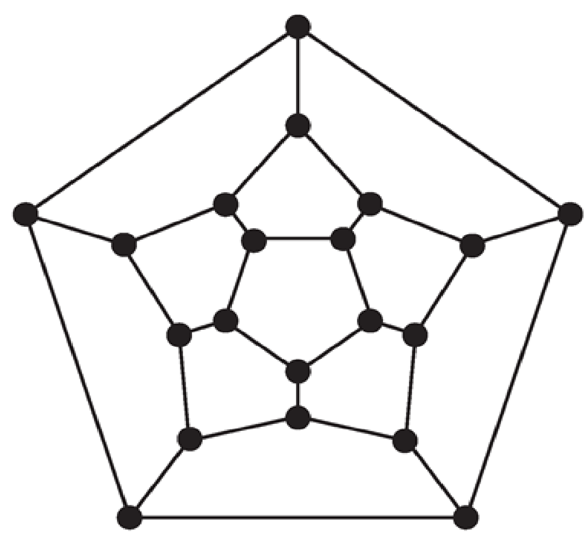
Only 5 Platonic Solids

				
Tetrahedron {3, 3}	Cube {4, 3}	Octahedron {3, 4}	Dodecahedron {5, 3}	Icosahedron {3, 5}

Only 5 Platonic Solids

				
Tetrahedron {3, 3}	Cube {4, 3}	Octahedron {3, 4}	Dodecahedron {5, 3}	Icosahedron {3, 5}





(b)

柏拉图立体的定义特性

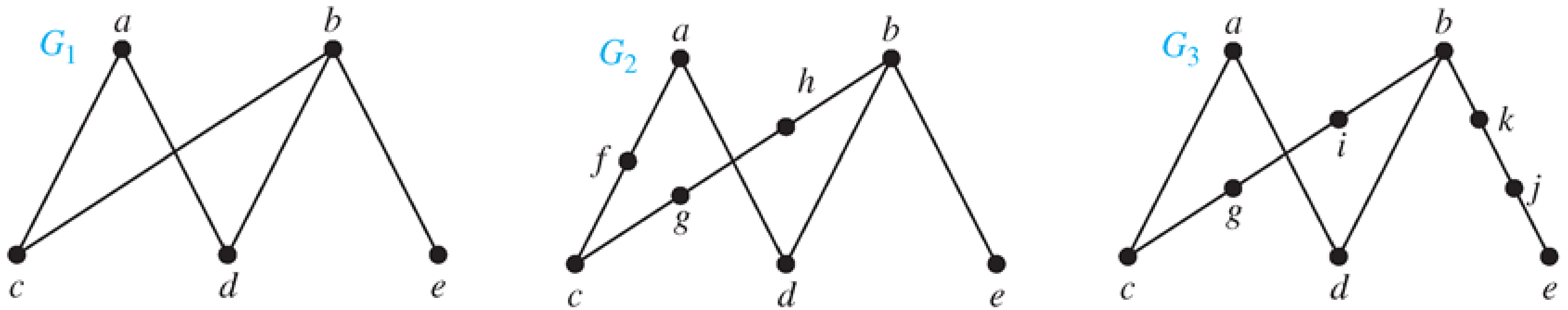
1. 所有的面都是全等的正多边形。
2. 每个顶点的邻接面数相同（即顶点是规则的）。
3. 内部角度规则。
4. 三维空间中的唯一性：只有五种这样的正多面体。

- **Definition** If a graph is planar, so will be any graph obtained by removing an edge $\{u, v\}$ and adding a new vertex w together with edges $\{u, w\}$ and $\{w, v\}$. Such an operation is called an *elementary subdivision*. The graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are called *homomorphic* if they can be obtained from the same graph by a sequence of elementary subdivisions.

1. **初等细分 (Elementary Subdivision)**：
   - 如果一个图是平面图（planar graph），那么通过以下操作所得的图仍然是平面图：
     - **移除一条边** $\{u, v\}$。
     - **在这条边上添加一个新顶点** w，并连接两条新边 $\{u, w\}$ 和 $\{w, v\}$。
   - 这种操作被称为**初等细分**（Elementary Subdivision）。
2. **同胚图 (Homomorphic Graphs)**：
   - 如果两个图 $G_1 = (V_1, E_1)$ 和 $G_2 = (V_2, E_2)$ 可以通过一系列初等细分从同一个图生成，那么它们被称为**同胚图**。

Kuratowski's Theorem

- **Definition** If a graph is planar, so will be any graph obtained by removing an edge $\{u, v\}$ and adding a new vertex w together with edges $\{u, w\}$ and $\{w, v\}$. Such an operation is called an *elementary subdivision*. The graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are called *homomorphic* if they can be obtained from the same graph by a sequence of elementary subdivisions.

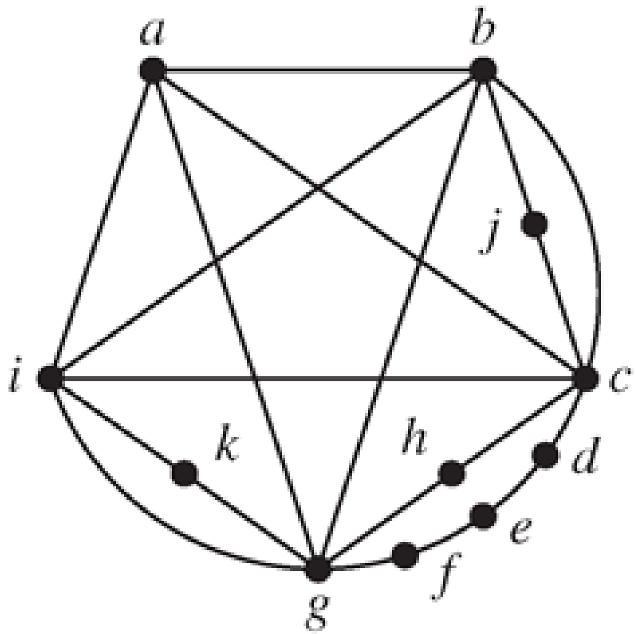


Kuratowski's Theorem

- **Definition** If a graph is planar, so will be any graph obtained by removing an edge $\{u, v\}$ and adding a new vertex w together with edges $\{u, w\}$ and $\{w, v\}$. Such an operation is called an *elementary subdivision*. The graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are called *homomorphic* if they can be obtained from the same graph by a sequence of elementary subdivisions.

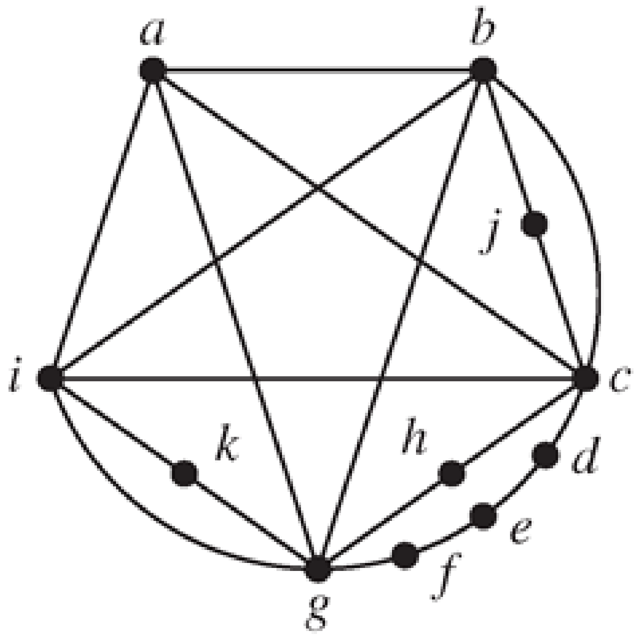
  Theorem A graph is nonplanar if and only if it contains a subgraph homomorphic to $K_{3,3}$ or K_5.

