Graphs

Stepan Kuznetsov

Computer Science Department, Higher School of Economics

Outline

The Notion of Graph

Trees

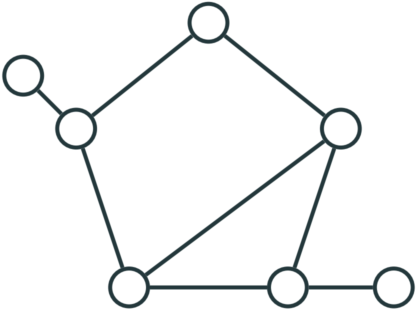
Colorings. Bipartite Graphs

The Notion of Graph

A *graph* is a set of *vertices*, some of which are connected by *edges*.

The Notion of Graph

A *graph* is a set of *vertices*, some of which are connected by *edges*.

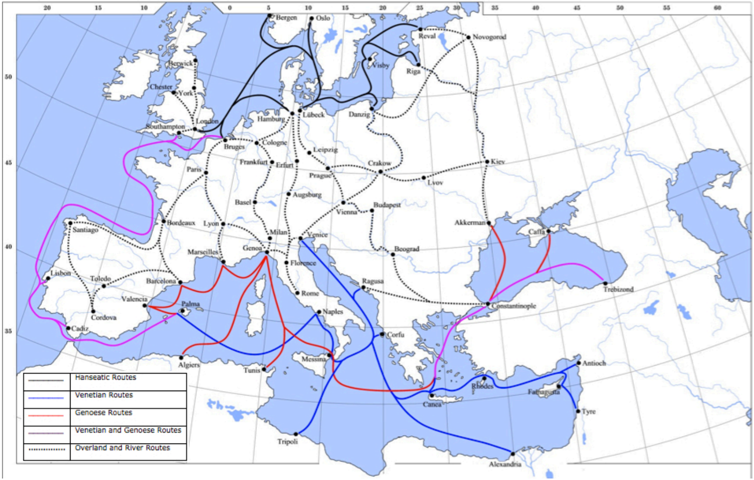


Applications of Graphs

— Graphs are everywhere!

Applications of Graphs

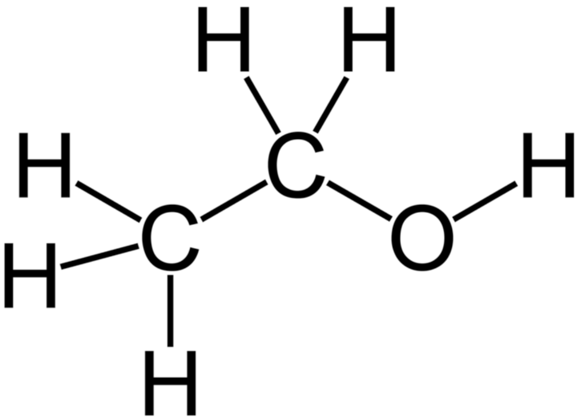
Maps (GIS): vertices = cities, edges = routes.



Wikipedia

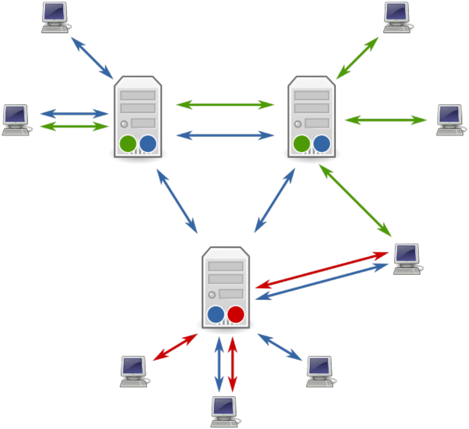
Applications of Graphs

Chemistry: graphs of molecular structure.



Applications of Graphs

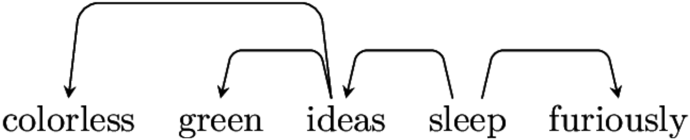
Internet: network topology.



Benjamin D. Esham @ Wikipedia

Applications of Graphs

Linguistics: syntactic dependencies.



Applications of Graphs

... and many, many more!

