Data Structures II : Greedy graph algorithms

Mauricio Toro
Department of Systems and Informatics
Universidad EAFIT

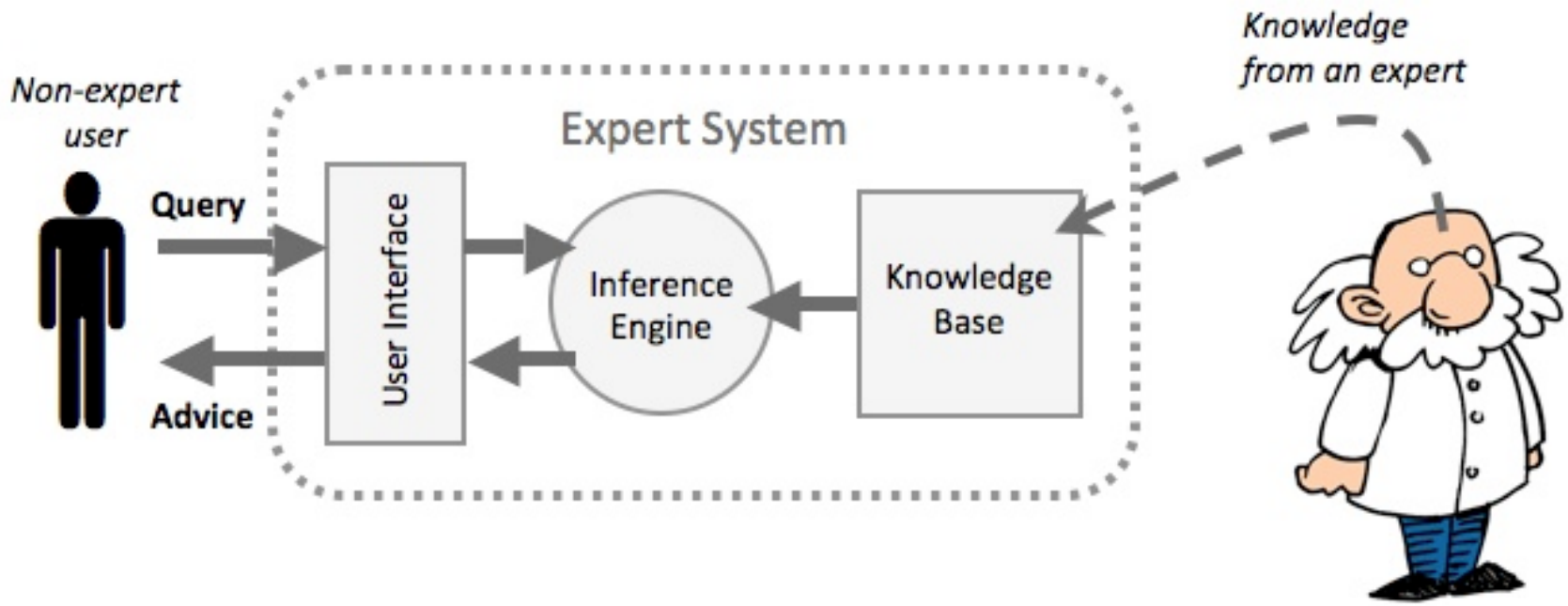
Cocktail of the day: Cosmopolitan

Disclaimer: Keep alcohol out of the hands of minors.

Cocktail of the day: Cosmopolitan

- 40 ml Vodka
- 15 ml Cointreau
- 15 ml Lime juice
- 30 ml Cranberry juice

Expert systems use graphs



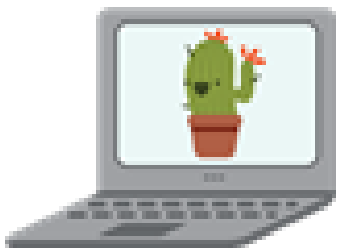
https://www.youtube.com/watch?v=uWEahgy3Iyc

Single-source shortest path problems

The problem is to determine the cost of the shortest path from the source to every other vertex in V, where the length of a path is just the sum of the costs of the arcs on the path.