Examples

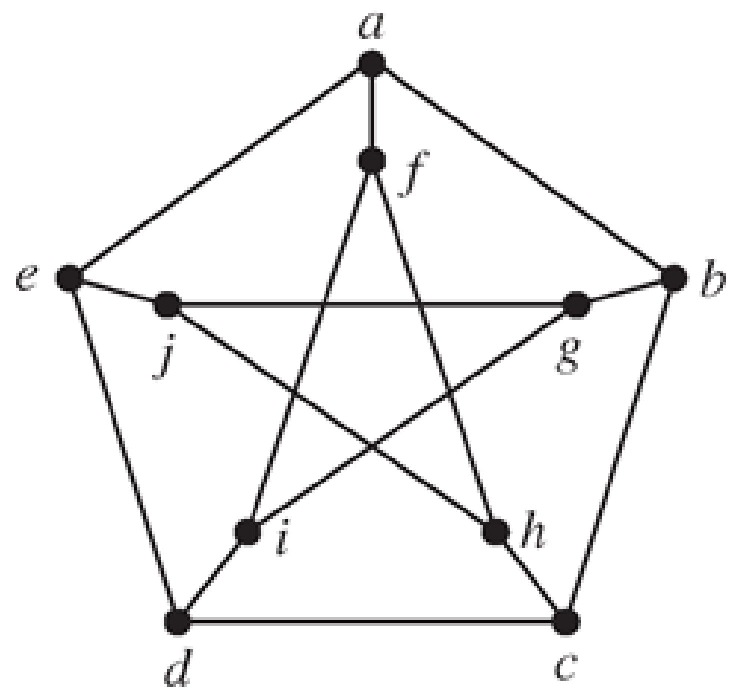


G

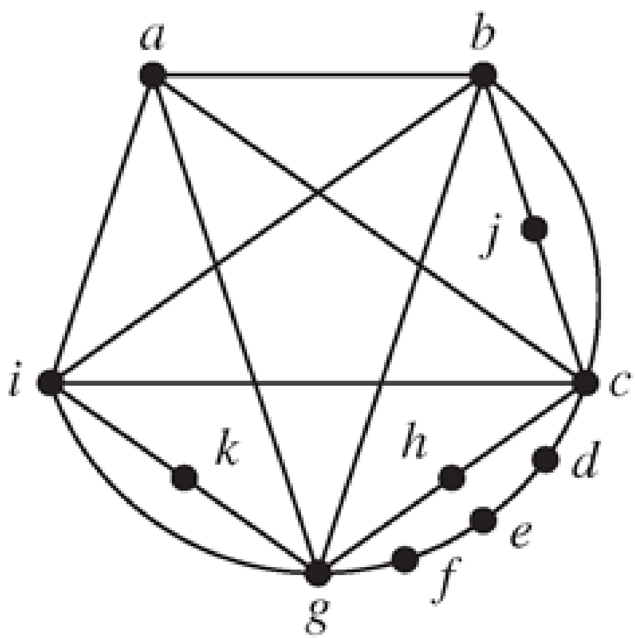
Examples



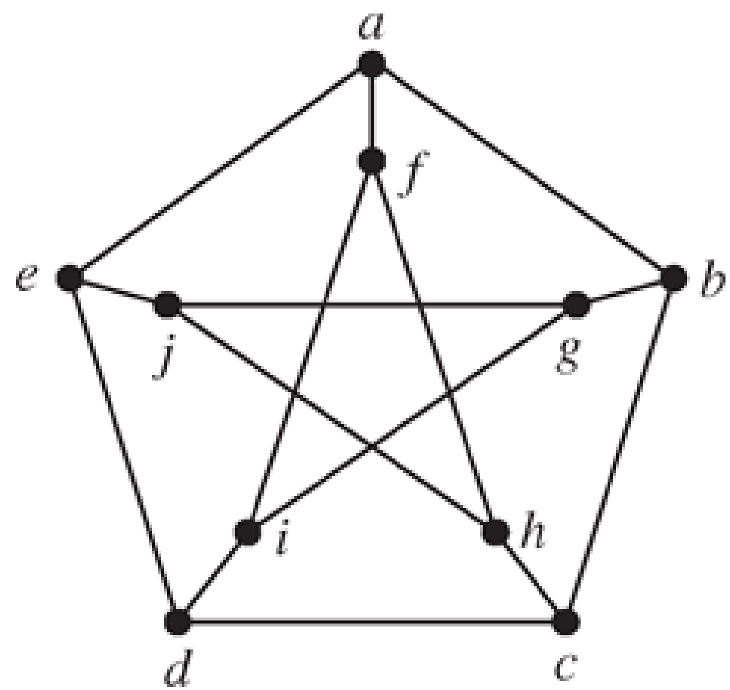
G

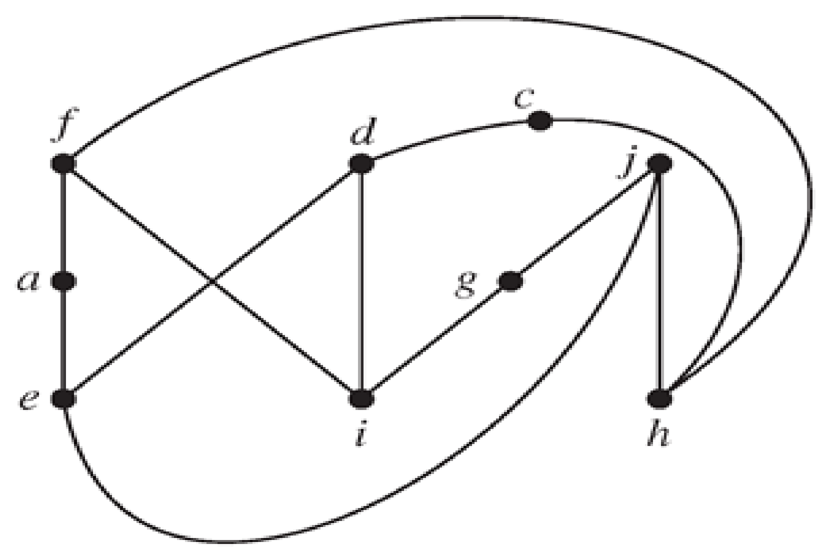


Examples



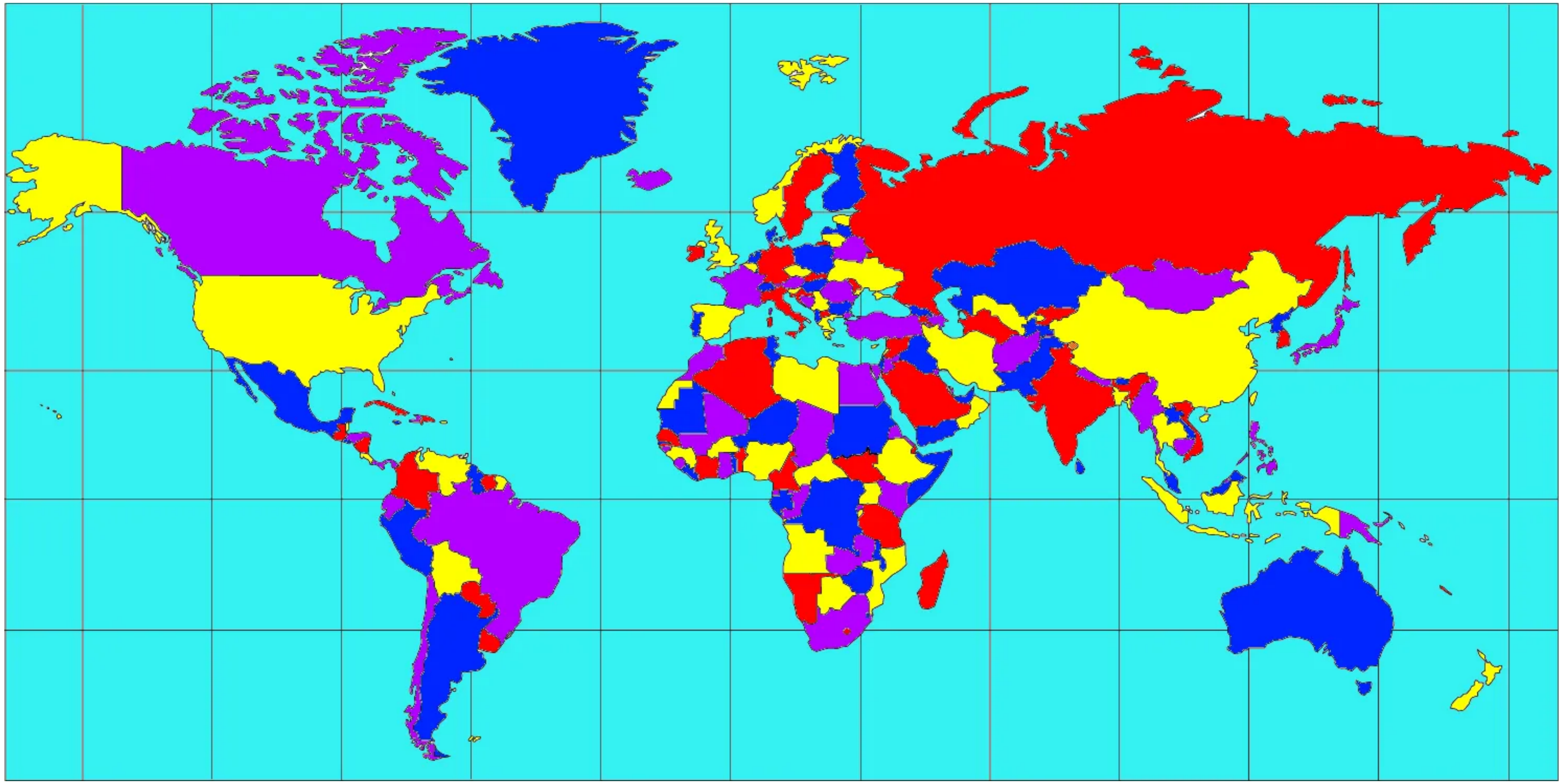
G





Graph Coloring

- **Four-color theorem** Given any separation of a plane into contiguous regions, producing a figure called a *map*, no more than four colors are required to color the regions of the map so that no two adjacent regions have the same color.



- **Four-color theorem**
  - ◇ first proposed by Francis Guthrie in 1852
  - ◇ his brother Frederick Guthrie told Augustus De Morgan
  - ◇ De Morgan wrote to William Hamilton
  - ◇ Alfred Kempe proved it incorrectly in 1879
  - ◇ Percy Heawood found an error in 1890 and proved the *five-color theorem*
  - ◇ Finally, Kenneth Appel and Wolfgang Haken proved it with case by case analysis by computer in 1976 (*the first computer-aided proof*)