Directed and Undirected Graphs

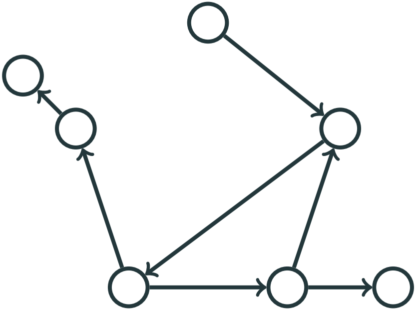
- Sometimes the graph is *directed.*

Directed and Undirected Graphs

- Sometimes the graph is *directed.*
- This means that each edge has its direction from one vertex to another.

Directed and Undirected Graphs

- Sometimes the graph is *directed.*
- This means that each edge has its direction from one vertex to another.
- Example:



Directed and Undirected Graphs

- In social networks, for example, the *friendship* relation can be represented as an *undirected* graph.

Directed and Undirected Graphs

- In social networks, for example, the *friendship* relation can be represented as an *undirected* graph.
- If Alex is a friend of Bob, then Bob is a friend of Alex.

Directed and Undirected Graphs

- In social networks, for example, the *friendship* relation can be represented as an *undirected* graph.
- If Alex is a friend of Bob, then Bob is a friend of Alex.
- In contrast, the *subscription* graph is *directed.*

Directed and Undirected Graphs

- In social networks, for example, the *friendship* relation can be represented as an *undirected* graph.
- If Alex is a friend of Bob, then Bob is a friend of Alex.
- In contrast, the *subscription* graph is *directed.*
- Say, Carl is subscribed to Wall Street Journal, while the editorial of Wall Street Journal doesn’t even know about Carl.

Directed and Undirected Graphs

- In social networks, for example, the *friendship* relation can be represented as an *undirected* graph.
- If Alex is a friend of Bob, then Bob is a friend of Alex.
- In contrast, the *subscription* graph is *directed.*
- Say, Carl is subscribed to Wall Street Journal, while the editorial of Wall Street Journal doesn't even know about Carl.
- In this course, most graphs will be undirected.

Loops and Parallel Edges

- Beware of two special kinds of edges in graphs.

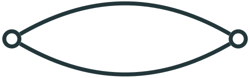
Loops and Parallel Edges

- Beware of two special kinds of edges in graphs.
- *Loop:* a vertex connected to itself.

Loops and Parallel Edges

- Beware of two special kinds of edges in graphs.
- *Loop:* a vertex connected to itself.

- *Parallel edges:* two vertices connected by more than one edge.



Loops and Parallel Edges

- By default, loops and parallel edges are **disallowed.**

Loops and Parallel Edges

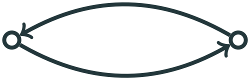
- By default, loops and parallel edges are **disallowed.**
- A graph with parallel edges is called a *multigraph*.

Loops and Parallel Edges

- By default, loops and parallel edges are **disallowed.**
- A graph with parallel edges is called a *multigraph*.
- A graph with parallel edges and loops is called a *pseudograph*.

Loops and Parallel Edges

- By default, loops and parallel edges are **disallowed.**
- A graph with parallel edges is called a *multigraph*.
- A graph with parallel edges and loops is called a *pseudograph*.
- Note that in a directed graph edges connecting two vertices in different directions are **not** considered parallel.



Outline

The Notion of Graph

Trees

Colorings. Bipartite Graphs

Paths and Cycles

- A *path* in a graph is a sequence of vertices $v_0, v_1, \ldots, v_n$, such that v_i is connected to v_{i+1}.

Paths and Cycles

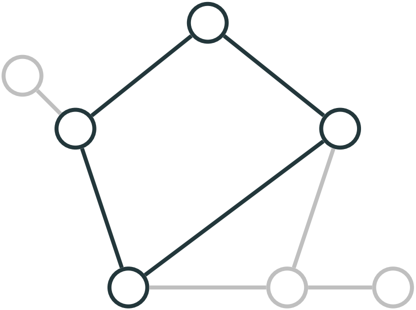
- A *path* in a graph is a sequence of vertices $v_0, v_1, \ldots, v_n$, such that v_i is connected to v_{i+1}.
- If $v_n = v_0$, then this path is a *cycle*.

