Problem 1: 10 Points

Consider the following tuple-independent probabilistic database:

<table>
<thead>
<tr>
<th>Student</th>
<th>Course</th>
<th>Pr(·)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>Artificial Intelligence (AI)</td>
<td>0.7</td>
</tr>
<tr>
<td>Alice</td>
<td>Programming Languages (PL)</td>
<td>0.4</td>
</tr>
<tr>
<td>Bob</td>
<td>Artificial Intelligence (AI)</td>
<td>0.3</td>
</tr>
</tbody>
</table>

(a) The enrollment table, $E$.

<table>
<thead>
<tr>
<th>Student</th>
<th>Pr(·)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>0.7</td>
</tr>
<tr>
<td>Bob</td>
<td>0.4</td>
</tr>
<tr>
<td>Charlie</td>
<td>0.9</td>
</tr>
</tbody>
</table>

(b) The honors table, $H$.

Figure 1: A tuple-independent probabilistic database with two tables.

Part A:

A tuple-independent probabilistic database describes a probability distribution on a collection of classical deterministic databases (the possible worlds). For just the table $T_H$, give (1) the set of all possible world databases of this table; (2) the probability of each such deterministic database.

Solution:

<table>
<thead>
<tr>
<th>Tuples in the World</th>
<th>Calculation</th>
<th>Pr(w)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice, Bob, Charlie</td>
<td>$0.7 \times 0.4 \times 0.9$</td>
<td>0.252</td>
</tr>
<tr>
<td>Alice, Bob</td>
<td>$0.7 \times 0.4 \times 0.1$</td>
<td>0.028</td>
</tr>
<tr>
<td>Alice, Charlie</td>
<td>$0.7 \times 0.6 \times 0.9$</td>
<td>0.378</td>
</tr>
<tr>
<td>Alice</td>
<td>$0.7 \times 0.6 \times 0.1$</td>
<td>0.042</td>
</tr>
<tr>
<td>Bob, Charlie</td>
<td>$0.3 \times 0.4 \times 0.9$</td>
<td>0.108</td>
</tr>
<tr>
<td>Bob</td>
<td>$0.3 \times 0.4 \times 0.1$</td>
<td>0.012</td>
</tr>
<tr>
<td>Charlie</td>
<td>$0.3 \times 0.6 \times 0.9$</td>
<td>0.162</td>
</tr>
<tr>
<td>$\emptyset$</td>
<td>$0.3 \times 0.6 \times 0.1$</td>
<td>0.018</td>
</tr>
</tbody>
</table>
Part B:

1. \( \Pr(\exists x. H(x)) \)

**Solution:**

\[
\Pr(\exists x. H(x)) = 1 - \Pr(\forall x. \neg H(x)) = 1 - \prod_x \Pr(\neg H(x)) = 1 - \prod_x 1 - \Pr(H(x)) = 1 - (1 - 0.7)(1 - 0.4)(1 - 0.9) = 1 - 0.018 = 0.982
\]

2. \( \Pr(\exists x. H(x) \land E(x, \text{Artificial Intelligence(AI)})) \)

**Solution:**

\[
\Pr(\exists x. H(x) \land E(x, \text{AI})) = 1 - \Pr(\forall x. \neg H(x) \lor \neg E(x, \text{AI})) = 1 - \prod_x \Pr(\neg H(x) \lor \neg E(x, \text{AI})) = 1 - \prod_x \Pr(\neg H(x)) + \Pr(\neg E(x, \text{AI})) - \Pr(\neg H(x) \land \neg E(x, \text{AI})) = 1 - (0.3 + 0.3 - 0.3 \cdot 0.3)(0.6 + 0.7 - 0.6 \cdot 0.7)(0.1 + 1 - 1) = 0.5512
\]

This can also be done more simply without using inclusion exclusion, either way you will arrive at the same final answer.