  - ◇ Kempe's incorrect proof serves as a basis

Graph Coloring

- A *coloring* of a simple graph is the assignment of a color to each vertex of the graph so that no two adjacent vertices are assigned the same color.

Graph Coloring

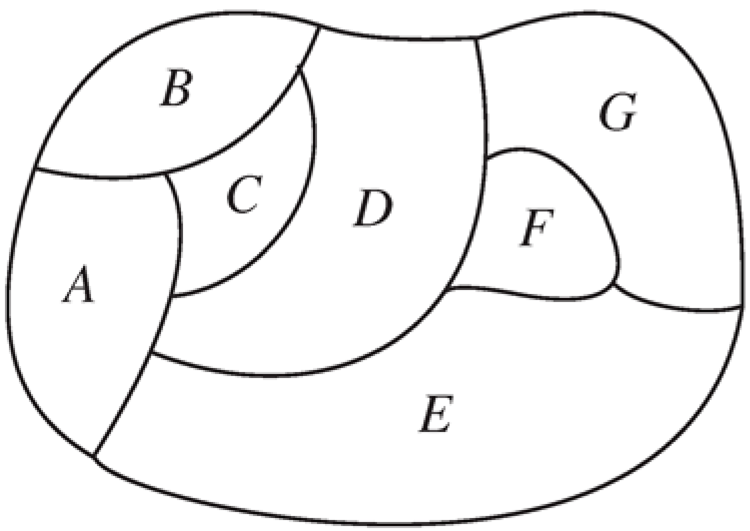
- A *coloring* of a simple graph is the assignment of a color to each vertex of the graph so that no two adjacent vertices are assigned the same color.

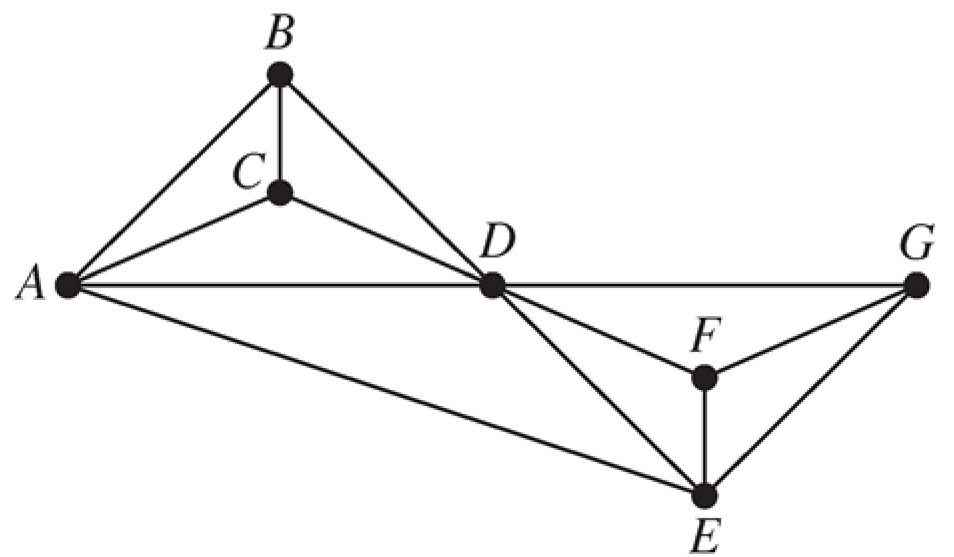
  The *chromatic number* of a graph is the least number of colors needed for a coloring of this graph, denoted by $\chi(G)$.

Graph Coloring

- A *coloring* of a simple graph is the assignment of a color to each vertex of the graph so that no two adjacent vertices are assigned the same color.

  The *chromatic number* of a graph is the least number of colors needed for a coloring of this graph, denoted by $\chi(G)$.