Paths and Cycles

- A *path* in a graph is a sequence of vertices $v_0, v_1, \dots, v_n$, such that v_i is connected to v_{i+1}.
- If $v_n = v_0$, then this path is a *cycle*.
- A cycle is *simple,* if $v_1, \dots, v_n$ are different vertices and $n \geq 3$.

Paths and Cycles

- A *path* in a graph is a sequence of vertices $v_0, v_1, \ldots, v_n$, such that v_i is connected to v_{i+1}.
- If $v_n = v_0$, then this path is a *cycle*.
- A cycle is *simple,* if $v_1, \ldots, v_n$ are different vertices and $n \geq 3$.

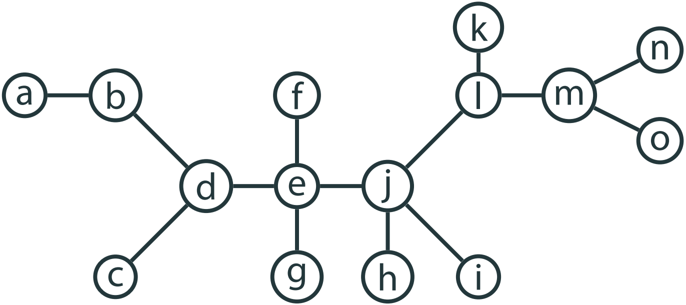


Trees

- A graph without simple cycles is a *tree*.

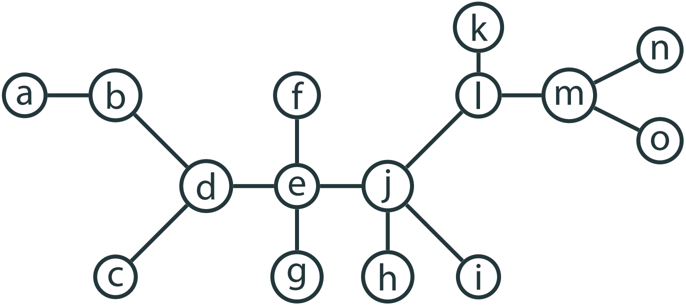
Trees

- A graph without simple cycles is a *tree*.



Trees

- A graph without simple cycles is a *tree*.



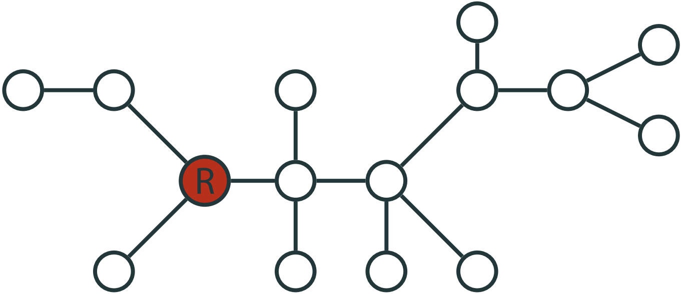
- Non-simple cycles could still exist, for example d–e–f–e–g–d on the graph above.

Rooted Trees

- One can pick an **arbitrary** vertex of a tree and declare it as the *root*:

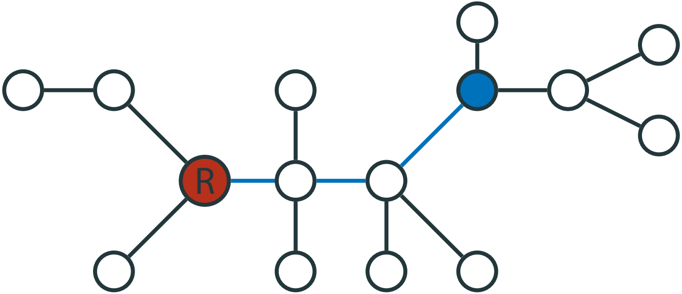
Rooted Trees

- One can pick an **arbitrary** vertex of a tree and declare it as the *root*:



Rooted Trees

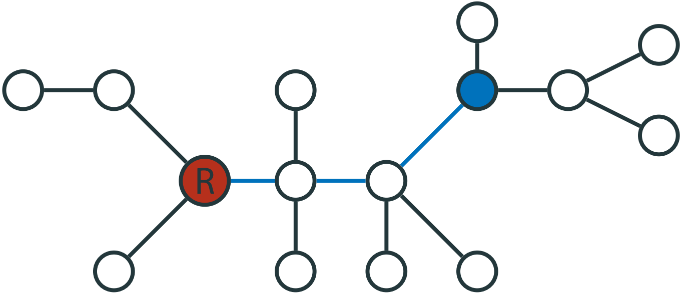
- One can pick an **arbitrary** vertex of a tree and declare it as the *root*:



- For any vertex (except the root) there is a unique path from the root to this vertex.

