Assignment 1
Due April 21st, 2016

Part 1: Integrity Constraints

Explain the relationship between the expressive power of the following types of constraints:

1. Functional dependencies
2. Denial constraints
3. Inclusion dependencies
4. Tuple-generating dependencies
5. Equality-generating dependencies

From your answer one should derive, for every two types \( T_1 \) and \( T_2 \), one of the following:

- A translation procedure of every constraint of \( T_1 \) into \( T_2 \).
- An example of a constraint of type \( T_1 \) that cannot be expressed by means of \( T_2 \)

You do not need to prove the correctness of the translation or the example.

Part 2: Complexity

Question 1 (CQ Complexity)

In the lecture we saw a reduction from the maximum clique problem to that of evaluating a boolean CQ (under combined complexity). In that reduction, the constructed CQ had a single relation repeated many times. Is boolean CQ evaluation still NP-hard if we assume that each relation symbol occurs only once (i.e., there are no “self joins”)?

Question 2 (Polynomial Delay A)

Prove that the algorithm of Figure 1 enumerates the maximal cliques of an input graph \( g \) with polynomial delay.

Note that you need to prove all of the following:

1. The algorithm enumerates only maximal cliques.
2. The algorithm prints every maximal clique. (This is the hardest part of the proof.)
3. The algorithm does not print the same clique more than once.
4. The time between every two prints is polynomial in the size of the input \( g \).
**EnumMaxCliques**

\[ O := \emptyset \]

Generate an arbitrary maximal clique \( C \)

\[ Q := \{ C \} \]

**while** \( Q \neq \emptyset \) **do**

\[ C := Q . \text{pop()} \]

Print \( C \)

\[ O . \text{push}(C) \]

**for all** nodes \( v \) of \( g \) **do**

\[ D := \{ v \} \cup \{ u \in C | u \text{ is a neighbor of } v \text{ in } g \} \]

Arbitrarily extend \( D \) into a maximal clique \( C' \)

**if** \( C' \notin Q \cup O \) **then**

\[ Q . \text{push}(C') \]

---

**Figure 1:** An algorithm for enumerating the maximal cliques

---

**Question 3 (Polynomial Delay B)**

A *bipartite graph* is a triple \((L, R, E)\) where \( L \) and \( R \) are disjoint sets and \( E \) is a subset of \( L \times R \).

A *perfect matching* in a bipartite graph \((L, R, E)\) is a subset \( M \) of \( E \) such that every member of \( L \cup R \) occurs in precisely one pair in \( M \). Show an algorithm or enumerating all the perfect matchings of a bipartite graph with polynomial delay and polynomial space. You can assume the existence of a polynomial-time procedure for generating a single perfect matching (e.g., the Hopcroft-Karp algorithm).

---

**Question 4 (Parameterized Complexity)**

Recall that the *Vertex Cover* problem is the following one. Given a graph \( g \) and a number \( k \), determine whether there is a set \( U \) of \( k \) or fewer nodes, such that every edge of \( g \) is incident to at least one node of \( U \). This problem is known to be NP-complete. Prove that it is fixed-parameter tractable for the parameter \( k \).

**Hint:** try to construct a cover of size at most \( k \). If you pick an arbitrary edge \( e \), then at least one of the nodes of \( e \) should occur in such a cover. If you knew which node it was, then you could reduce the problem into that of finding a cover of size at most \( k - 1 \).

Good luck!