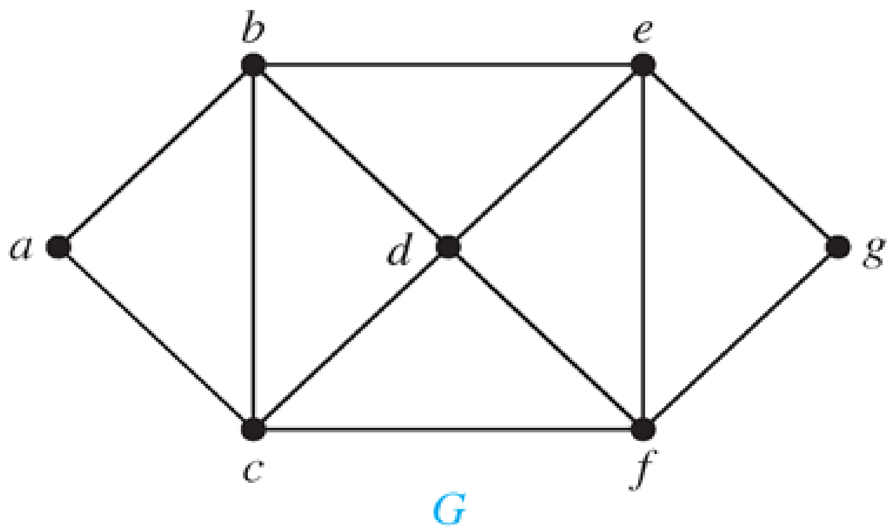


Graph Coloring

- **Theorem** (Four Color Theorem) The chromatic number of a planar graph is no greater than four.

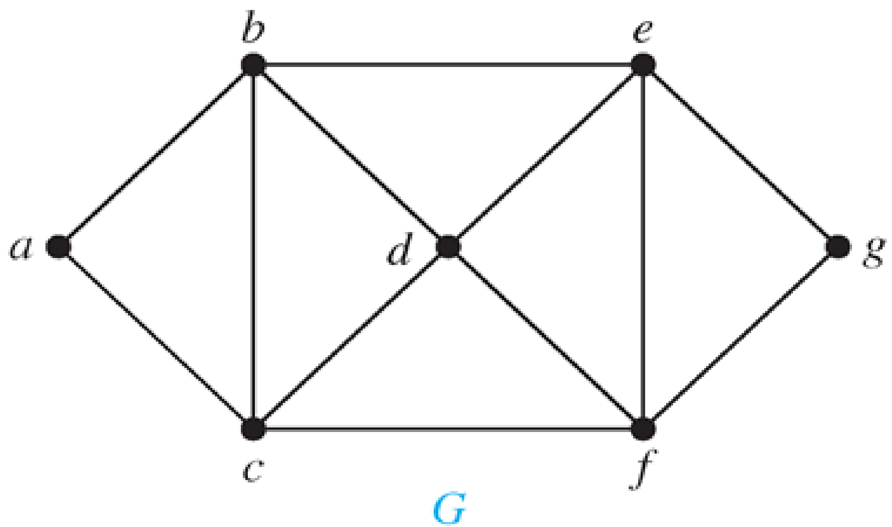
Graph Coloring

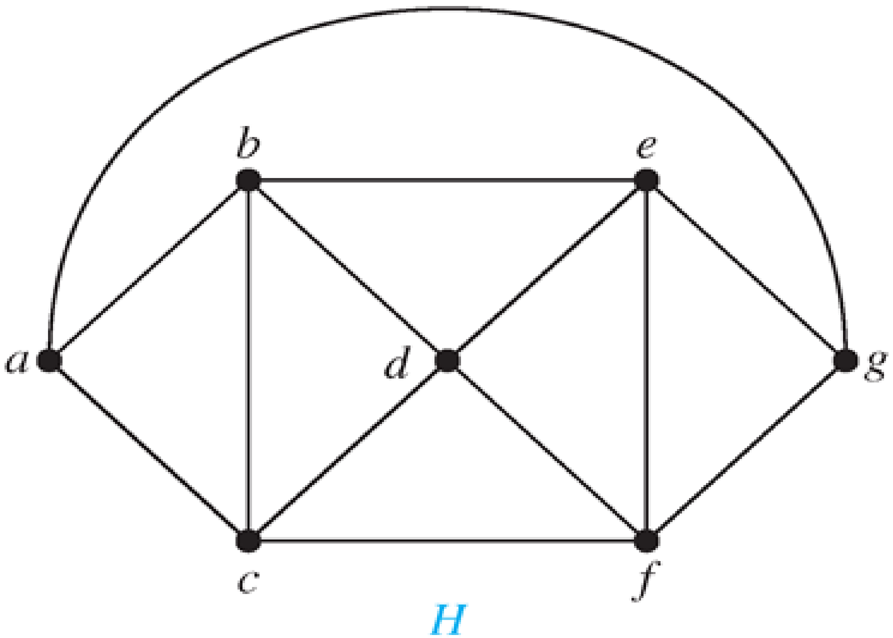
- **Theorem** (Four Color Theorem) The chromatic number of a planar graph is no greater than four.



Graph Coloring

- **Theorem** (Four Color Theorem) The chromatic number of a planar graph is no greater than four.





Graph Coloring

- **Theorem** (Six Color Theorem) The chromatic number of a planar graph is no greater than six.

Graph Coloring

- **Theorem** (Six Color Theorem) The chromatic number of a planar graph is no greater than six.

  Proof (by induction on the number of vertices)
  w.l.o.g., assume that the graph is connected.

Graph Coloring

- **Theorem** (Six Color Theorem) The chromatic number of a planar graph is no greater than six.

  Proof (by induction on the number of vertices)
  w.l.o.g., assume that the graph is connected.

  Basic step: For one single vertex, pick an arbitrary color.

Graph Coloring

- **Theorem** (Six Color Theorem) The chromatic number of a planar graph is no greater than six.

  Proof (by induction on the number of vertices)
  w.l.o.g., assume that the graph is connected.

  Basic step: For one single vertex, pick an arbitrary color.
  Inductive hypothesis: Assume that every planar graph with $k \geq 1$ or fewer vertices can be 6-colored.

- **Theorem** (Six Color Theorem) The chromatic number of a planar graph is no greater than six.

  Proof (by induction on the number of vertices)
  w.l.o.g., assume that the graph is connected.

  Basic step: For one single vertex, pick an arbitrary color.
  Inductive hypothesis: Assume that every planar graph with $k \geq 1$ or fewer vertices can be 6-colored.
  Inductive step: Consider a planar graph with $k+1$ vertices.

Graph Coloring

- **Theorem** (Six Color Theorem) The chromatic number of a planar graph is no greater than six.

  Proof (by induction on the number of vertices)
  w.l.o.g., assume that the graph is connected.

  Basic step: For one single vertex, pick an arbitrary color.
  Inductive hypothesis: Assume that every planar graph with $k \geq 1$ or fewer vertices can be 6-colored.
  Inductive step: Consider a planar graph with $k + 1$ vertices. Recall Corollary 2 (the graph has a vertex of degree 5 or fewer). Remove this vertex, by i.h., we can color the remaining graph with 6 colors. Put the vertex back in. Since there are at most 5 colors adjacent, so we have at least one color left.

Graph Coloring

- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

Graph Coloring

- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

  Proof (by induction on the number of vertices)
  w.l.o.g., assume that the graph is connected.

- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

  Proof (by induction on the number of vertices)
  w.l.o.g., assume that the graph is connected.

  Basic step: For one single vertex, pick an arbitrary color.

- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

  Proof (by induction on the number of vertices)
  w.l.o.g., assume that the graph is connected.

  Basic step: For one single vertex, pick an arbitrary color.
  Inductive hypothesis: Assume that every planar graph with $k \geq 1$ or fewer vertices can be 5-colored.

Graph Coloring

- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

  Proof (by induction on the number of vertices)
  w.l.o.g., assume that the graph is connected.

  Basic step: For one single vertex, pick an arbitrary color.
  Inductive hypothesis: Assume that every planar graph with $k \geq 1$ or fewer vertices can be 5-colored.
  Inductive step: Consider a planar graph with $k+1$ vertices.

- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

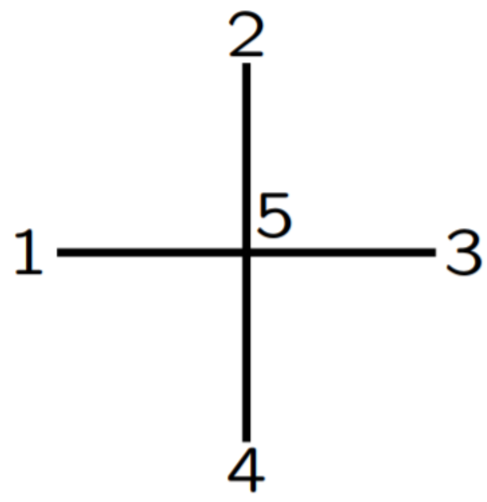
  Proof (by induction on the number of vertices)
  w.l.o.g., assume that the graph is connected.

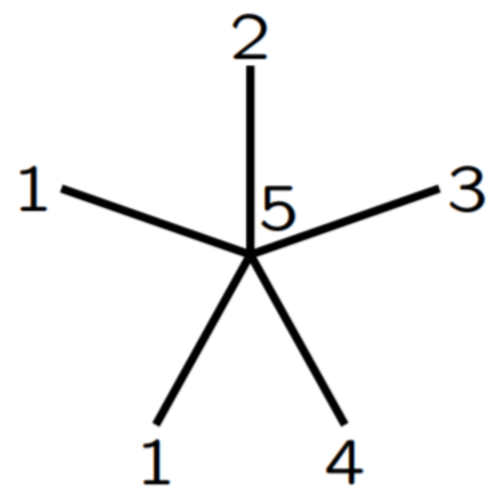
  Basic step: For one single vertex, pick an arbitrary color.
  Inductive hypothesis: Assume that every planar graph with $k \geq 1$ or fewer vertices can be 5-colored.
  Inductive step: Consider a planar graph with $k + 1$ vertices. Recall Corollary 2 (the graph has a vertex of degree 5 or fewer). Remove this vertex, by i.h., we can color the remaining graph with 5 colors. Put the vertex back in.

- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

  Proof (by induction on the number of vertices)
  w.l.o.g., assume that the graph is connected.

  If the vertex has degree less than 5, or if it has degree 5 and only ≤ 4 colors are used for vertices connected to it, we can pick an available color for it.

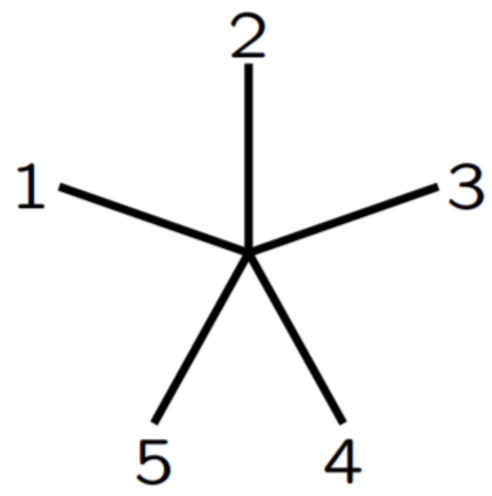




- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

  Proof (by induction on the number of vertices)

  If the vertex has degree 5, and all 5 colors are connected to it, we label the vertices adjacent to the "special" vertex (degree 5) 1 to 5 (in order).



Graph Coloring

- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

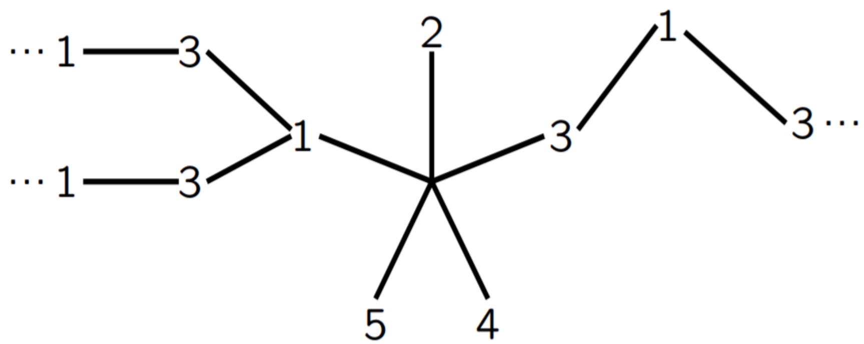
  Proof (by induction on the number of vertices)

  We make a subgraph out of all the vertices colored 1 or 3. If the adjacent vertex colored 1 and the adjacent vertex colored 3 are not connected by a path in the subgraph.

- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

  Proof (by induction on the number of vertices)

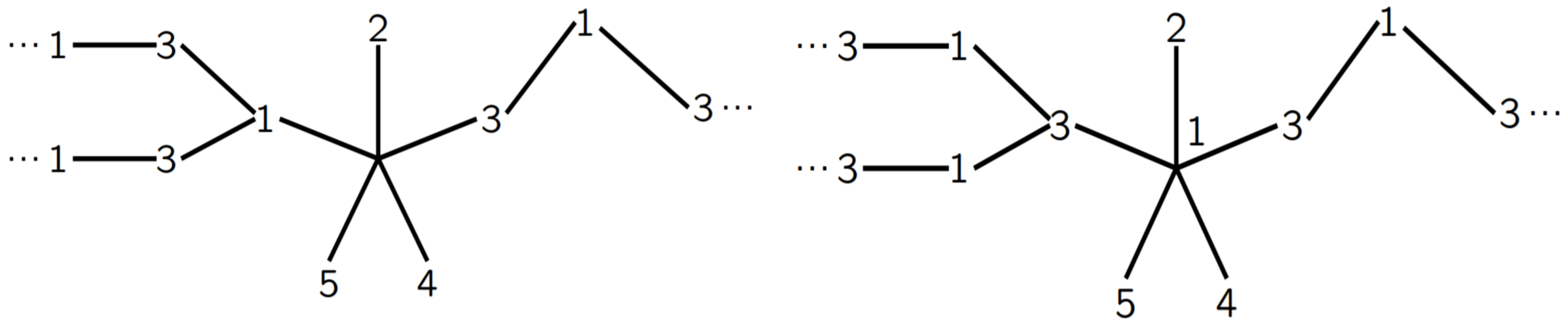
  We make a subgraph out of all the vertices colored 1 or 3. If the adjacent vertex colored 1 and the adjacent vertex colored 3 are not connected by a path in the subgraph.



- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

  Proof (by induction on the number of vertices)

  We make a subgraph out of all the vertices colored 1 or 3. If the adjacent vertex colored 1 and the adjacent vertex colored 3 are not connected by a path in the subgraph.