3. \( \Pr(\exists x \exists y. H(x) \land E(x, y)) \)
Solution:

\[ \text{Pr}(Q) = 1 - \text{Pr}(\forall x \neg H(x) \lor \exists y E(x, y)) \]
\[ = 1 - \prod_{x_i} \text{Pr}\left( \neg H(x_i) \lor \forall y \neg E(x, y) \right) \]
\[ = 1 - \prod_{x_i} 1 - \text{Pr}\left( H(x_i) \land \exists y E(x, y) \right) \]
\[ = 1 - \prod_{x_i} 1 - \text{Pr}\left( H(x_i) \right) \text{Pr}\left( \exists y E(x_i, y) \right) \]
\[ = 1 - \prod_{x_i} 1 - \text{Pr}\left( H(x_i) \right) \left( 1 - \text{Pr}(\forall y \neg E(x_i, y)) \right) \]
\[ = 1 - \prod_{x_i} 1 - \text{Pr}\left( H(x_i) \right) \left( 1 - \prod_{y_j} (1 - \text{Pr}(E(x_i, y_j))) \right) \]
\[ = 1 - \left( 1 - 0.7(1 - 0.7)(1 - 0.4) \right) \left( 1 - 0.4(1 - 0.3) \right) \left( 1 - 0.9(1 - 1) \right) \]
\[ = 0.62512 \]
Problem 2: 20 Points

1. \( \Pr(\exists x. R(x) \land T(x)) \)

Solution:
\[
\Pr(\exists x. R(x) \land T(x)) = 1 - \Pr \left( \forall x. \neg(R(x) \land T(x)) \right)
\]
\[
= 1 - \prod_{x_i} \Pr \left( \neg(R(x_i) \land T(x_i)) \right)
\]
\[
= 1 - \prod_{x_i} 1 - \Pr(R(x_i)) \Pr(T(x_i))
\]

2. \( \Pr(\exists x. \exists y. S(x, y) \land R(x)) \)

Solution:
\[
\Pr \left( \exists x. \exists y. S(x, y) \land R(x) \right) = 1 - \Pr \left( \forall x. \forall y. \neg(S(x, y) \land R(x)) \right)
\]
\[
= 1 - \prod_{x_i} \Pr \left( \neg(S(x_i, y_i) \land R(x_i)) \right)
\]
\[
= 1 - \prod_{x_i} 1 - \Pr(R(x_i)) \cdot \Pr \left( \exists y. S(x_i, y) \right)
\]
\[
= 1 - \prod_{x_i} 1 - \Pr(R(x_i)) \cdot \left( 1 - \prod_{y_j} \Pr(S(x_i, y_j)) \right)
\]

3. \( \Pr(\exists x. \exists y. R(x) \land S(x, y) \land T(y)) \lor (\exists x. U(x)) \)

Solution:
\[
\Pr(Q) = 1 - \left( 1 - \Pr \left( \exists x. \exists y. R(x) \land S(x, y) \land T(y) \right) \right) \left( 1 - \Pr \left( \exists x. U(x) \right) \right)
\]
\[
= 1 - \left( 1 - \Pr(Q_1) \right) \left( 1 - \Pr(Q_2) \right)
\]
\[
= \cdots
\]

But in \( Q_1 : \exists x. \exists y. R(x) \land S(x, y) \land T(y) \) there is no separator so the algorithm fails and hence the original query \( Q \) is \#P.
4. \( \Pr(\exists x_1. \exists x_2. \exists y_1. \exists y_2. R(x_1) \land S(x_1, y_1) \land T(x_2) \land S(x_2, y_2)) \)

**Solution:**

\[
\Pr(Q) = \Pr\left( (\exists x_1 \exists y_1 R(x_1) \land S(x_1, y_1)) \land (\exists y_2 \exists x_2 T(x_2) \land S(x_2, y_2)) \right)
\]

\[
= \Pr\left( (\exists x \exists y R(x) \land S(x, y)) \land (\exists x \exists y T(x) \land S(x, y)) \right)
\]

\[
= \Pr(Q_1) + \Pr(Q_2) - \Pr(Q_1 \lor Q_2)
\]

We can easily do \( \Pr(Q_1) \) and \( \Pr(Q_2) \), similar to part 2, so if we compute \( \Pr(Q_1 \lor Q_2) \) we are done.

