CSC373 Fall'19
Tutorial 5 with Solutions
Nov 11, 2019

Q1 LP and IP

For each problem below, describe how to represent the problem as a linear or integer program, including a brief justification of your inequalities and an explanation of how solutions to your program correspond to solutions to the original problem.

1. Simple Scheduling with Prerequisites (SSP):

You are given a set of jobs, some of which need to be finished before others (prerequisites). You need to assign start times to jobs to complete all jobs in the least amount of time. (Note: If there are circular prerequisites, then there is no feasible solution, and your program should indicate so.)

Formally, you are given n jobs with a list of durations $d_1, d_2, \ldots, d_n$, and a boolean $p_{i,j}$ for every pair (i, j) of jobs such that if it is true, job i must finish before job j can begin. You need to find start times $s_1, s_2, \ldots, s_n$ for the jobs (no job can start earlier than time 0) that minimize the total time to complete all jobs, while ensuring each job i finishes no later than the start time of all jobs j for which i is a prerequisite. Write a linear program for this.

Solution:

For each job i, we have a variable s_i, which is the start time of job i. We also have a variable T, which is an upper bound on the total time to completion. The LP is as follows.

Minimize T
Subject to:
$T \geq s_i + d_i$, for $i \in \{1, 2, \ldots, n\}$;
$s_j \geq s_i + d_i$, for $i, j \in \{1, 2, \ldots, n\}$ such that $i \neq j$ and $p_{i,j}$ is true;
$s_i \geq 0$, for $i \in \{1, 2, \ldots, n\}$;
$T \geq 0$.

The constraints ensure that each job starts only after each of its prerequisites has completed, and that T is at least the total completion time. If we minimize T subject to this, it will be set equal to the total completion time, and the values of s_i will be chosen to minimize this time.

2. Set Cover (SC):

Given a set of elements $E = \{x_1, x_2, \ldots, x_n\}$ and a collection of subsets of E, $S = \{H_1, H_2, \ldots, H_m\}$, we want to find the smallest subset C of E such that for each set $H_j \in S$, $C \cap H_j \neq \phi$. Write an integer program for this.

Solution:

We have one binary variable $v_i \in \{0, 1\}$ to indicate whether element $i \in E$ is chosen. The IP is as follows.

Minimize $\sum_{i=1}^{n} v_i$
Subject to:
$\sum_{i:x_i \in H_j} v_i \geq 1$, for each $j \in \{1, \ldots, m\}$;
$v_i \in \{0, 1\}$, for each $i \in \{1, \ldots, n\}$.

Note that the constraint ensures that for each set H_j, at least one of the v_i's corresponding to its elements x_i's will be set to 1. Hence, the resulting C ensures $C \cap H_j \neq \emptyset$ for each j. Subject to this, we are minimizing $|C| = \sum_i v_i$, which gives us the desired solution.

Q2 Complexity

Are the following decision problems in P/NP/coNP?

1. TRIANGLE
Input: An undirected graph $G = (V, E)$.
Question: Does G contain a "triangle" (i.e. a subset of three vertices such that there is an edge between any two of them)?

2. CLIQUE
Input: An undirected graph $G = (V, E)$ and a positive integer k.
Question: Does G contain a k-clique (i.e. a subset of k vertices such that there is an edge between any two of them)?

3. NON-ZERO
Input: A set of integers S.
Question: Does every non-empty subset of S have non-zero sum?

4. HAMILTONIAN-PATH (HP)
Input: An undirected graph $G = (V, E)$.
Question: Does G contain a simple path that includes every vertex?

Solution:

- P: One can brute-force and check all triplets of vertices for triangles in $O(n^3)$ time.
- NP: Given a k-clique as advice, a TM can verify in polynomial time whether all $\binom{k}{2}$ pairs of vertices have an edge.
- coNP: If the answer is NO (i.e. there is a non-empty subset of S with zero sum), then given such a subset as advice, a TM can verify in polynomial time that its sum is indeed zero and thus the answer to the problem is NO.
- NP: Given a Hamiltonian path as advice, a TM can verify that it includes every vertex exactly once and there is an edge between every pair of adjacent vertices (i.e. it is indeed a path).

Q3 Reductions

Consider the Hamiltonian Cycle (HC) problem, which is similar to the HP problem described above.

HAMILTONIAN-CYCLE (HC)
Input: An undirected graph $G = (V, E)$.
Question: Does G contain a simple cycle that includes every vertex?

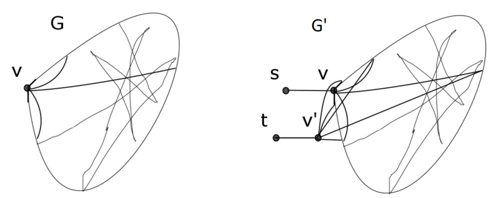
1. The textbook CLRS shows that HC is is NP-complete (Subsection 34.5.3). Give a reduction from HC to HP (i.e. HC $\leq_p$ HP) to prove HP is also NP-complete.

2. Suppose instead that we knew HP is NP-complete, and wanted to use it to show that HC is NP-complete. Give a reduction from HP to HC (i.e. HP $\leq_p$ HC).

Solution:

1. Given $G = (V, E)$, create $G' = (V', E')$ as follows. Start with $G' = G$. Choose a vertex $v \in V$, and add a copy of it (say v') to G': that is, for every $(v, u) \in E$, we also add $(v', u) \in E'$. Then, add a new start vertex s and a new end vertex t, and add edges (s, v) and (v', t) to G'.

Note that G' has a HP iff this HP has s and t as the two endpoints iff the second and the second to last vertices in the HP are v and v' iff G has a HC.



2. Given $G = (V, E)$, create $G' = (V', E')$ as follows. Start with $G' = G$. Add a new vertex u, and add edges (u, v) for every $v \in V$.

Note that G' has a HC $(s, v_1, \ldots, v_n)$ iff G has a HP $(v_1, \ldots, v_n)$.