不连通可以调整颜色

- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

  Proof (by induction on the number of vertices)

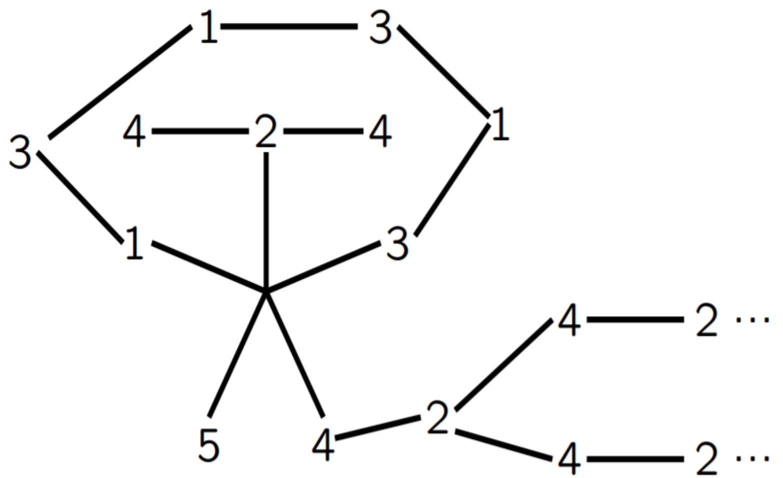
  On the other hand, if the vertices colored 1 and 3 are connected via a path in the subgraph, we do the the same for the vertices colored 2 and 4. Note that this will be a disconnected pair of subgraphs, separated by a path connecting the vertices colored 1 and 3 (Why?)

Graph Coloring

- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

  Proof (by induction on the number of vertices)

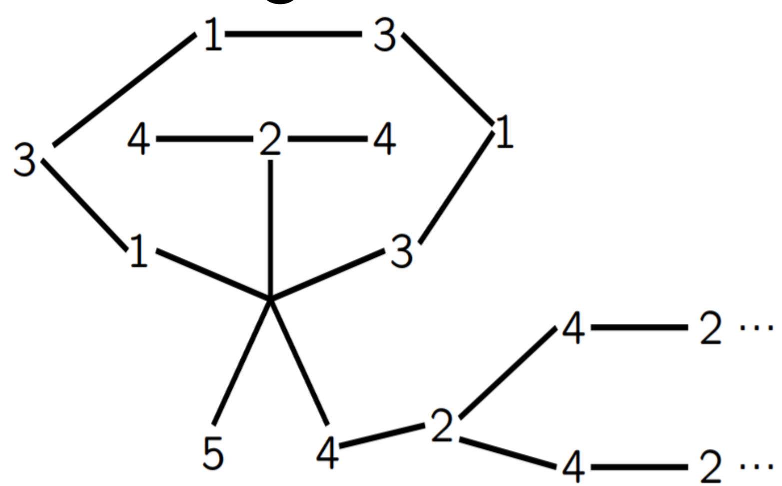
  On the other hand, if the vertices colored 1 and 3 are connected via a path in the subgraph, we do the the same for the vertices colored 2 and 4. Note that this will be a disconnected pair of subgraphs, separated by a path connecting the vertices colored 1 and 3 (Why?)

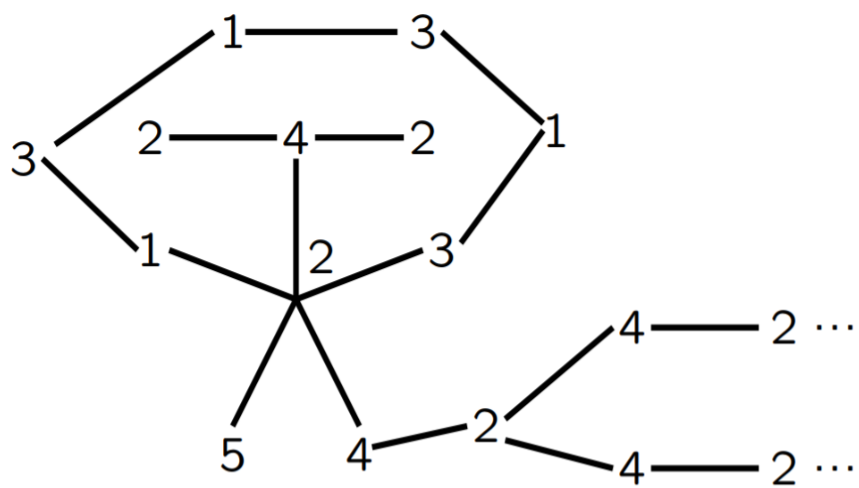


- **Theorem** (Five Color Theorem) The chromatic number of a planar graph is no greater than five.

  Proof (by induction on the number of vertices)

  On the other hand, if the vertices colored 1 and 3 are connected via a path in the subgraph, we do the the same for the vertices colored 2 and 4. Note that this will be a disconnected pair of subgraphs, separated by a path connecting the vertices colored 1 and 3 (Why?)



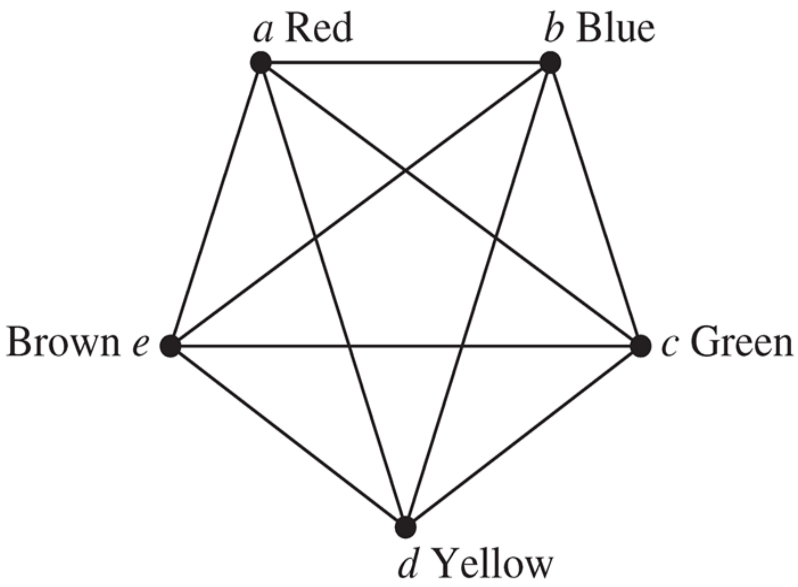


Examples

- What is the chromatic number of K_n, $K_{m,n}$, C_n?