Dijkstra's Algorithm

- A greedy algorithm to solve the single-source shortest path problem.
- Dijkstra's algorithm does not work when weights on the arcs are negative numbers.
- Algorithm can be found here: `http://en.wikipedia.org/wiki/Dijkstra%27s_algorithm`



Example of Dijkstra's Algorithm

Simulator:
`https://www.cs.usfca.edu/~galles/visualization/Dijkstra.html`

Running time over adjacency matrices

- Suppose Dijkstra's algorithm operates on a digraph with n vertices and e edges.
- If we use an adjacency matrix to represent the digraph, then the inner loop takes $O(n)$ time, and it is executed n-1 times for a total time of $O(n^2)$.
- The rest of the algorithm is easily seen to require no more time than this.

Running time over adjacency lists

http://www.geeksforgeeks.org/greedy-algorithms-set-7-dijkstras-algorithm-for-adjace

Figure: Taken from Inc. [Inc13]

Multiple-source shortest path problem

- Floyd-Warshall's algorithm.
- What does Floyd-Warshall algorithms does?
- What is the complexity of Floyd-Warshall algorithm?

Minimum-Cost Spanning Trees

- Suppose $G = (V, E)$ is a connected graph in which each edge $(u, v) \in E$ has a cost $c(u, v)$ attached to it.
- A spanning tree for G is a free tree that connects all the vertices in V.
- The cost of a spanning tree is the sum of the costs of the edges in the tree.
- Prim's algorithm finds a minimum-cost spanning tree.

Taken from [Aho77].

Minimum-Cost Spanning Trees

- Suppose $G = (V, E)$ is a connected graph in which each edge $(u, v) \in E$ has a cost $c(u, v)$ attached to it.
- A spanning tree for G is a free tree that connects all the vertices in V.
- The cost of a spanning tree is the sum of the costs of the edges in the tree.
- Prim's algorithm finds a minimum-cost spanning tree.

Taken from [Aho77].

Minimum-Cost Spanning Trees

- Suppose $G = (V, E)$ is a connected graph in which each edge $(u, v) \in E$ has a cost $c(u, v)$ attached to it.
- A spanning tree for G is a free tree that connects all the vertices in V.
- The cost of a spanning tree is the sum of the costs of the edges in the tree.
- Prim's algorithm finds a minimum-cost spanning tree.

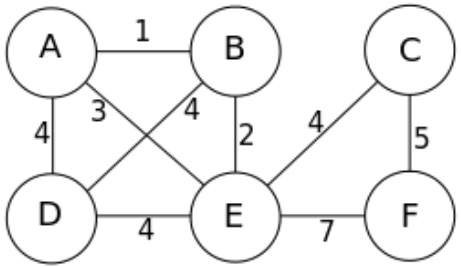
Taken from [Aho77].

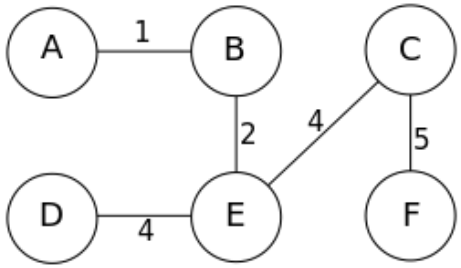
Minimum-Cost Spanning Trees

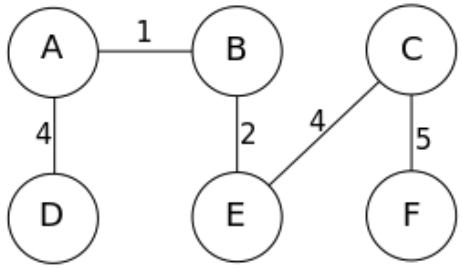
- Suppose $G = (V, E)$ is a connected graph in which each edge $(u, v) \in E$ has a cost $c(u, v)$ attached to it.
- A spanning tree for G is a free tree that connects all the vertices in V.
- The cost of a spanning tree is the sum of the costs of the edges in the tree.
- Prim's algorithm finds a minimum-cost spanning tree.

Taken from [Aho77].

Minimum-Cost Spanning Tree Examples







Prim's Algorithm

- Input: A non-empty connected weighted graph with vertices V and edges E (the weights can be negative).
- Initialize: $V_{new} = \{x\}$, where x is an arbitrary node (starting point) from V, $E_{new} = \{\}$
- Repeat until $V_{new} = V$:
  1. Choose an edge $\{u, v\}$ with minimal weight such that u is in V_{new} and v is not (*if there are multiple edges with the same weight, any of them may be picked*).
  2. Add v to V_{new}, and $\{u, v\}$ to E_{new}.
- Output: V_{new} and E_{new} describe a minimal spanning tree.

