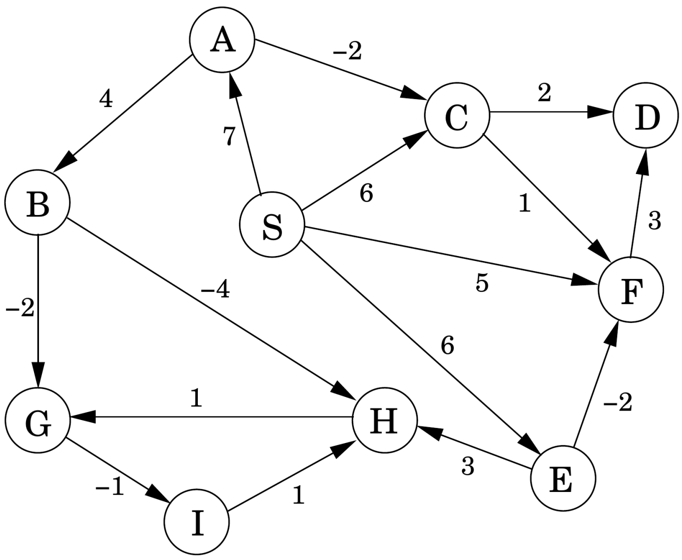
5. tutorial

1. Dijkstra fail. Construct a graph with integral lengths (without a negative cycle) where Dijkstra's algorithm would run exponentially long. (This is a variant of the algorithm which, if it relaxes a vertex v and changes the estimate for its neighbor u, it opens the neighbor, that is, puts it in the priority queue even if it extracted it previously.)

2. Negative edges $+k$? Can we get rid of negative edges by adding some number $k \in \mathbb{N}$ to all edge lengths, so that we can use Dijkstra's algorithm to find the shortest path even in a graph with negative edges?

3. Generalized Shortest Paths. You are given a directed graph $G = (V, E)$ with edge lengths $\ell : E \to \mathbb{R}$ and vertex costs $c : V \to R$. For instance, imagine that this is a computer network and there is not only a delay for passing through a network cable, but also a delay incurred by passing through a router. The length of a path $v_0, e_0, v_1, e_1, \ldots, e_k, v_{k+1}$ is $(\sum_{i=0}^{k} c(v_i)) + (\sum_{i=0}^{k+1} \ell(e_i))$. How to find the shortest paths from $s \in V$ to all other nodes in G?

4. Bellman-Ford. Suppose Bellman-Ford's algorithm is run on the following graph from node A:



Draw a table displaying the intermediate values h (the distance estimates for the nodes), and draw the final shortest-path tree.

5. Paths with at most k edges. A graph G with possibly negative edge lengths is given, and you are told that all shortest paths in G contain at most k edges. Construct an algorithm which, in time $\mathcal{O}(k|E|)$, solves the single-source shortest paths problem (for a given s, finds the shortest paths to all other vertices of G).

6. Shortest-paths Tree? You are given a directed graph $G = (V, E)$ with (possibly negative) weighted edges, along with a specific node $s \in V$ and a tree $T = (V, E_0)$, $E_0 \subseteq E$. Give an algorithm that checks whether T is a shortest-path tree for G with starting point s. Your algorithm should run in linear time.

7. All-pairs, through v_0. You are given a strongly connected directed graph $G = (V, E)$ with positive edge weights along with a particular node $v_0 \in V$. Give an efficient algorithm for finding shortest paths between all pairs of nodes, with the one restriction that these paths must all pass through v_0.

8. Currency Exchange. A currency exchange trades n currencies (say that currency number one is CZK) and it announces a matrix of exchange rates K. The exchange rate K_{ij} says how many units of currency j we get for one unit of currency i. Design an algorithm which decides whether there exists a sequence of currency exchanges which begins with 1 CZK and ends with >1 CZK.

9. Dijkstra with Small Lengths. Suppose we want to run Dijkstra's algorithm on a graph whose edge weights are integers in the range $0, 1, \ldots, W$, where W is a relatively small number.

(1) Show how Dijkstra's algorithm can be made to run in time $\mathcal{O}(W|V| + |E|)$.
(2) Show an alternative implementation that takes time just $\mathcal{O}((|V| + |E|) \log W)$.

10. Floyd-Warshall and Shortest Cycle. Modify Floyd-Warshall's algorithm to detect, for each vertex v, the shortest cycle in G containing v. (Assume G has no negative cycles.)

11. Edges on Shortest Paths. Construct an algorithm which detects all edges lying on at least one shortest path from s to any vertex, or from s to a particular vertex t.

12. Critical Edges. Say that an edge e is *critical for s* if removing it from G changes the distances from $s \in V$, that is, there exists a v such that $d_G(s,v) < d_{G-e}(s,v)$. Design an algorithm which detects all edges critical for s.