Assignment 3
Due May 21, 2018

Question 1: Enumeration

Let $G = (L, R, E)$ be a bipartite graph; that is, $L$ and $R$ are disjoint finite sets of left nodes and right nodes, respectively, and $E$ is a subset of $L \times R$. A matching is a subset $M$ of $E$ such that each node appears in at most one edge in $M$. A matching $M$ is maximal if it is not strictly contained in any other matching, or in other words, there is no matching $M'$ such that $M \subsetneq M'$.

Devise an algorithm for enumerating with polynomial delay the maximal matchings of a given bipartite graph. Prove the correctness (soundness, completeness, lack of repetitions) and efficiency (polynomial delay) of your algorithm.

Hint: Adapt the algorithm taught in class for enumerating the maximal cliques.

Question 2: Hypergraph Acyclicity

In this question, you will devise an algorithm for constructing a join tree, if one exists, of a given hypergraph.

Question 2.1

Prove the following about a hypergraph $H$ with a nonempty set of hyperedges.

1. If $H$ has a join tree, then $H$ has an ear.
2. If $H$ has a join tree and we remove an ear from $H$, then the resulting hypergraph also has a join tree.

Question 2.2

Use the insights from the previous question to devise an efficient algorithm for constructing a join tree, if one exists, of a given hypergraph.

Question 3: Yannakakis Algorithm

Consider a CQ expression $\alpha = \pi_A(R_1 \Join \cdots \Join R_k)$. A fake join tree of $\alpha$ is similar to a join tree of $\alpha$, except that we do not require that every node occurs in a connected subtree. More formally, a fake join tree of $\alpha$ is a tree $T$ such that the node set of $T$ is $\{R_1, \ldots, R_k\}$.

Suppose that the Yannakakis algorithm taught in class used a fake join tree $T$ instead of an ordinary one for computing $\alpha(D)$ on an input database $D$. In the following questions, if the answer is “yes” then no explanation is needed. If the answer is “no” then provide an example of an input $D$ and execution of Yannakakis to prove the answer.
Would the new algorithm return all tuples in \( \alpha(D) \)?

Would the new algorithm return only tuples in \( \alpha(D) \)?

**Question 4: Incomplete Databases**

Let \( \mathcal{I} \) be a v-instance, and let \( Q(x_1, \ldots, x_k) \) be a CQ. Prove that the following algorithm computes precisely the certain answers of \( Q \) on \( \mathcal{I} \).

1. Evaluate \( Q \) over \( \mathcal{I} \), assuming that each variable is a unique (fresh) constant.

2. Remove from the result of the first step every tuple that contains one or more variables.

Good luck!