Taken from `http://en.wikipedia.org/wiki/Prim%27s_algorithm`

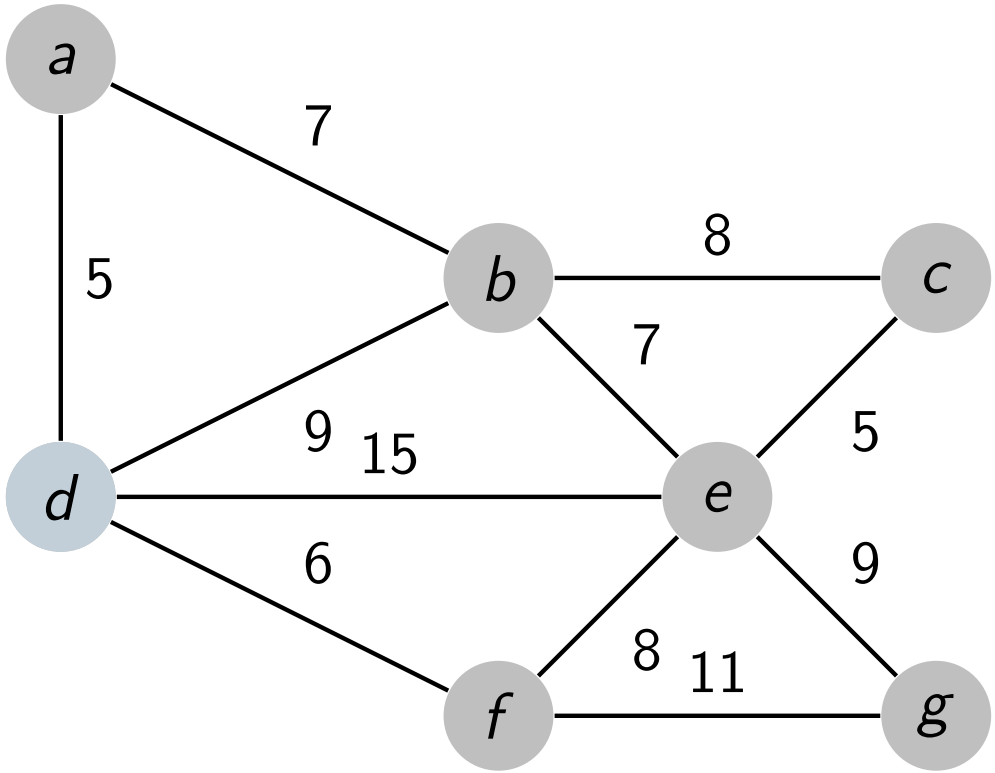
Example of Prims's Algorithm

- `https://www.cs.usfca.edu/~galles/visualization/Prim.html`

Running time of Prim's algorithm

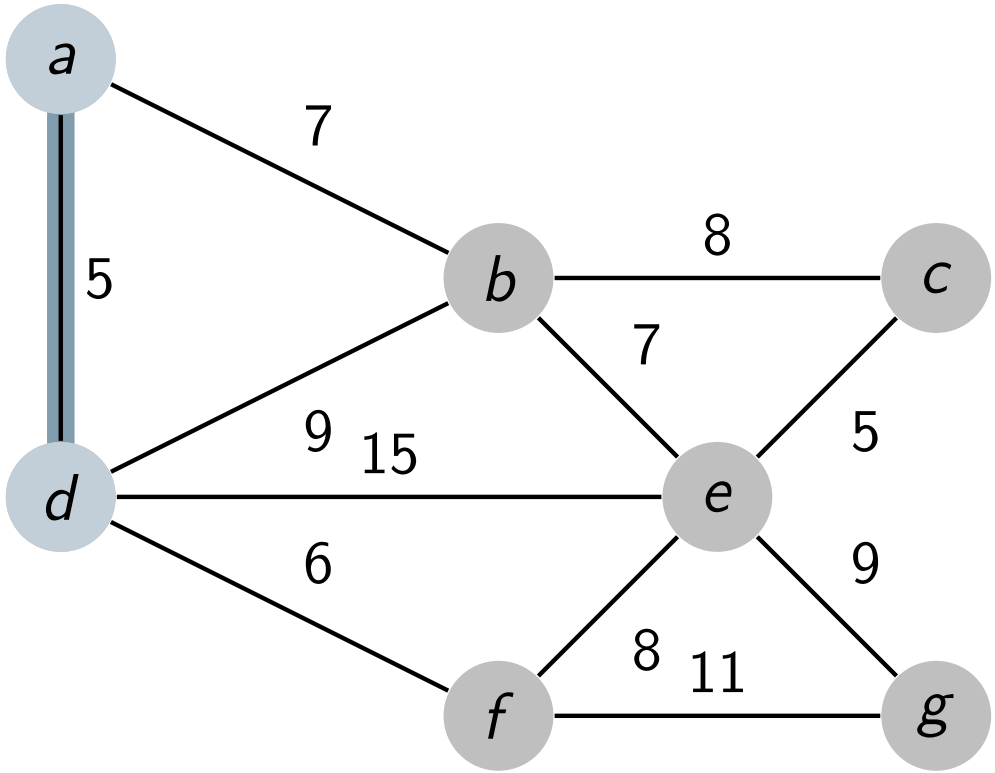
