Tutorial 11
Graph algorithms

Learning Objectives

> In this tutorial you will apply graph algorithms and compute their complexity.
>
> These problems relate to the following course learning objectives: *Select and justify appropriate tools to analyze a counting problem* and *Analyze a counting problem by proving an exact or approximate enumeration, or a method to compute one efficiently.*

1 Kruskal's algorithm

In class we covered Prim's Algorithm, a greedy algorithm that quickly finds a minimal weight spanning tree by starting at an arbitrary vertex and adding vertices one by one, always adding a vertex which connects to the already added ones by an edge of least weight. In this way, it *builds a tree* starting from a vertex.

Kruskal's algorithm uses a different idea: In our spanning tree, we want to have edges with minimal weights, provided they don't create cycles.

Given a weighted graph $G = (V, E, w)$, Kruskal's algorithm proceeds by first sorting the edges according to their weight, we number the edges $e_1, \ldots, e_k$ such that $w(e_1) \leq w(e_2) \leq \ldots \leq w(e_k)$. We will inductively build the spanning tree $T = (V, S)$. We initialize $S = \emptyset, i = 0$. Then we repeat the following inductive step:

While $|S| < n - 1$, let j be the least non-negative integer such that $j > i$ and there are no cycles in $S \cup \{e_j\}$. Then set $i = j$ and $S = S \cup \{j\}$.

1. Consider the graph on Figure 1.

   (a) Use Kruskal's algorithm to find a minimal spanning tree

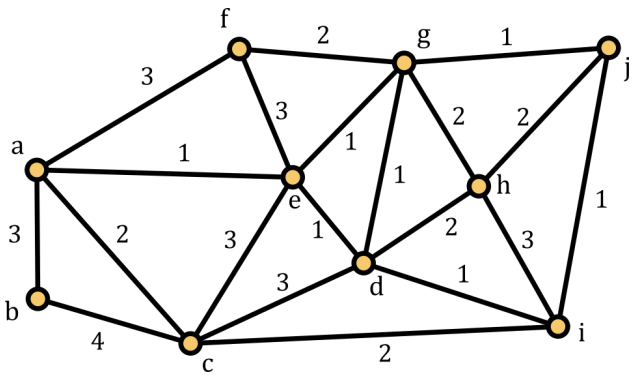

Figure 1: A weighted graph

   (b) Let a basic operation be: "check if a given edge creates a cycle". At most how many such operations will Kruskal's algorithm perform on a graph with n vertices? Try to come up with a worst-case example.

2 Uploading and downloading

2. Figure 2 shows a computer network of computers. You have a large amount of data on computer a. The connections between the computers vary in quality, and it will take a certain number of minutes (indicated next to the edges) to transfer the data from one computer to the other.

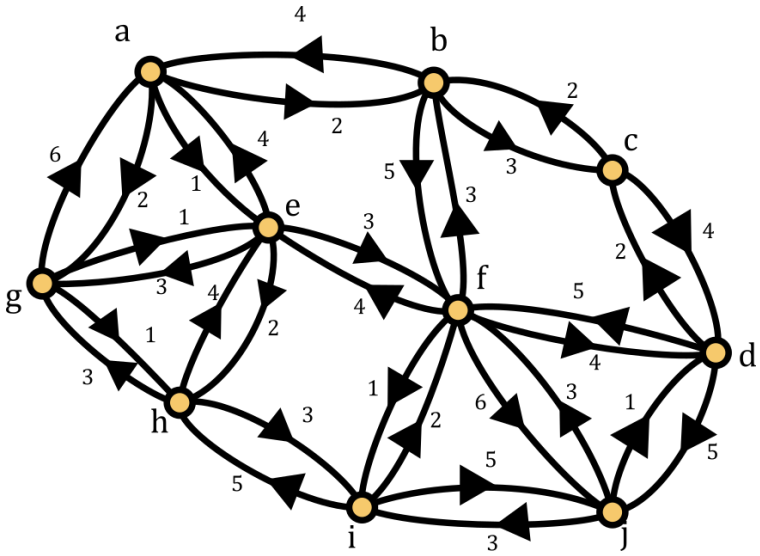

Figure 2: A network of computers

(a) What is the quickest way for you to upload your data to computer j? Find the amount of time required and the quickest path.

(b) A colleague of yours says that they also need the file at computer c. Now it is on both computers a and j. Which one should they download the file from?

(c) Another colleague at computer j does some work on your file, and now you want to download it (to computer a). What is the minimal amount of time required, and the quickest?

(d) Let a basic operation be "updating a δ-value for a vertex" in Dijkstra's algorithm. At most how many operations will Dijkstra's algorithm perform if we want to use it to find the distance between two given vertices in a directed graph on n vertices? How about if we want to find the distance from one vertex to all the other ones?

(e) How about if we want to find all pairwise distances in a directed graph using Dijkstra's algorithm?

1. (a) One solution is on Figure 3

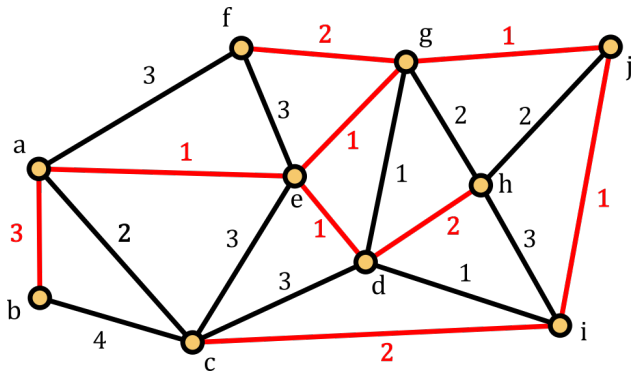

Figure 3: A minimal spanning tree

   (b) If G a complete graph with n vertices, with all weights equal to 1, and the binary sort is unlucky, we may need to perform this check $\binom{n}{2} = O(n^2)$ times.

2. (a) The distance is 10 and the path is (a, e, f, i, j).

   (b) We already found that the distance from a to c is 5. We see quickly that the distance from j to c is at most 3 (through d). So they should download it from j.

   (c) For this, we need to look at download speeds, so we look at the edges which point towards a to find paths. The download distance to j is 9, through (a, e, f, d, j).

   (d) In both cases, the complexity is $O(n^2)$.

   (e) We perform the algorithm n times, so the complexity is $O(n^3)$.