\[
\Pr(Q_1 \lor Q_2) = \Pr\left( \exists x_1 \exists y_1 R(x_1) \land S(x_1, y_1) \lor \exists y_2 \exists x_2 T(x_2) \land S(x_2, y_2) \right)
\]

\[
= \Pr\left( \exists x \exists y R(x) \land S(x, y) \lor \exists x \exists y T(x) \land S(x, y) \right)
\]

\[
= \Pr\left( \exists x (T(x) \lor R(x)) \land \exists y S(x, y) \right)
\]

\[
= \prod_{x_i} \Pr\left( (T(x_i) \lor R(x_i)) \land \exists y S(x_i, y) \right)
\]

\[
= \prod_{x_i} \Pr\left( (T(x_i) \lor R(x_i)) \right) \cdot \Pr\left( \exists y S(x_i, y) \right)
\]

\[
= \prod_{x_i} \left( 1 - (1 - \Pr(T(x_i)))(1 - \Pr(R(x_i))) \right) \cdot \left( 1 - \Pr(\forall y \neg S(x_i, y)) \right)
\]

\[
= \prod_{x_i} \left( 1 - (1 - \Pr(T(x_i)))(1 - \Pr(R(x_i))) \right) \cdot \left( 1 - \prod_{y_j} 1 - \Pr(S(x_i, y_j)) \right)
\]

Combining result from \( \Pr(Q_1) \), \( \Pr(Q_2) \), and \( \Pr(Q_1 \lor Q_2) \) we can get the final answer.

5. (רחמנה \( \P \), Bonus Spooky Halloween Question, worth 0 points) \( \Pr((\exists x_1. \exists y_1. R(x_1) \land S(x_1, y_1)) \lor (\exists x_2. \exists y_2. S(x_2, y_2) \land T(y_2)) \lor (\exists x_3. \exists y_3. R(x_3) \land T(y_3))) \)

*(Hint: you may need to symbolically cancel some queries Good Luck)*

\[
\int_{\P} \text{不失魂魄} \, d\text{≠≠≠} = \text{≠≠≠}
\]
Problem 3: 10 Points

Recall the following query from class:

\[ H_0 = \exists x. \exists y. S(x) \land F(x, y) \land R(y) \]  

(1)

In class, we showed that evaluating \( H_0 \) for an arbitrary database is \( \#P \)-hard in the size of the database. We proved this by reduction to the positive partitioned 2-CNF (\( \text{#PP2CNF} \)) counting problem, for which we now give a formal definition.

**Definition 1.** A 2-CNF is a CNF where each clause has exactly 2 literals. A positive partitioned 2-CNF is a 2-CNF where variables are partitioned into 2 sets \( X, Y \), \( X \cap Y = \emptyset \), and every clause is of the form \((x \lor y)\) with \( x \in X \) and \( y \in Y \). Finally, \( \text{#PP2CNF} \) is the problem of counting how many models a PP2CNF formula has.

For each of the following PP2CNF formulae, give three tables \( S(x), F(x, y), \) and \( R(y) \) such that evaluating \( H_0 \) on these tables can be used to compute the model count of the formula:

1. \( f = x_1 \lor y_1 \).

   **Hint:** What is the model count for this formula? Since you can compute it by hand, use it to test your answer.

   **Solution:** There are 3 models.

2. \( f = (x_1 \lor y_1) \land (x_2 \lor y_2) \land (x_3 \lor y_3) \land (x_1 \lor y_3) \).

   **Hint:** Represent the \( x_i \) variables as tuples in table \( S(x) \), the \( y_i \) variables as tuples in table \( R(y) \), and each clause as a tuple in \( F(x, y) \). How do you relate “a variable is true” to the presence of a tuple in \( S \) or \( R \)? How do you relate “a clause is true” to the presence of a tuple in \( F \)? How can you choose the weights of each tuple so that the query can be used to compute (but does not necessarily equal) the model count?

   **Solution:** First we define the tables in our PDB, which we label \( D \). Let \( f \) be a PP2CNF with variables \( X = \{x_i\} \) and \( Y = \{y_i\} \) and clauses \( C = \{(x_i, y_j)\} \). Let \( F(x, y) \) contain a tuple \((x_i, y_j)\) if and only if the clause \((x_i, y_j)\) is in \( C \). Let \( S(x) \) contain a tuple \( x_i \) if and only if \( x_i \) is a variable in \( X \), and similarly for \( R(x) \) and \( Y \). Let the probability of each tuple in \( S(y) \) and \( R(y) \) be 0.5, and the probability of each tuple in \( F(x, y) \) be 1.