- Suppose Prim's algorithm operates on a graph with V vertices and E edges.
- The complexity of Prim's algorithm is $O(V^2)$. Why?

Prim's algorithm



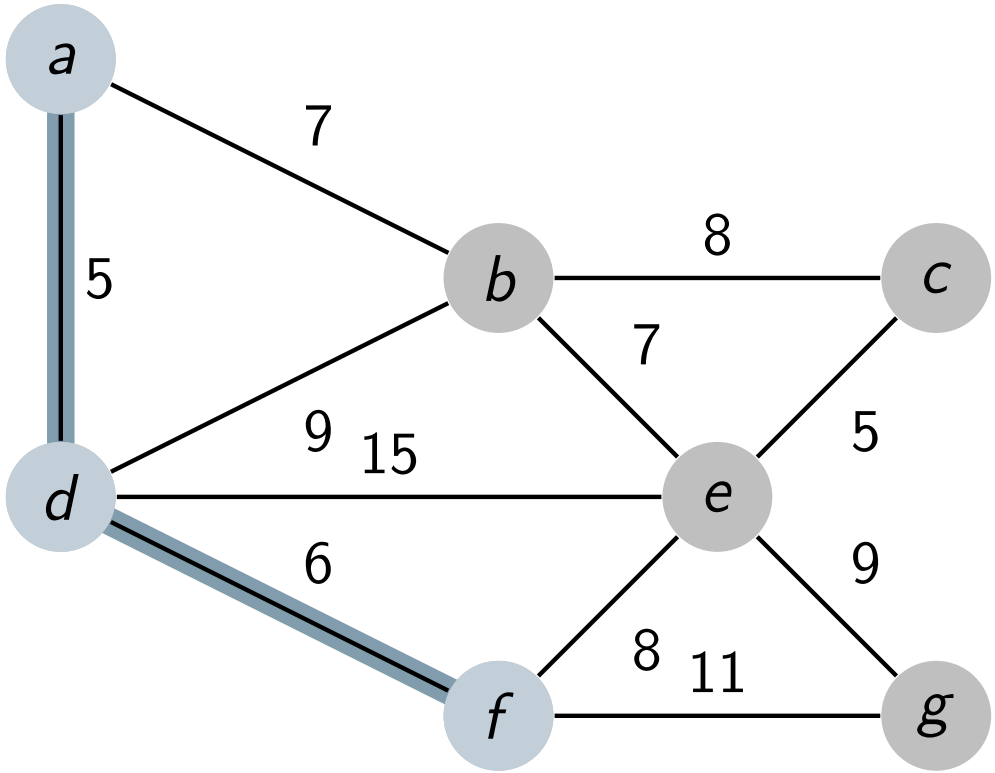
http://www.texample.net/tikz/examples/prims-algorithm/