Rooted Trees

- One can pick an **arbitrary** vertex of a tree and declare it as the *root*:



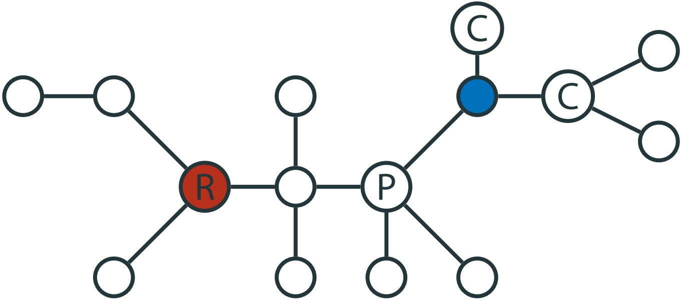
- For any vertex (except the root) there is a unique path from the root to this vertex.
- Otherwise, there would be a cycle.

Parents and Children

- Thus, for any vertex (except the root), there is exactly one vertex, which is next to it on the path towards the root.
- This is the *parent* of the given vertex.
- Other vertices are *children*.

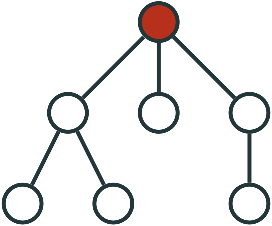
Parents and Children

- Thus, for any vertex (except the root), there is exactly one vertex, which is next to it on the path towards the root.
- This is the *parent* of the given vertex.
- Other vertices are *children*.



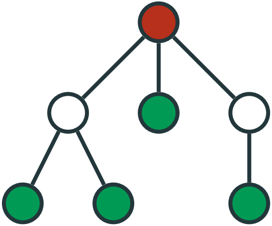
Drawing Rooted Trees

- A tree with a root is usually drawn in such a way that for each vertex its parent is located above and its children below the vertex.



Drawing Rooted Trees

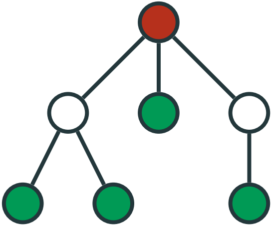
- A tree with a root is usually drawn in such a way that for each vertex its parent is located above and its children below the vertex.



- Vertices without children are called *leaves*.

Drawing Rooted Trees

- A tree with a root is usually drawn in such a way that for each vertex its parent is located above and its children below the vertex.



- Vertices without children are called *leaves*.
- *Notice:* in computer science, trees usually grow down (root at the top, leaves at the bottom)!

Application: Decision Trees

- Decision trees are used in machine learning as an interpretable output of an ML algorithm.

Application: Decision Trees

- Decision trees are used in machine learning as an interpretable output of an ML algorithm.
- The tree shows **how** the classification is performed.

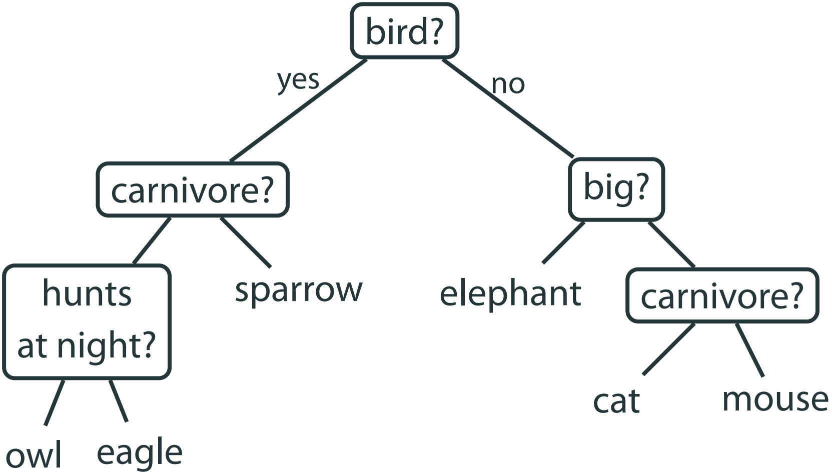
Application: Decision Trees

- Decision trees are used in machine learning as an interpretable output of an ML algorithm.
- The tree shows **how** the classification is performed.
- Decision trees are also used as a computational model in complexity theory.