   Then, we convert \( f \) into a PP2DNF by negating it:

   \[ \neg f = \neg \left( \bigwedge_{(x_i, y_j) \in C} x_i \lor y_j \right) = \bigvee_{(x_i, y_j) \in C} \neg x_i \land \neg y_j \]

   We know that \( \#(f) = 2^n - \#(\neg f) \), where \( n \) is the number of variables in \( f \). Then, we give a bijection between models of \( \neg f \) and deterministic databases from \( D \) which satisfy \( H_0 \).

   - Let \( m \) be a model of \( \neg f \). Then, we translate this model into a deterministic database which satisfies \( H_0 \). If \( x_i \) is false in \( m \), then we include the tuple \( x_i \) in \( S(x) \), and similarly for \( y_j \) and \( R(y) \). We include a tuple \( \{(x_i, y_j)\} \) in \( F(x, y) \) iff either \( x_i \) both \( y_j \) are false in \( m \). Clearly, this deterministic database satisfies \( H_0 \), and the weight of this deterministic database is equal to \( 0.5^n \times 1 \), where \( n \) is the number of variables.

Problem 3 continued on next page...
Problem 3 (continued)

- Let $D$ be a deterministic database which satisfies $H_0$. Then, convert $D$ to model by setting $x_i = T$ iff $x_i \in R(x)$, and similarly for $y_j$. This is a model for $\neg f$, since if it did not satisfy each term in $\neg f$ then it would not satisfy $H_0$.

Thus there is a one-to-one mapping between deterministic databases and models of $\neg f$. Now we can use the evaluation of $H_0$ to compute the number of models of $\neg f$:

$$\Pr(\exists x. \exists y. R(x) \land F(x, y) \land R(y)) = (0.5^n) \times \#(\neg f).$$

Now we can relate the outcome of the PDB query to the number of models of a PP2CNF $f$. Let $\#(f)$ be the number of models of a formula $f$. If we assume that each world has an equal probability, then we can define $\Pr(f)$ as the probability that a uniformly randomly drawn world satisfies $f$ (Recall the sampling programming question from Homework #2). Thus, we can compute $\#(f)$:

$$\Pr(f) = \frac{\#(f)}{2^n} = 1 - \frac{\#(\neg f)}{2^n} = 1 - \Pr(H_0)$$

So the model count becomes:

$$\#(f) = 2^n \cdot (1 - \Pr(H_0))$$

And in the next page we see how to make the database so that above equations work. For example, $F(x_i, y_j)$ is only 1, when we have term $x_i \lor y_j$ in $f$. Also $S(x_i)$ is defined to be $\neg x_i$ but the probability will still be 0.5, similarly for $R(y_j)$.

<table>
<thead>
<tr>
<th>X</th>
<th>Pr(·)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>0.5</td>
</tr>
</tbody>
</table>

(a) The table $S$

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Pr(·)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$y_1$</td>
<td>1</td>
</tr>
</tbody>
</table>

(b) The table $F$. Everything else has $p = 0$

<table>
<thead>
<tr>
<th>Y</th>
<th>Pr(·)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_1$</td>
<td>0.5</td>
</tr>
</tbody>
</table>

(c) The table $R$.

Figure 2: Problem 3 Part 1

<table>
<thead>
<tr>
<th>X</th>
<th>Pr(·)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>0.5</td>
</tr>
<tr>
<td>$x_2$</td>
<td>0.5</td>
</tr>
<tr>
<td>$x_3$</td>
<td>0.5</td>
</tr>
</tbody>
</table>

(a) The table $S$

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Pr(·)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$y_1$</td>
<td>1</td>
</tr>
<tr>
<td>$x_2$</td>
<td>$y_2$</td>
<td>1</td>
</tr>
<tr>
<td>$x_3$</td>
<td>$y_3$</td>
<td>1</td>
</tr>
</tbody>
</table>

(b) The table $F$. Everything else has $p = 0$

<table>
<thead>
<tr>
<th>Y</th>
<th>Pr(·)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_1$</td>
<td>0.5</td>
</tr>
<tr>
<td>$y_2$</td>
<td>0.5</td>
</tr>
<tr>
<td>$y_3$</td>
<td>0.5</td>
</tr>
</tbody>
</table>

(c) The table $R$.

Figure 3: Problem 3 Part 2

For part 1, there is 4 possible worlds. Out of those 4 only in one of them the query $H_0$ holds, so $P(H_0) = 1/4$, additionally $\#(f) = 3$. As we see our formula works in this case:

$$3 = \#(f) = 2^2(1 - P(H_0))$$
References