Prim's algorithm



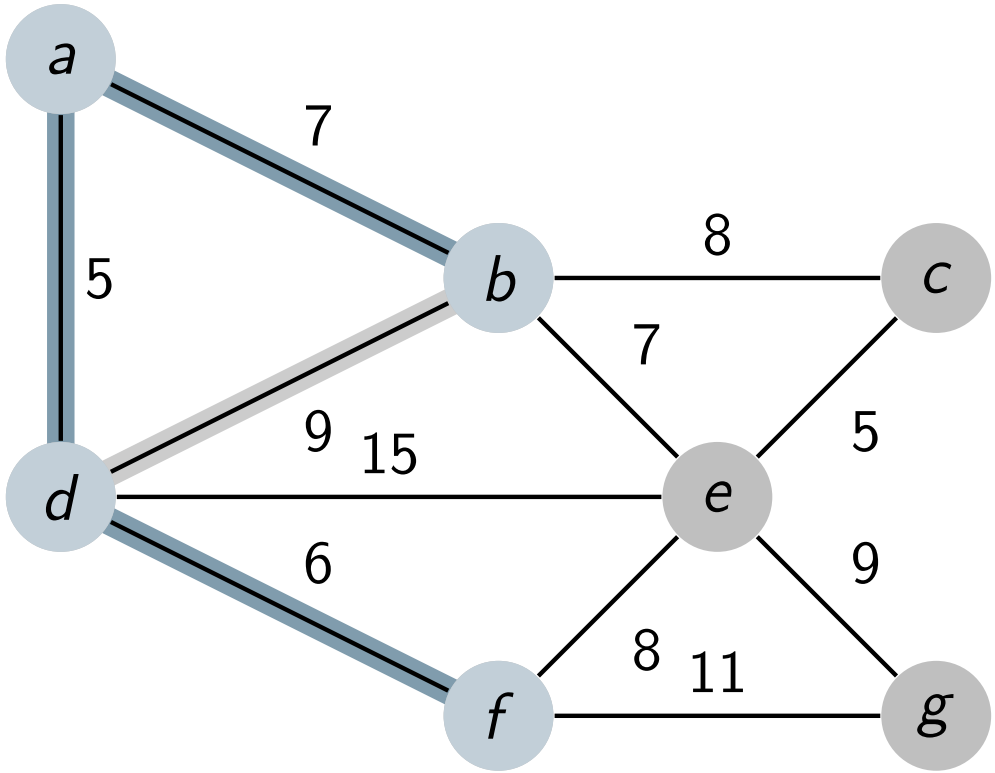
http://www.texample.net/tikz/examples/prims-algorithm/

Prim's algorithm



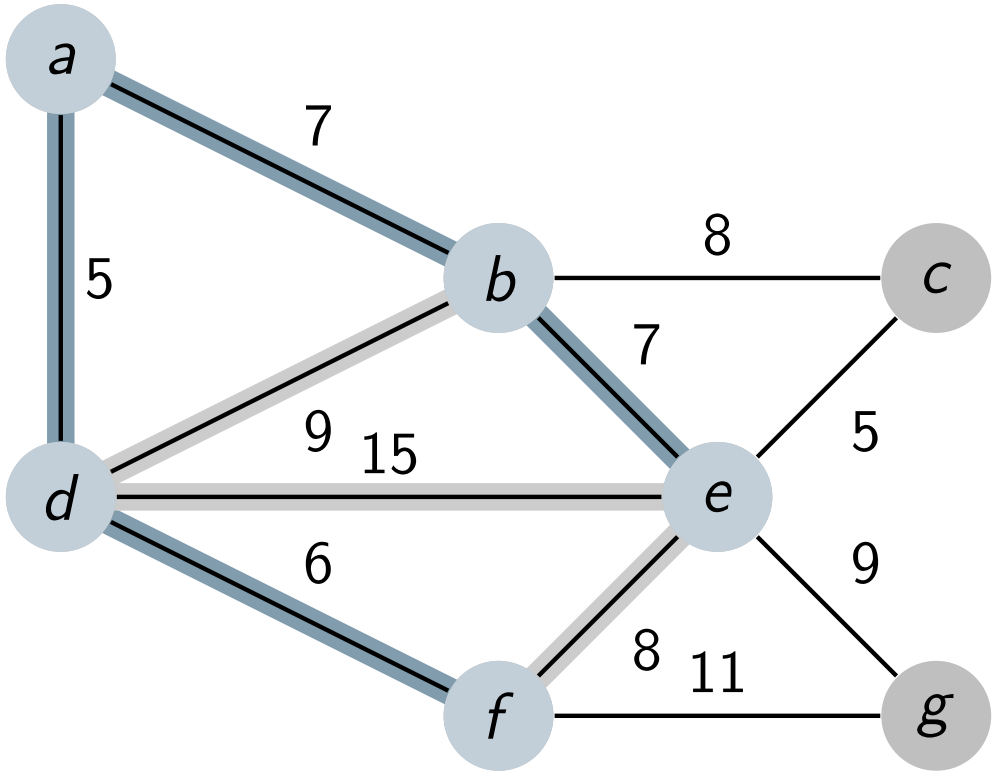
http://www.texample.net/tikz/examples/prims-algorithm/

