CSOR W4231.001–Spring, 2021
Eleni Drinea

Homework 5 (130 points)

Out: Friday, March 26, 2021
Due: 11:59pm, Monday, April 12, 2021

Homework Instructions.

1. For all algorithms that you are asked to "give" or "design", you should
   - Describe your algorithm clearly in English.
   - Give pseudocode.
   - Argue correctness.
   - Give the best upper bound that you can for the running time.
2. If you give a reduction, you should do so as we did in class, that is
   (a) Give the inputs to the two problems.
   (b) Describe in English the reduction transformation and argue that it requires polynomial time. (You do not need to give pseudocode.)
   (c) Prove carefully equivalence of the original and the reduced instances.
3. **You should not use any external resources for this homework.** Failure to follow this instruction will have a negative impact on your performance in the exam (and possibly in interviews). For the same reason, you should avoid collaborating with your classmates, at least not before you have thought through the problems for a while on your own. I also encourage you to work on all the recommended exercises.
4. You should submit this assignment as a **pdf** file on Gradescope. Other file formats will not be graded, and will automatically receive a score of 0.
5. I recommend you type your solutions using LaTeX. For every assignment, you will earn 5 extra credit points if you type your solutions using LaTeX or other software that prints equations and algorithms neatly. If you do not type your solutions, make sure that your hand-writing is very clear and that your scan is high quality.
6. You should write up the solutions **entirely on your own**. Collaboration is limited to discussion of ideas only. You should adhere to the department's academic honesty policy (see the course syllabus). Similarity between your solutions and solutions of your classmates or solutions posted online will result in receiving a 0 in this assignment, and possibly further disciplinary actions. There will be no exception to this policy and it may be applied retro-actively if we have reasons to re-evaluate this homework.

Homework Problems

1. (30 points) A *flow network with demands* is a directed capacitated graph with potentially multiple sources and sinks, which may have incoming and outgoing edges respectively. In particular, each node $v \in V$ has an integer *demand* $d(v)$; if $d(v) > 0$, v is a *sink*, while if $d(v) < 0$, it is a *source*. Let S be the set of source nodes and T the set of sink nodes.

   A *circulation with demands* is a function $f : E \to R^+$ that satisfies

   (a) *capacity constraints:* for each $e \in E$, $0 \leq f(e) \leq c(e)$.

   (b) *demand constraints:* For each $v \in V$, $f^{\text{in}}(v) - f^{\text{out}}(v) = d(v)$.

   We are now concerned with a decision problem rather than a maximization one: *is there* a circulation f with demands that meets both capacity and demand conditions?

   i. Derive a necessary condition for a feasible circulation with demands to exist.

   ii. Reduce the problem of finding a feasible circulation with demands to Max Flow.

2. (30 points) In many applications, on top of the node demands introduced in the previous problem, the traffic must also make use of certain edges. To capture such constraints, consider the following variant of the previous problem.

   You are given a flow network $G = (V, E)$ with demands where every edge e has an integer capacity c_e, *and* an integer *lower bound* $\ell_e \geq 0$. A circulation f must now satisfy $\ell_e \leq f(e) \leq c_e$ for every $e \in E$, as well as the demand constraints. Determine whether a feasible circulation exists.

3. (15 points) `Stingy SAT` is the following problem: on input a `SAT` formula ϕ and an integer k, is there a satisfying assignment for ϕ in which at most k variables are `True`?

   Prove that `Stingy SAT` is $\mathcal{NP}$-complete.

4. (25 points) Suppose you had a polynomial-time algorithm that, on input a graph, answers **yes** if and only if the graph has a Hamiltonian cycle.

   Show how, on input a graph $G = (V, E)$, you can return in polynomial time

   - a Hamiltonian cycle in G, if one exists,
   - **no**, if G does not have a Hamiltonian cycle.

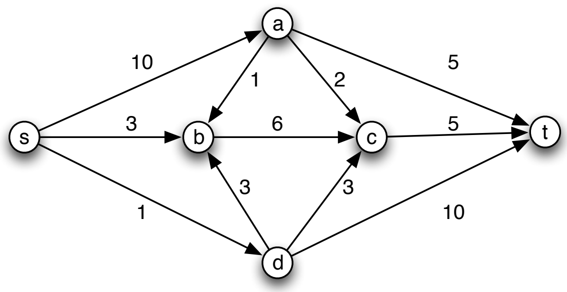
5. (30 points) There is a set of ground elements $E = \{e_1, e_2, \ldots, e_n\}$ and a collection of m subsets $S_1, S_2, \ldots, S_m$ of the ground elements (that is, $S_i \subseteq E$ for $1 \leq i \leq m$).

   The goal is to select a minimum cardinality set A of ground elements such that A contains at least one element from each subset S_i.

   Give a polynomial time algorithm for this problem or state its decision version and prove that it is $\mathcal{NP}$-complete.

RECOMMENDED exercises: do NOT return, they will not be graded.)

1. Run the Ford-Fulkerson algorithm on the following network, with edge capacities as shown, to compute the max s-t flow. At every step, draw the residual graph and the augmenting paths. Report the maximum flow along with a minimum cut.



2. There are many variations on the maximum flow problem. For the following two natural generalizations, show how to solve the more general problem by **reducing** it to the original max-flow problem (thereby showing that these problems also admit efficient solutions).

   - There are multiple sources and multiple sinks, and we wish to maximize the flow between all sources and sinks.
   - Both the edges *and the vertices* (except for s and t) have capacities. The flow into and out of a vertex cannot exceed the capacity of the vertex.

3. *(Using reductions to prove $\mathcal{NP}$-completeness)*

   (a) A *clique* in an undirected graph $G = (V, E)$ is a subset S of vertices such that *all* possible edges between the vertices in S appear in E. Computing the maximum clique in a network (or the number of cliques of at least a certain size) is useful in analyzing social networks, where cliques corresponds to groups of people who all know each other.

   State the decision version of the above maximization problem and show that it is $\mathcal{NP}$-complete. *Hint: reduction from* `Independent Set`*.*

   (b) We say that G is a *subgraph* of H if, by deleting certain vertices and edges of H we obtain a graph that is, up to renaming of the vertices, identical to G.

   The following problem has applications, e.g., in pattern discovery in databases and in analyzing the structure of social networks.

   `Subgraph Isomorphism:` Given two undirected graphs G and H, determine whether G is a subgraph of H and if so, return the corresponding mapping of vertices in G to vertices in H.

   Show that `Subgraph Isomorphism` is $\mathcal{NP}$-complete.

   (c) Similarly, consider the following problem.

   `Dense Subgraph:` Given a graph G and two integers a and b, find a set of a vertices of G such that there are at least b edges between them.

   Show that `Dense Subgraph` is $\mathcal{NP}$-complete.

4. Suppose you are given n cities and a set of non-negative distances d_{ij} between pairs of cities.

   (a) Give an $O(n^2 2^n)$ dynamic programming algorithm to solve this instance of **TSP**; that is, compute the cost of the optimal tour and output the actual optimal tour.

   (b) What are the space requirements of your algorithm?

   Hint: Let $V = \{1, \ldots, n\}$ *be the set of cities. Consider progressively larger subsets of cities; for every subset* S *of cities including city* 1 *and at least one other city, compute the shortest path that starts at city* 1*, visits all cities in* S *and ends up in city* j*, for every* $j \in S$*.*