CSE 21
Solutions to Practice Exam for Midterm 2
Fall 2018

1. **Representing Problems as Graphs** I have \$10, and I plan to spend some or all of my money on three types of candy, which I will buy one piece at a time:

   - chocolate bars cost \$3,
   - almond rocca cost \$2, and
   - caramel chunks cost \$5.

   I want to know what combinations of candy I can afford; I might buy more than one of the same type.

   (a) Describe how you would model this situation using a directed graph, where paths in your graph should represent possible sequences of candy purchases. What are the vertices, and when are two vertices connected with an edge?

   Solution: Create eleven vertices to represent how much money I could have remaining, with possible values \$0, \$1, \$2, \$3, and so on up to \$10.

   Add edges from each vertex representing an amount a to $a-2$ (unless $a < 2$), $a-3$ (unless $a < 3$) and $a-5$ (unless $a < 5$). Label the edges rocca, chocolate, and caramel, respectively.

   The labels of a path are possible sequences of candy purchases.

   (b) How can this graph be used to determine which amounts of change I might have left over when I have had my fill of candy?

   Solution: Any node reachable from my starting amount is a possible amount of change. So I could use a graph search algorithm to determine this set of reachable nodes.

   (c) Give the adjacency list representation for this graph.

   Solution:
   0: [];
   1: [];
   2: [0];
   3: [0, 1];
   4: [1, 2];
   5: [0, 2, 3];
   6: [1, 3, 4];
   7: [2, 4, 5];
   8: [3, 5, 6];
   9: [4, 6, 7];
   10: [5, 7, 8].

   (d) Is your graph a DAG? Explain why or why not.

   Solution: Since the amount of money I have decreases with every purchase, there can be no cycles. The only edges in the graph connect one amount of money to a smaller amount. If there were a cycle from a to b to c and back to a, this would mean $a > b$, $b > c$, and $c > a$ which cannot happen because $a > b$ and $b > c$ means $a > c$. So, graph is a DAG.

2. Graphs, Trees, Tours, and Counting A complete bipartite graph is an undirected graph where the vertex set V can be partitioned into two sets V_1 and V_2 so that

(i) there is an edge between every vertex in V_1 and every vertex in V_2,

(ii) there are no edges between the vertices of V_1, and

(iii) there are no edges between the vertices of V_2.

Answer the following questions about complete bipartite graphs, and show how you came to your conclusions.

(a) How many vertices are there in a complete bipartite graph with $|V_1| = m$ and $|V_2| = n$?

Solution: Number of vertices: $m + n$ since the vertex sets are disjoint.

(b) How many edges are there in a complete bipartite graph with $|V_1| = m$ and $|V_2| = n$?

Solution: Number of edges: $m \cdot n$ because every vertex in V_1 will be connected to every vertex in V_2. So, the degree of every vertex in V_1 is n and there are m vertices in V_1. This accounts for all the edges.

(c) For which values of m and n is a complete bipartite graph an unrooted tree?

Solution: We need $m = 1$ **or** $n = 1$.

To prove this, suppose for a contradiction that we have a complete bipartite graph which is an unrooted tree, such that $m \geq 2$ **and** $n \geq 2$. Then we can find two vertices, say a and b, which are in V_1 and two vertices, say c and d, which are in V_2. By the definition of a complete bipartite graph (first bullet point), there is an edge from a to c, an edge from c to b, an edge from b to d, and an edge from d to a. Thus, there is a cycle from a back to itself which is a contradiction because unrooted trees have no cycles.

(d) For which values of m and n does a complete bipartite graph have an Eulerian tour that starts and ends at different vertices?

Solution: In the complete bipartite graph, every node in V_1 has degree n and every node in V_2 has degree m. A graph has an Eulerian tour that starts and ends at different vertices if and only if there are exactly two nodes of odd degree. Thus, at least one of n and m must be odd.

If both are odd, there must be exactly one node on both sides, so $n = m = 1$.

If one is even and the other odd, since every node on the even side has odd degree, the even side must have exactly two nodes. The odd side can have any odd number of nodes.

(e) For which values of m and n does a complete bipartite graph have a Hamiltonian tour?

Solution: Since any tour alternates between the two sides, at any point either both sides have been visited the same number of times, or one side has been visited one more time. Thus, if a Hamilton tour exists, either $n = m$ or they differ by exactly one. If either of these things are true, we can find a Hamiltonian tour in the following way: order the vertices in V_1 from 1 to m, and the vertices in V_2 from 1 to either $m - 1, m$ or $m + 1$. Starting on the large side, a tour that visits the vertices in order on the two sides will be a Hamiltonian tour. So a Hamiltonian tour exists iff n and m differ by at most 1.

(f) If $m = n$ how many Hamiltonian tours does a complete bipartite graph have?