Prim's algorithm



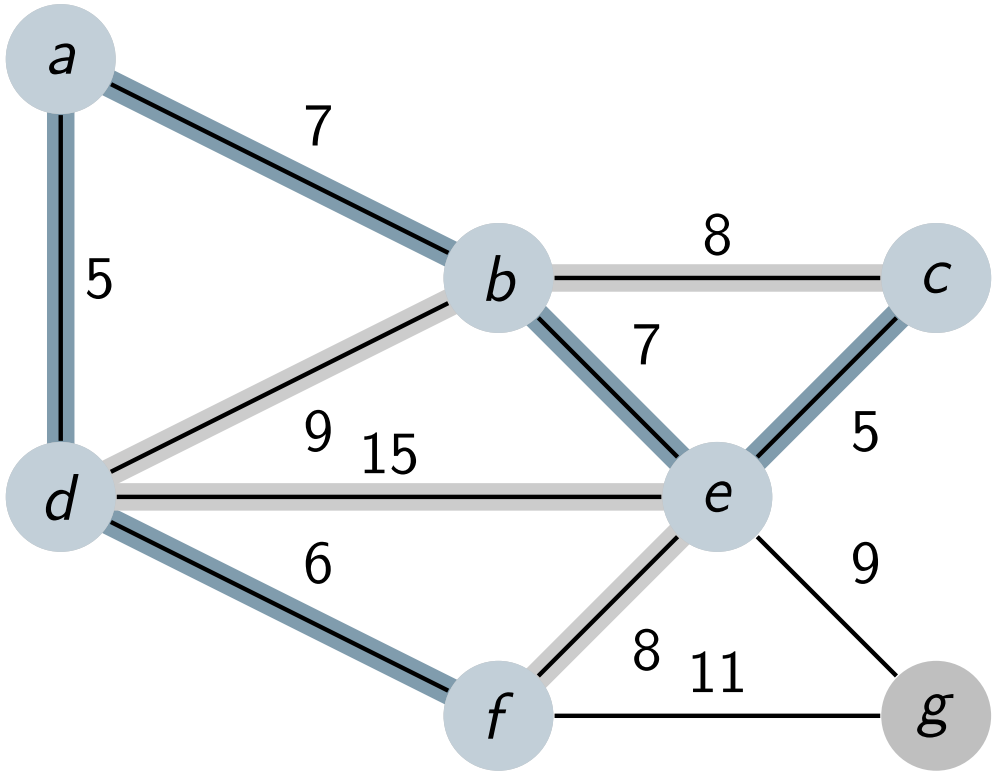
http://www.texample.net/tikz/examples/prims-algorithm/