Application: Decision Trees

- Decision trees are used in machine learning as an interpretable output of an ML algorithm.
- The tree shows **how** the classification is performed.
- Decision trees are also used as a computational model in complexity theory.
- In a decision tree, the inner nodes represent queries (usually binary) of input data, and leaves are answers.

Application: Decision Trees



The Number of Edges in a Tree

Theorem

If a tree has n vertices and m edges, then $m = n - 1$.

The Number of Edges in a Tree

Theorem

If a tree has n vertices and m edges, then $m = n - 1$.

Proof:

The Number of Edges in a Tree

Theorem

If a tree has n vertices and m edges, then $m = n - 1$.

Proof:

- For any vertex (except the root) there is a unique path from the root to this vertex.

The Number of Edges in a Tree

Theorem

If a tree has n vertices and m edges, then $m = n - 1$.

Proof:

- For any vertex (except the root) there is a unique path from the root to this vertex.
- The non-root vertices are in one-to-one correspondence with edges (each vertex corresponds to the last edge of such path).

The Number of Edges in a Tree

Theorem

If a tree has n vertices and m edges, then $m = n - 1$.

Proof:

- For any vertex (except the root) there is a unique path from the root to this vertex.
- The non-root vertices are in one-to-one correspondence with edges (each vertex corresponds to the last edge of such path).
- Thus, $m = n - 1$.

Outline

The Notion of Graph

Trees

Colorings. Bipartite Graphs

Vertex Colorings

- We consider only *vertex* colorings.

Vertex Colorings

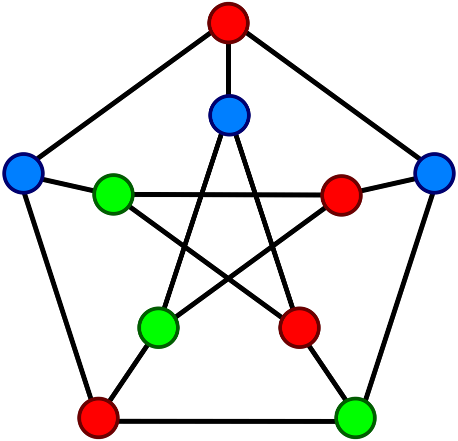
- We consider only *vertex* colorings.
- In a coloring, each vertex is marked by a *color* taken from a finite set of possible colors (for example, { red, green, blue }).

Vertex Colorings

- We consider only *vertex* colorings.
- In a coloring, each vertex is marked by a *color* taken from a finite set of possible colors (for example, { red, green, blue }).
- A coloring is *proper,* if endpoints of any edge have different colors.

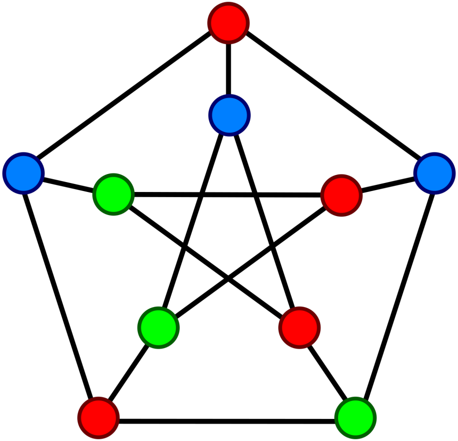
Vertex Colorings

- Example: Petersen graph properly 3-colored.



Vertex Colorings

- Example: Petersen graph properly 3-colored.



- Further we consider only proper colorings.

The Four Color Theorem

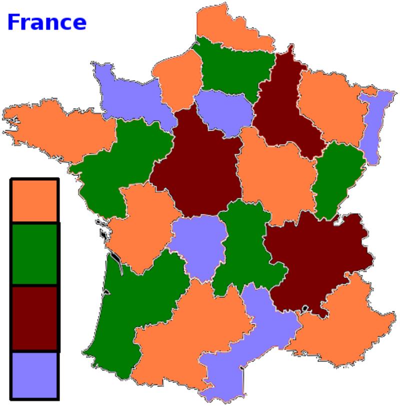
Theorem

Any map can be colored in 4 colors, such that adjacent areas have different colors.

