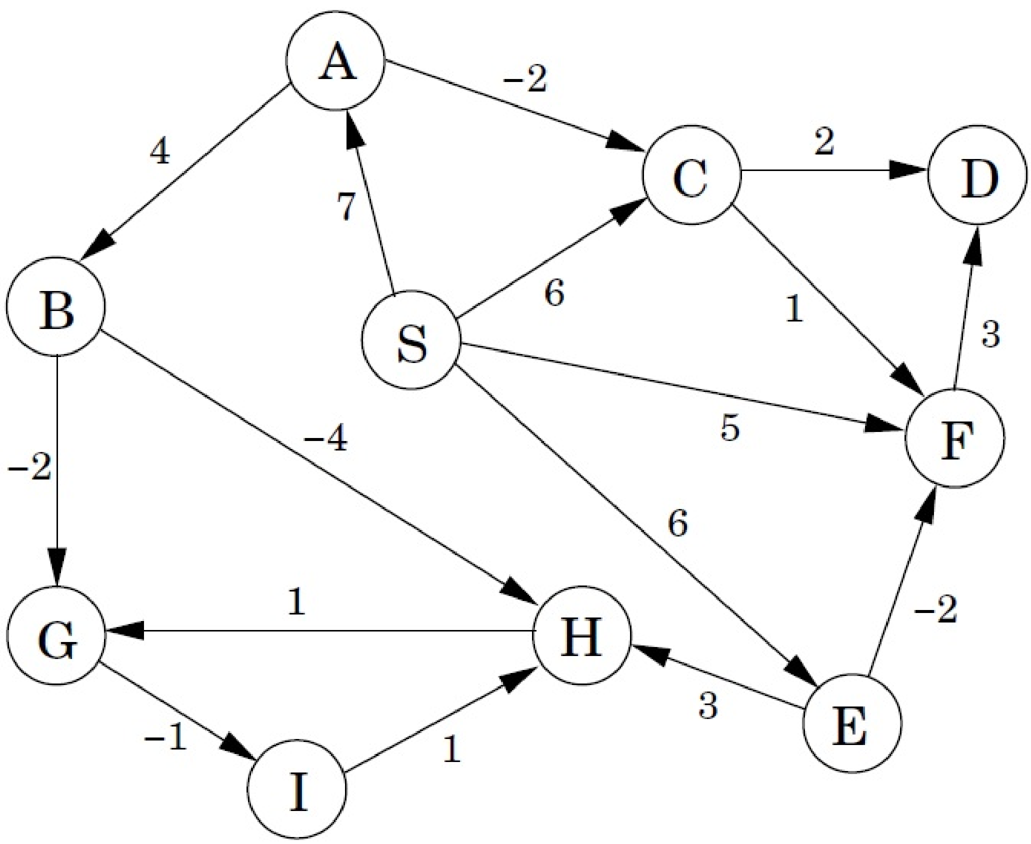
Weekly Assignment 5:
Bellman-Ford and Flow algorithms

October 2022

1. Run the Bellman-Ford algorithm on the following directed graph **with source** A. Pick edges in lexicographic order (so (A, B) before (A, C), (G, I) before (H, G), etc).



2. Decide whether the following statement is true or false. If it is true, give a short explanation. If it is false, give a counterexample.

   Let G be an arbitrary flow network, with a source s, a sink t, and positive integer capacity $c(e)$ on every edge e. Let (A, B) be a minimum $s - t$ cut with respect to these capacities. Now suppose we add 1 to every capacity; then (A, B) is still a minimum $s - t$ cut with respect to the new capacities.

3. Adapt the Bellman-Ford algorithm in such a way that it enumerates all the vertices v for which there is a cycle with negative weight on some path from the source to v. Explain your solution.

4. Suppose someone claims that a function $f : E \to \mathbb{N}$ is a max-flow for network $G = (V, E)$ and capacity function c. Develop a linear time, i.e. in $O(|V| + |E|)$, algorithm to determine whether this claim is correct.

5. The Bellman-Ford presented during the lecture initially explores several edges (u, v) with $d[v] = \infty$. This is clearly inefficient. In order to improve the algorithm, one student has the following proposal:

   > Run a BFS starting from source s, and then run Bellman-Ford where (in each iteration) the order in which edges are visited is given by their distance from s as computed by the BFS. This way we first perform a relaxation step for all the outgoing transitions of s, then we do a relaxation step for all outgoing edges of vertices that can be reached via a single edge from s, etc.
   >
   > We stop the algorithm when after an iteration (i.e. after visiting all the edges) not a single d-value has been updated.

   Show that this modified algorithm still has the same time complexity for graphs without negative weight cycles