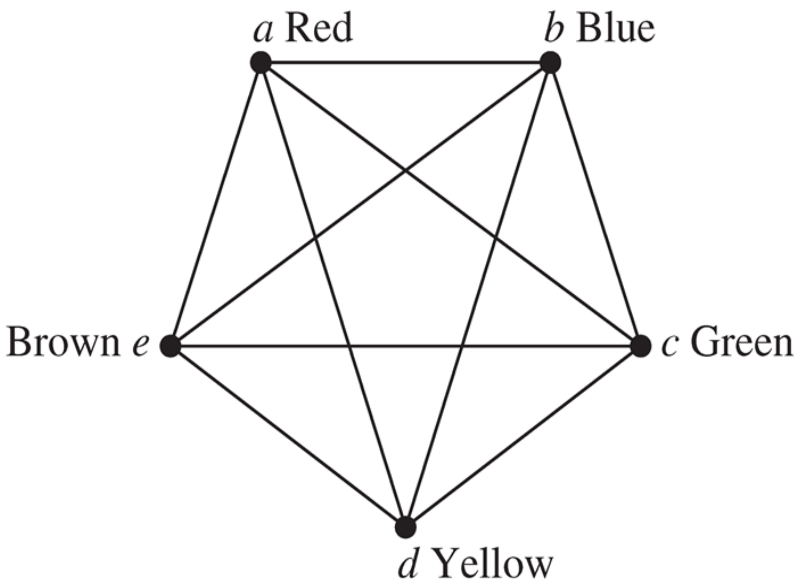
Examples

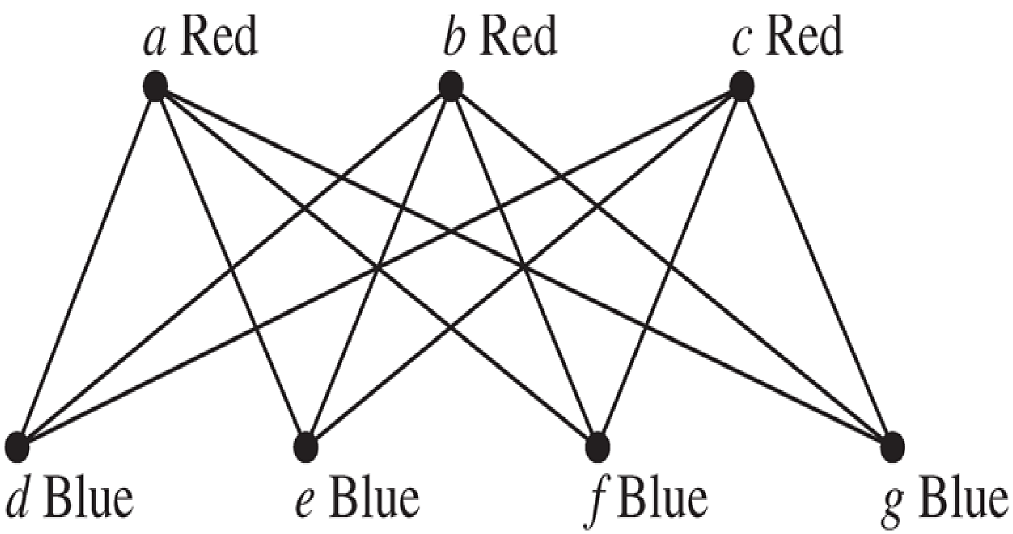
- What is the chromatic number of K_n, $K_{m,n}$, C_n?



Examples

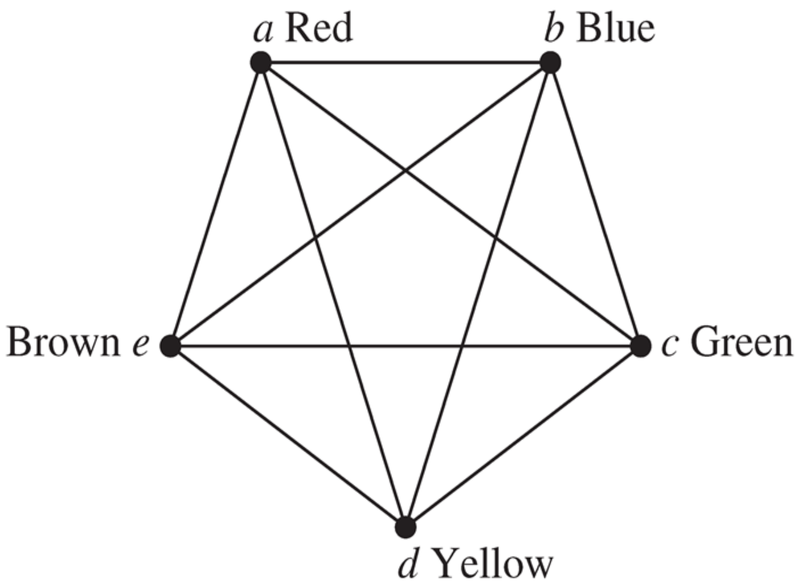
- What is the chromatic number of K_n, $K_{m,n}$, C_n?

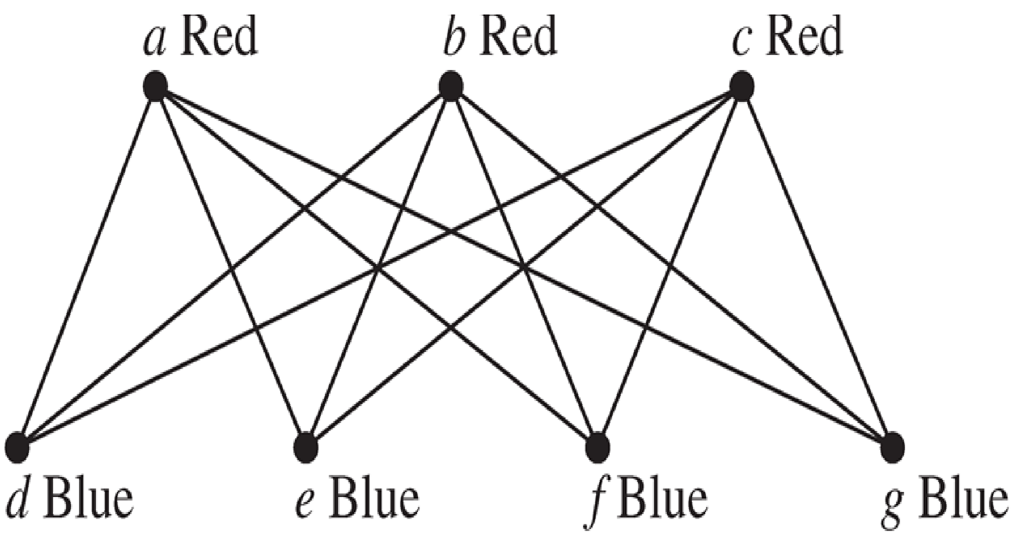


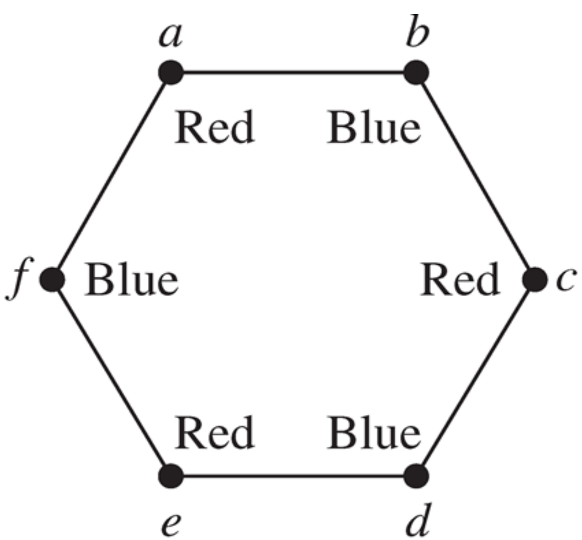


Examples

- What is the chromatic number of K_n, $K_{m,n}$, C_n?

