CECS 419-519, Writing Assignment 7, Due 8:00 am, March 29th, 2024, Dr. Ebert

Directions

Make sure name is on all pages. Order pages (front and back) so that solutions are presented in their original numerical order. **Please no staples or folding of corners (your papers won't get lost). A paper clip is OK** Show all necessary work and substantiate all claims. Avoid plagiarism.

Problems

1. Consider the following method for constructing a binary encoding of a probability distribution $p_1 \geq p_2 \geq \cdots \geq p_n$. Define q_i by $q_1 = 0$ and, for $i \geq 2$,

$$q_i = p_1 + \cdots + p_{i-1}.$$

Let $m_i = \lceil \log(\frac{1}{p_i}) \rceil$. Finally, let codeword c_i be the binary expansion of q_i truncated at the first m_i bits to the right of the binary point. For example, if $q_i = 37/64$ and $p_i = 1/20$, then $m_i = \lceil \log(20) \rceil = 5$ and c_i is the first 5 bits of q_i, that is $c_i = 10010$, since $q_i = 100101$.

(a) Apply the algorithm to the probability distribution $P = \{0.3, 0.3, 0.15, 0.10, 0.10, 0.05\}$. (10 pts)

Solution. The following table gives the desired code.

i	q_i	p_i	m_i	c_i
1	0	0.3	2	00
2	0.3	0.3	2	01
3	0.6	0.15	3	100
4	0.75	0.1	4	1100
5	0.85	0.1	4	1101
6	0.95	0.05	5	11110

(b) Prove that the code obtained is a prefix code. (10 pts)

Solution. Let $i \in \{1, \ldots, n\}$ be an arbitraray codeword index and c_i denote the ith codeword. It suffices to show that c_i is not a prefix of any codeword c_j for which $j > i$. This is obvious if $i = n$, so suppose $i < n$. Now let c_j be the jth constructed codeword, where $j > i$. Let $r = |c_i|$ denote the length of c_i.

Claim: for every codeword c_j, $j > i$, the first r-bits of c_j (when viewed as a binary number) give a value that exceeds c_i (when also viewed as a binary number).

Proof of Claim. If $|c_j| = r = |c_i|$, then the claim is certainly true, since

$$q_j \geq q_i + p_{j-1} > q_i,$$

and, since $\lceil \log \frac{1}{p_{j-1}} \rceil = r$ which implies $p_{j-1} \geq \frac{1}{2^r}$, adding p_{j-1} to q_i makes the r bits of c_j greater than the r bits of c_i.

Now suppose $|c_j| = s > r = |c_i|$. It suffices to assume that c_j is the first codeword whose length exceeds r since, if we can show that its first r bits has a value that exceeds that of c_i, then the first r bits of all subsequent codewords will also have a value that exceeds that of c_i (once the first r bits reaches a value, the value of the first r bits of subsequent codewords cannot decrease since the q values are increasing and the codeword lengths are nondecreasing). So suppose $|c_{j-1}| = r < s = |c_j|$. Then $\lceil \log \frac{1}{p_{j-1}} \rceil = r$ which means that $p_{j-1} \geq \frac{1}{2^r}$. But $q_j = q_{j-1} + p_{j-1}$ and so q_j has a value in the first r bits that exceeds that of the first r bits of c_{j-1} and hence also that of the first r bits of c_i.

To finish the proof, we see that c_i cannot be a prefix of any c_j, $j > i$, since it would have to agree with c_j in the first r bits, but cannot since c_j exceeds c_i when both are viewed as r-bit numbers (by taking the first r bits of c_j). □

(c) Prove that the code obtained has an average codeword length A that satisfies

$$H_2(p_1, \ldots, p_n) \leq A \leq H_2(p_1, \ldots, p_n) + 1.$$

(10 pts)

Solution. The lower bound for A is due to Shannon's Noisless Coding theorem. For the upper bound, We have that the average length equals

$$A = \sum_{i=1}^{n} p_i \lceil \log \frac{1}{p_i} \rceil \leq \sum_{i=1}^{n} p_i (\log \frac{1}{p_i} + \varepsilon_i) \leq$$

$$\sum_{i=1}^{n} p_i (\log \frac{1}{p_i} + 1) = \sum_{i=1}^{n} (p_i \log \frac{1}{p_i} + p_i) = \sum_{i=1}^{n} p_i \log \frac{1}{p_i} + \sum_{i=1}^{n} p_i = H(p_1, \ldots, p_n) + 1. \quad \square$$

2. Recall the `Max Cut` decison problem from Writing Assignment 6. A special case of this problem is the `Half Max Cut` decision problem, where an instance is a graph G, and the problem is to decide if there is a way to color the vertices of G using the colors red and blue and results in there being at least $\lfloor |V|/2 \rfloor$ edges $e = (u, v)$ for which u and v are assigned different colors.

   (a) Provide a complete description of a polynomial-time mapping reduction from `Max Cut` to `Half Max Cut`. Provide a convincing argument for why the reduction works. (20 pts)

   Solution. We have the following cases.

   Case 1: $k = \lfloor |V|/2 \rfloor$. Then $f(G, k) = G$.