Prim's algorithm



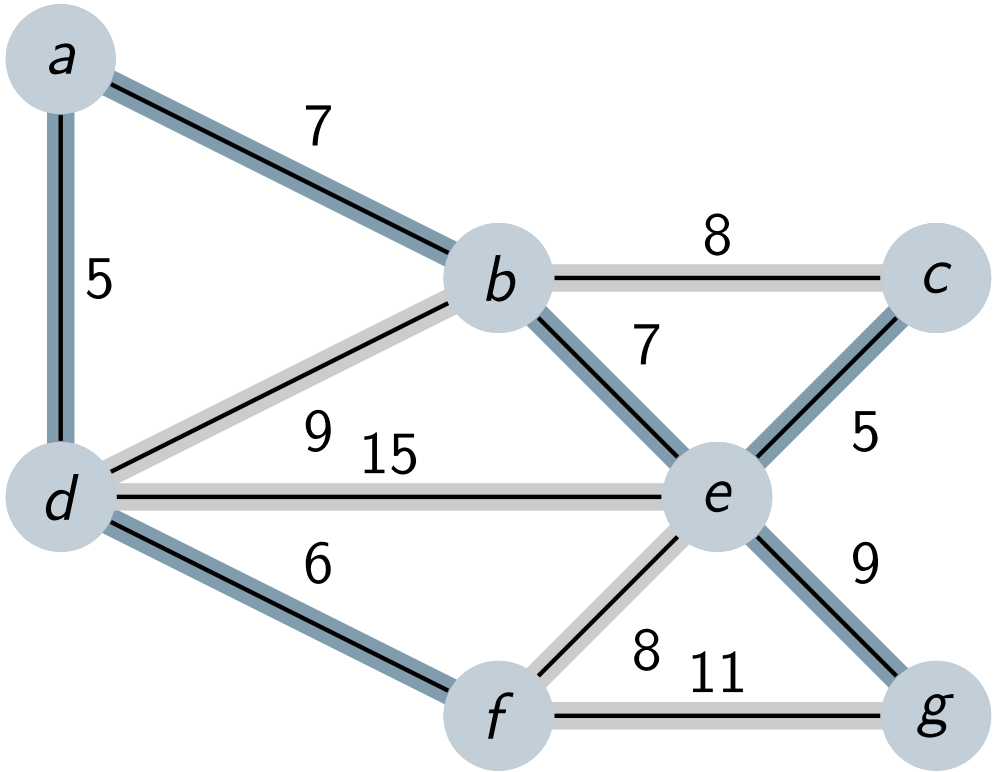
http://www.texample.net/tikz/examples/prims-algorithm/

Prim's algorithm



http://www.texample.net/tikz/examples/prims-algorithm/

Prim's algorithm



http://www.texample.net/tikz/examples/prims-algorithm/

Kruskal's algorithm

- We examine edges from E, in order of increasing cost.
- If the edge connects two vertices in two different connected components, then we add the edge to T.
- If the edge connects two vertices in the same component, then we discard the edge. Why?
- When all vertices are in one component, T is a minimum-cost spanning tree for G.

Taken from [Aho77].

Kruskal's algorithm

- We examine edges from E, in order of increasing cost.
- If the edge connects two vertices in two different connected components, then we add the edge to T.
- If the edge connects two vertices in the same component, then we discard the edge. Why?
- When all vertices are in one component, T is a minimum-cost spanning tree for G.

Taken from [Aho77].

Examples of Kruskal's algorithm

- https://www.cs.usfca.edu/~galles/visualization/Kruskal.html
- https://www.youtube.com/watch?v=71UQH7Pr9kU

Running time of Kruskal's algorithm

- Suppose Kruskal's algorithm operates on a graph with V vertices and E edges.
- The complexity of Kruskal's algorithm is $O(E.log(E))$. Why?

Kruskal Vs Prim

- The time complexity of Prim's algorithm is $O(V^2)$.
- As V gets large the performance of this algorithm may become unsatisfactory.
- The complexity of Krukal's algorithm is $O(E.log(E))$.
- If E is much less than V^2, Kruskal's algorithm is superior, although if E is about V^2, we would prefer Prim's algorithm.

Taken from [Aho77].

Kruskal Vs Prim

- The time complexity of Prim's algorithm is $O(V^2)$.
- As V gets large the performance of this algorithm may become unsatisfactory.
- The complexity of Krukal's algorithm is $O(E.log(E))$.
- If E is much less than V^2, Kruskal's algorithm is superior, although if E is about V^2, we would prefer Prim's algorithm.

Taken from [Aho77].

Quiz Questions

- An algorithm for s.c.s.p is Dijkstra's algorithm
- Running time of Dijkstra's algorithm (over an adjacency matrix) is $O(n^2)$
- Algorithms to find a minimum spaning tree are Prim's algorithm and Kruskal's algorithm
- Running time of Prim's algorithm is $O(V^2)$; running time of Kruskal's is $O(E.log(E))$
- If E is much less than V^2, Kruskal's algorithm is superior, although if E is about V^2, we would prefer Prim's algorithm.