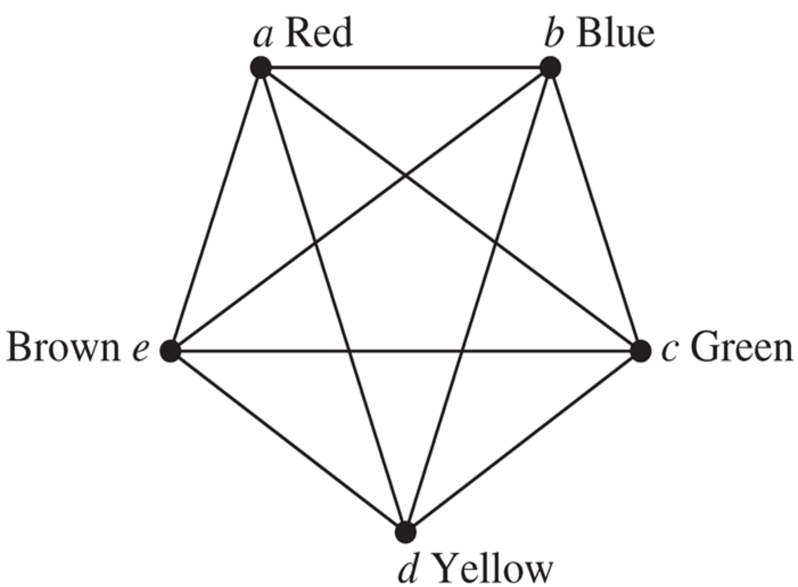


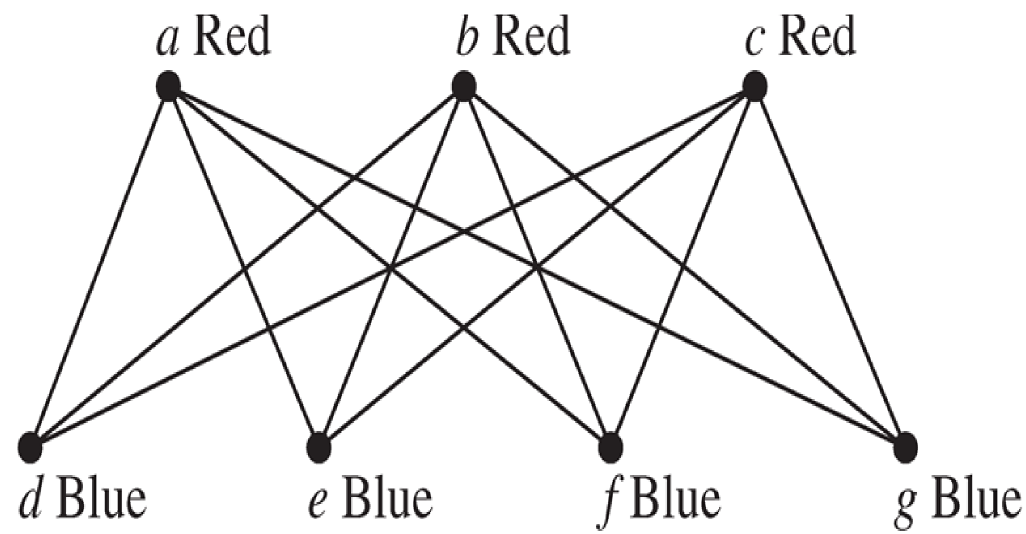


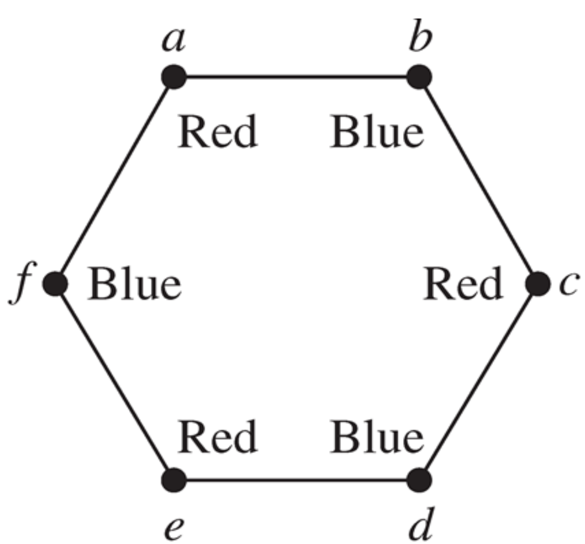


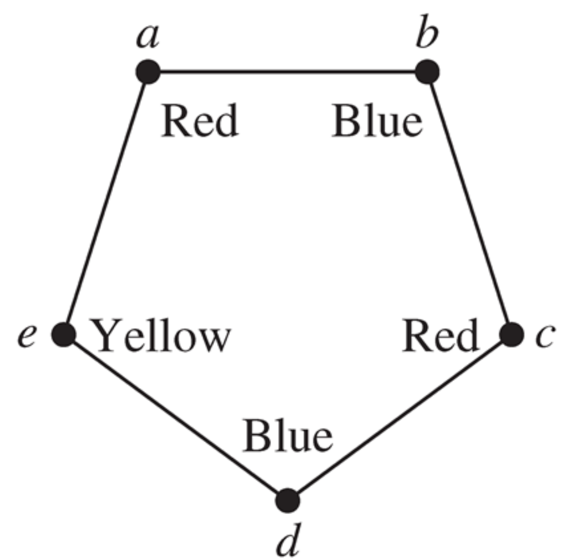
Examples

- What is the chromatic number of K_n, $K_{m,n}$, C_n?





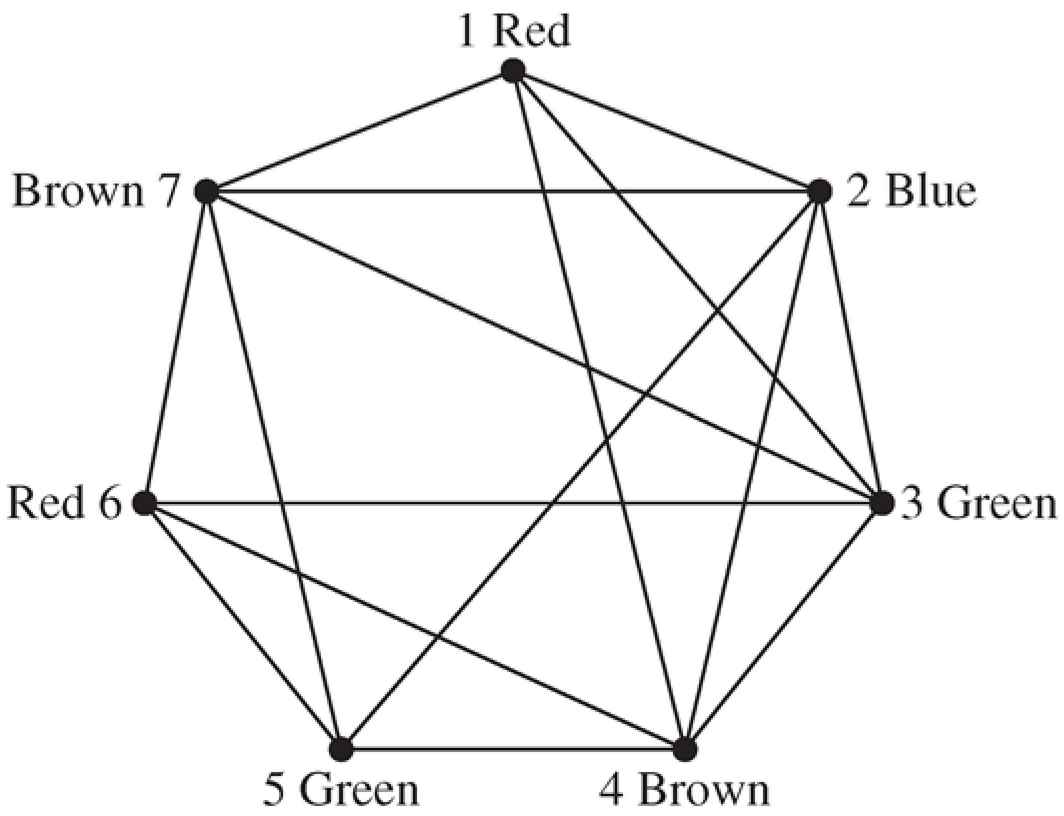




Applications of Graph Coloring

- Scheduling Final Exams

  Vertices represent courses, and there is an edge between two vertices if there is a common student in the courses.



Time Period	Courses
I	1, 6
II	2
III	3, 5
IV	4, 7

Applications of Graph Coloring

- Channel Assignments

  Television channels 2 through 13 are assigned to stations in North America so that no two stations within 150 miles can operate on the same channel . How can the assignment of channels be modeled by graph coloring?

Applications of Graph Coloring

- Channel Assignments

  Television channels 2 through 13 are assigned to stations in North America so that no two stations within 150 miles can operate on the same channel . How can the assignment of channels be modeled by graph coloring?

Graph Coloring $\in$ NPC

Next Lecture

- tree ...