   Case 2: $k < \lfloor |V|/2 \rfloor$. Then $f(G, k) = G'$, where G' is G with J disjoint triangles, where

   $$k + 2J = (|V| + 3J)/2.$$

   The idea is that, for every triangle added to G, it allows for a coloring of the vertices of G' in such a way that 2 cut edges can be produced for each triangle (i.e. every 3 additional vertices). This has the effect of boosting the vertex-to-cut-edge ratio towards 1/2. Moreover, solving for J, we get

   $$J = |V| - 2k$$

   triangles that need to be added.

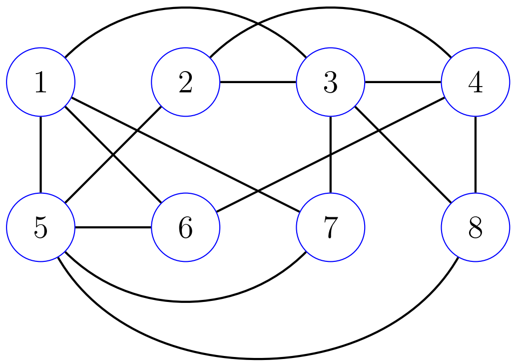
Case 3: $k > \lfloor |V|/2 \rfloor$. In this case, we may have too many cut edges (more than half the number of vertices). So $f(G,k) = G'$, where G' is G with J additional isolated vertices, where

$$k = (|V| + J)/2,$$

which implies

$$J = 2k - |V|.$$

(b) Apply your reduction to the following instance (G, k) of `Max Cut` where G is the graph shown below and $k = 10$. (5 pts)



Solution. Since $10 > |V|/2 = 4$, we have $f(G,k) = G'$, where G' is G with $J = 2k - |V| = 12$ additional isolated vertices. Thus, G will have at least 10 cut edges iff G' has at least 10 cut edges which is one half of its number of vertices $8 + 12 = 20$.

3. Given `3SAT` instance

$$\mathcal{C} = \{(x_1, x_2, \overline{x}_5), (x_1, \overline{x}_3, \overline{x}_4), (x_1, x_4, x_5), (\overline{x}_1, x_2, x_4),$$
$$(\overline{x}_1, \overline{x}_2, \overline{x}_5), (\overline{x}_1, \overline{x}_2, x_5), (x_2, x_3, x_5), (x_2, \overline{x}_3, x_5),$$
$$(x_2, \overline{x}_4, \overline{x}_5), (\overline{x}_2, x_3, x_4), (\overline{x}_2, x_3, \overline{x}_5), (\overline{x}_2, \overline{x}_4, x_5),$$
$$(\overline{x}_2, x_3, \overline{x}_4), (x_3, \overline{x}_4, x_5), (\overline{x}_3, x_4, x_5), (\overline{x}_3, \overline{x}_4, \overline{x}_5)\},$$

answer the following questions.

(a) For the mapping reduction f from `3SAT` to `Subset Sum` presented in lecture, if $f(\mathcal{C}) = (S, t)$, then what is $|S|$ and what is the largest number in S? Explain and show work. (10 pts)

Solution. Theere are a total of $2n + 2m = 10 + 32 = 42$ numbers in S. The largest number is

$$y_1 = 100,001,110,000,000,000,000.$$

(b) Is there a subset A of S that sums to t? If yes, provide the members of A (write each number using its identifier that is provided in the reduction). In any case, defend your answer. (10 pts)

Solution. Since assignment $\alpha = (x_1 = 0, x_2 = 1, x_3 = 1, x_4 = 0, x_5 = 1)$ satisfies $\mathcal{C}$, it follows that

$$A = \{z_1, y_2, y_3, z_4, y_5, g_1, h_1, g_2, h_2, g_3, h_3, g_4, g_5, h_5, g_6, g_8,$$
$$g_9, g_{10}, h_{10}, g_{11}, h_{11}, g_{12}, g_{13}, g_{15}, h_{15}, g_{16}, h_{16}\}$$

sums to $t = 1111133333333333333333$.

(c) For the mapping reduction f from `3SAT` to `DHP` provided in lecture, describe the orientation of the edges (lc_8, c_8) and (rc_8, c_8), that connect Diamond 3 to clause vertex c_8. Explain. (5 pts)

Solution. $c_8 = (x_2, \overline{x}_3, x_5)$, which means that c_8 can only be satisfied when x_3 is assigned 0. Hence we should only be able to visit c_8 when moving from left to right through the x_3-diamond. In other words, we should have a directed edge directed from lc_8 and towards c_8, and also a directed edge from c_8 to rc_8.

4. Provide the state diagram for a Turing machine that, on input $x \in \{0,1\}^+$, replaces x with x 1's. Here, we are thinking of x as a binary number. For example, 1001 would be replaced with 111111111, while 00 is replaced with λ (i.e. a tape with all blanks). Simulate your solution using the online Turing machine simulator and provide an in-class demo that includes an explanation of your algorithm. (20 pts)

Solution. Solutions may vary, but the most efficient solution alternately subtracts 1 from x and prints an additional 1 to the right of x (the printed 1's may be separated from x by a # symbol). To subtract 1 from the current value of x, moving from the LSB to the MSB and until the first 1 is encountered, change 0's to 1's. Then change the first-encountered 1 to a 0 (if no such 1 exists, then the machine erases all the 0's and the separating # and halts). Once the 1 is changed to a 0, the machine moves to the far right and prints the next 1 to the right of the # which separates the printed 1's from x.

5. For the Turing machine M with state diagram shown below, draw the computation tree $T(M, 001)$. Is this an accepting computation? Explain. (15 pts)