Solution: If $n = m$, then in this case the Hamiltonian tours are circuits. We can start at any node, which has $2n$ choices. From there, we can visit any node on the other side, which has n choices. For the other steps, if we have gone $2i$ steps already, there will be $n - i$ vertices remaining on either side, and we can visit any two vertices on the two sides next. So the total number is $2n * n(n-1)(n-1)(n-2)(n-2)...1 * 1 = 2(n!)^2$.

However, since we can now put $2n$ vertices in an ordered cycle and there are $2n$ ways to rotate the cycle, we have double counted every objects by a factor of $2n$. Therefore, the actual number of Hamiltonian circuits (after adjusting for symmetry) is $2(n!)^2/(2n) = n!(n-1)!$

3. Counting In each of the following problems, a hand of five cards will be dealt from a standard deck of cards, with thirteen cards in each of four suits, and no jokers or wild cards. A hand is a *set* of five cards, so the order in which the cards are dealt does not matter. Say how many different hands of the following types are possible. Here, as well as on the exam, you can leave your answer as an unsimplified algebraic expression involving binomial coefficients, factorials, or exponents. You do not need to simplify.

(a) Straight: A hand where the numbers of the cards are five consecutive integers (with Jack = 11, Queen = 12, King = 13, and Ace counting as 1 or 14).

Solution: We could **start** the straight at any number between 1 and 10. This means there are 10 possible sets of five numbers which could make up the straight. For each of the five numbers in the straight, we must use exactly one of that number and there are four suits to pick from. Thus, the total number of straights is $10 * 4^5 = 10,240$.

(b) Three of a kind: A hand with three cards of one number, one card of a second number, and one card of a third number.

Solution: Of the thirteen kinds of numbers, we must pick one to have three cards and two of the remaining to have one card. That is $13 * \binom{12}{2}$ possibilities. Then, to choose the suits, we must pick three of the four cards with the first number and one of four cards for each of the other two numbers. That is $\binom{4}{3}\binom{4}{1}\binom{4}{1}$. So the total is $13 * \binom{12}{2}\binom{4}{3}\binom{4}{1}\binom{4}{1} = 54,912$ possible hands.

(c) Two pair: A hand with two cards of one number, two cards of a second number, and one card of a third number.

Solution: We must pick two numbers to have two of, and one of the remaining numbers to have one of, for $\binom{13}{2} * 11$ possible combinations of numbers. Then to pick suits for each of the pairs, we pick two of the four cards with that number, and to pick a suit for the remaining card, we pick one of the four cards with that number. This gives $\binom{4}{2}\binom{4}{2}\binom{4}{1} = 6 * 6 * 4$ possibile ways to choose the suits. So the total number of hands is $(\binom{13}{2} * 11) * 6 * 6 * 4 = 123,552$.

4. Encoding and Decoding Consider ternary strings containing the symbols 0, 1, and 2. Say that we look at the set of such strings of length n that never have the same symbol appearing twice in a row. For example, 01212010 is such a string of length $n = 8$.

(a) How many ternary strings of length n are there that never have the same symbol appearing twice in a row?

Solution: There are three options for the first character of the string: 0, 1, 2. All the remaining characters can be one of only two possibilities, as the preceding character cannot be reused. As such, there are $3 * 2^{n-1}$ ternary strings of length n, for $n > 0$.

(b) How many bits (0s and 1s) are required to represent a ternary string of length n that never has the same symbol appearing twice in a row?

Solution: As a general rule, we take the ceiling of the log base 2 of the number of objects we are representing. In this case ,this gives

$$\lceil \log_2(3 * 2^{n-1}) \rceil = \lceil * \log_2(3) + \log_2(2^{n-1}) \rceil = \lceil \log_2(3) + n - 1 \rceil = n + 1$$

so it takes $n + 1$ bits to represent a ternary string of length n that never has the same symbol appearing twice in a row.

(c) Describe how to encode a ternary string of length n that never has the same symbol appearing twice in a row, using the number of bits you gave in part (b). Illustrate your description with an example.

Solution: Use two bits to encode the first character in the string (with 00 denoting 0, 01 denoting 1, and 10 denoting 2), and then use one additional bit to encode every subsequent character. This bit specifies the next ternary character among the two that are possible, with 0 denoting the smaller of the two possible ternary characters in that position and 1 denoting the larger. This encoding uses $n + 1$ bits to encode a ternary string of length n, since it uses 2 bits for the first ternary character and 1 bit for each of the remaining $n - 1$ ternary characters. Therefore, it is optimal and uses the number of bits given in part (b).

As an example, the ternary string 01212010 would be encoded as 000111000.

(d) Describe how to decode a ternary string of length n that never has the same symbol appearing twice in a row, if it has been encoded using the method you described in part (c). Illustrate your description with an example.

Solution: To decode, we would read the first two bits to get the first ternary character (with 00 denoting 0, 01 denoting 1, and 10 denoting 2). Then, for each bit after that, we interpret a 0 to mean the smaller of the two characters that was not in the previous position, and a 1 to mean the larger of the two characters that was not in the previous position.

As an example, the binary string 1011001 would be decoded as 212012.