The Four Color Theorem

Theorem

Any map can be colored in 4 colors, such that adjacent areas have different colors.



Proved by K. Appel and W. Haken in 1976.

The Four Color Theorem

The four color theorem can be formulated in graph terms.

Theorem

If a graph can be drawn on a plane without intersections, then it is 4-colorable.

The Four Color Theorem

The four color theorem can be formulated in graph terms.

Theorem

If a graph can be drawn on a plane without intersections, then it is 4-colorable.

Vertices = areas; edges connect adjacent areas.

Coloring in Two Colors

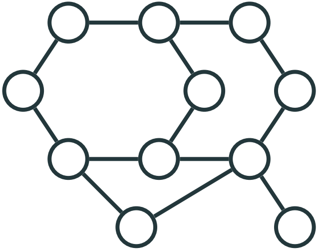
- A special case of vertex coloring is *2-coloring,* or coloring in two colors.

Coloring in Two Colors

- A special case of vertex coloring is *2-coloring,* or coloring in two colors.
- While in general coloring graphs is algorithmically hard, a 2-coloring can be constructed by a *greedy algorithm*.

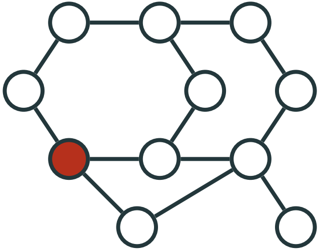
Coloring in Two Colors

- A special case of vertex coloring is *2-coloring,* or coloring in two colors.
- While in general coloring graphs is algorithmically hard, a 2-coloring can be constructed by a *greedy algorithm*.



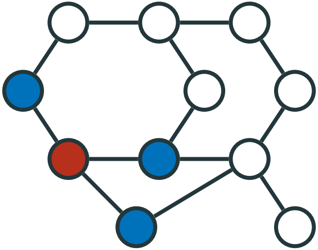
Coloring in Two Colors

- A special case of vertex coloring is *2-coloring,* or coloring in two colors.
- While in general coloring graphs is algorithmically hard, a 2-coloring can be constructed by a *greedy algorithm*.



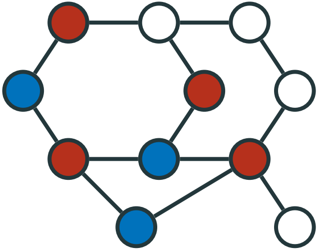
Coloring in Two Colors

- A special case of vertex coloring is *2-coloring,* or coloring in two colors.
- While in general coloring graphs is algorithmically hard, a 2-coloring can be constructed by a *greedy algorithm*.



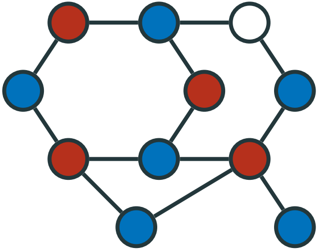
Coloring in Two Colors

- A special case of vertex coloring is *2-coloring,* or coloring in two colors.
- While in general coloring graphs is algorithmically hard, a 2-coloring can be constructed by a *greedy algorithm*.



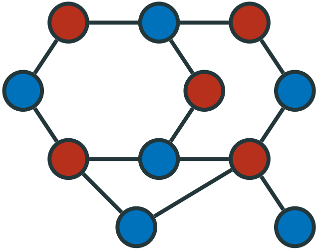
Coloring in Two Colors

- A special case of vertex coloring is *2-coloring,* or coloring in two colors.
- While in general coloring graphs is algorithmically hard, a 2-coloring can be constructed by a *greedy algorithm*.



Coloring in Two Colors

- A special case of vertex coloring is *2-coloring,* or coloring in two colors.
- While in general coloring graphs is algorithmically hard, a 2-coloring can be constructed by a *greedy algorithm*.

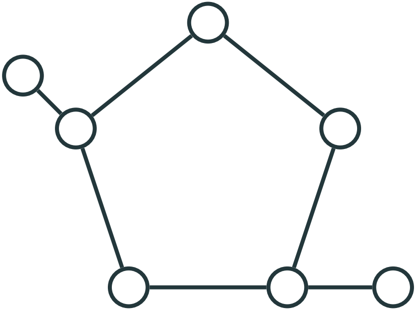


Coloring in Two Colors

- The greedy algorithm is not always successful.