Quiz Questions

- An algorithm for s.c.s.p is Dijkstra's algorithm
- Running time of Dijkstra's algorithm (over an adjacency matrix) is $O(n^2)$
- Algorithms to find a minimum spaning tree are Prim's algorithm and Kruskal's algorithm
- Running time of Prim's algorithm is $O(V^2)$; running time of Kruskal's is $O(E.log(E))$
- If E is much less than V^2, Kruskal's algorithm is superior, although if E is about V^2, we would prefer Prim's algorithm.

Quiz Questions

- An algorithm for s.c.s.p is Dijkstra's algorithm
- Running time of Dijkstra's algorithm (over an adjacency matrix) is $O(n^2)$
- Algorithms to find a minimum spaning tree are Prim's algorithm and Kruskal's algorithm
- Running time of Prim's algorithm is $O(V^2)$; running time of Kruskal's is $O(E.log(E))$
- If E is much less than V^2, Kruskal's algorithm is superior, although if E is about V^2, we would prefer Prim's algorithm.

Quiz Questions

- An algorithm for s.c.s.p is Dijkstra's algorithm
- Running time of Dijkstra's algorithm (over an adjacency matrix) is $O(n^2)$
- Algorithms to find a minimum spaning tree are Prim's algorithm and Kruskal's algorithm
- Running time of Prim's algorithm is $O(V^2)$; running time of Kruskal's is $O(E.log(E))$
- If E is much less than V^2, Kruskal's algorithm is superior, although if E is about V^2, we would prefer Prim's algorithm.

References

- Please how to reference images, trademarks, videos and fragments of code.
- Avoid plagiarism

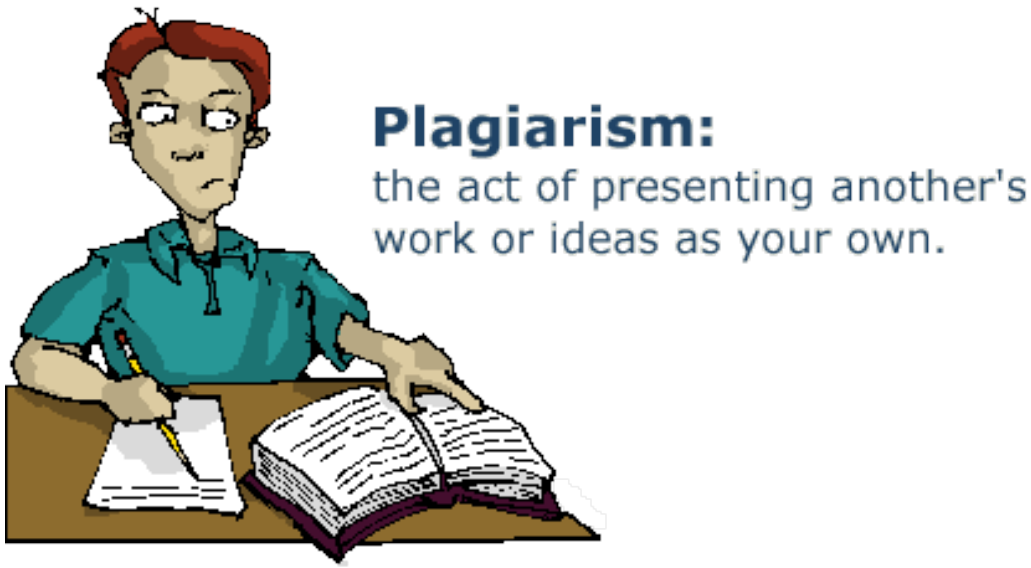


Figure: Figure about plagiarism, University of Malta [Uni09]

References I

Inc.

Questions Figure — Small Business Ideas and Resources for Entrepreneurs, 2013.

[Online; accessed 29-November-2013].

University of Malta.

Plagarism — The act of presenting another's work or ideas as your own, 2009.

[Online; accessed 29-November-2013].

Further reading

- R.C.T Lee, Introduction to the analysis and design of algorithms, Chapter 3, Pages 71 - 115.
- `http://www.geeksforgeeks.org/greedy-algorithms-set-7-dijkstras-algorithm-for-ad`