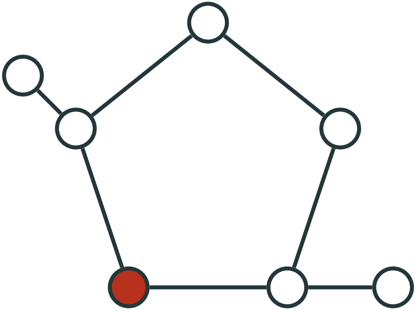
Coloring in Two Colors

- The greedy algorithm is not always successful.



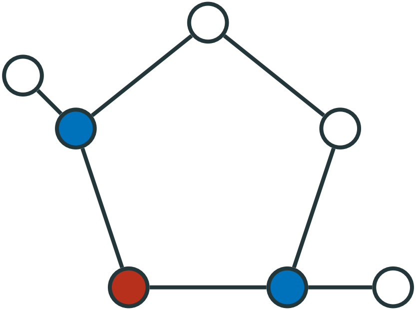
Coloring in Two Colors

- The greedy algorithm is not always successful.



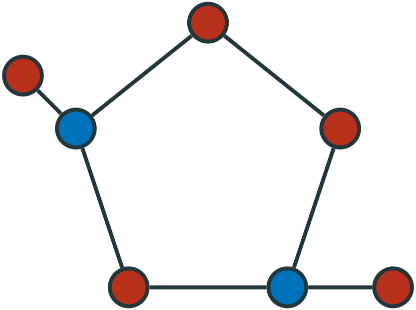
Coloring in Two Colors

- The greedy algorithm is not always successful.



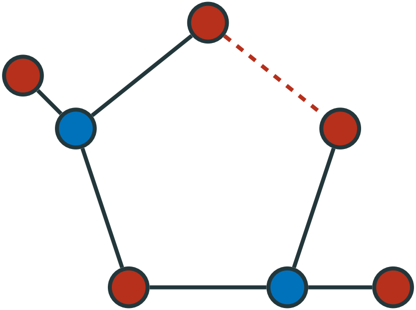
Coloring in Two Colors

- The greedy algorithm is not always successful.



Coloring in Two Colors

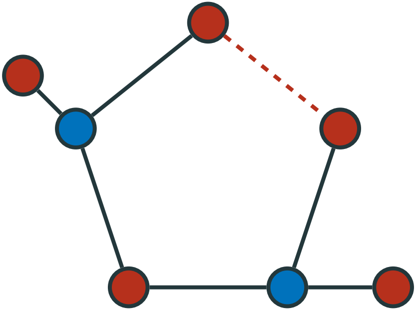
- The greedy algorithm is not always successful.



- However, if the greedy algorithm fails, there is a cycle with an odd number of vertices.

Coloring in Two Colors

- The greedy algorithm is not always successful.



- However, if the greedy algorithm fails, there is a cycle with an odd number of vertices.
- For such a graph, 2-coloring is impossible.

Bipartite Graphs

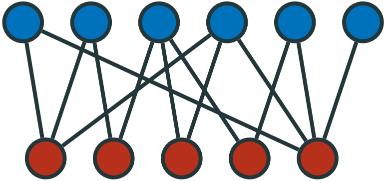
- 2-colorable graphs are also called *bipartite*.

Bipartite Graphs

- 2-colorable graphs are also called *bipartite*.
- Vertices in a bipartite graph can be split into two parts such that edges go only between parts.

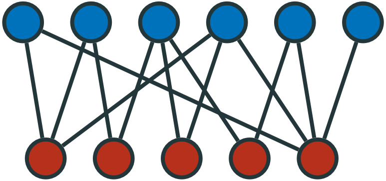
Bipartite Graphs

- 2-colorable graphs are also called *bipartite*.
- Vertices in a bipartite graph can be split into two parts such that edges go only between parts.
- Example:



Bipartite Graphs

- 2-colorable graphs are also called *bipartite*.
- Vertices in a bipartite graph can be split into two parts such that edges go only between parts.
- Example:



- Bipartite graphs can express *correspondences* between objects of different kind (say, students and courses they attend).