Problem Set 3

15-859 Information Theory and Applications in TCS
Name: Shashank Singh
Email: sss1@andrew.cmu.edu
Due: Thursday, March 28, 2013

Problem 1

$\forall n \in \mathbb{N}$, let F_n denote the permutation matrix defined by

$$(F_n \mathbf{x}) = \begin{bmatrix} x_1 & x_3 & \cdots & x_{2^n-1} & x_2 & x_4 & \cdots & x_{2^n} \end{bmatrix}.$$

It is clear from the permutation of the elements in each step of the recursive construction of $G_2^{\otimes n} B_n$ that, for all postitive $n \in \mathbb{N}$ ($B_0 = 1$),

$$B_n = F_n \begin{bmatrix} B_{n-1} & 0 \\ 0 & B_{n-1} \end{bmatrix}$$

(in each step, we switch every other element from the two previous recursively constructed halves, as described on page 6 of the relevant class notes).

Since $G_2^{\otimes n}$ is symmetric about its minor diagonal, it is clear by inspection that $G_2^{\otimes n}$ and F_n commute. Thus, if, for some $n \in \mathbb{N}$, $G_2^{\otimes n}$ and B_n commute, then

$$\begin{aligned} G_2^{\otimes(n+1)} B_{n+1} &= G_2^{\otimes(n+1)} F_{n+1} \begin{bmatrix} B_n & 0 \\ 0 & B_n \end{bmatrix} = F_{n+1} G_2^{\otimes(n+1)} \begin{bmatrix} B_n & 0 \\ 0 & B_n \end{bmatrix} \\ &= F_{n+1} \begin{bmatrix} G_2^{\otimes n} B_n & G_2^{\otimes n} B_n \\ 0 & G_2^{\otimes n} B_n \end{bmatrix} = F_{n+1} \begin{bmatrix} B_n G_2^{\otimes n} & B_n G_2^{\otimes n} \\ 0 & B_n G_2^{\otimes n} \end{bmatrix} \\ &= F_{n+1} \begin{bmatrix} B_n & 0 \\ 0 & B_n \end{bmatrix} G_2^{\otimes(n+1)} = B_{n+1} G_2^{\otimes(n+1)}, \end{aligned}$$

and so B_{n+1} and $G_2^{\otimes(n+1)}$. Since $B_0 = 1$ and $G_2^{\otimes 0} = 1$ trivially commute, by induction, B_n and $G_2^{\otimes n}$ commute for all $n \in \mathbb{N}$.

Thus, rather than encoding with $G_2^{\otimes 0} B_n$, we could encode with $G_2^{\otimes n}$, send the result through the channel, and then apply B_n before decoding. Since the channel is memoryless, the order in which the bits are sent does not affect the probability of losing a bit, and hence does not affect the polarization of the input; i.e, after applying B_n to the output, it is still the case that that

$$\lim_{n \to \infty} \mathsf{Pr}_i \left(H(U_i \,|\, U_0^{i-1}, Y_0^{N-1}) \in (\epsilon, 1-\epsilon) \right) = 0. \quad \blacksquare$$

Problem 2

(a)

(b)

$$\begin{aligned}
Z(U_1 \mid (Y_0, Y_1, U_0)) &= \sum_{(y_0,y_1,u_0)\in\{0,1\}^3} \sqrt{\mathsf{Pr}\,(u_1 = 0 \mid y_0, y_1, u_0)\,\mathsf{Pr}\,(u_1 = 1 \mid y_0, y_1, u_0)} \\
&= \frac{1}{2} \sum_{(y_0,y_1,u_0)\in\{0,1\}^3} \sqrt{\mathsf{Pr}\,(u_1 = 0, u_0 = 0 \mid y_0, y_1)\,\mathsf{Pr}\,(u_0 = 0 | y_0)} \\
&\cdot \sqrt{\mathsf{Pr}\,(u_1 = u_0 \otimes 1 \mid y_0, y_1, u_0)\,\mathsf{Pr}\,(u_0 = 1 | y_0)} \\
&= \left(\sum_{Y_0\in\{0,1\}} \sqrt{\mathsf{Pr}\,(X_0 = 0 \mid Y_0)\,\mathsf{Pr}\,(X_0 = 1 \mid Y_0)} \right)^2 = Z(X_0 \mid Y_0)^2. \quad \blacksquare
\end{aligned}$$

Problem 3

(i) Since C_i is added to the tree after C_j, by the construction rules, if C_i and C_j are not disjoint, then C_i is appended to either C_j or some node deeper in the tree than C_j, so that C_i is strictly deeper in T_t than C_j. $\blacksquare$

(ii) Given a tree T_t from the algorithm, we can uniquely determine the locations in R from which the algorithm took random values for each C_i as follows:

- For each variable v appearing (using a new copy of T_t each time):
  - While there is a clause appearing in the tree that contains v:
    1. Let C_i denote the deepest clause in the tree that contains v.
    2. C_i used the first value in the column of R corresponding to v that has not been used so far.
    3. Remove C_i and all of its descendants from the tree.

The uniqueness of C_i chosen in step 1., and hence the uniqueness of the reconstruction, follow from part (i), since a each v can appear in at most one clause at each depth in T_t. Correctness of the reconstruction is immediate from the construction of T_t and the definition of R. $\blacksquare$

(iii) If the algorithm runs for at least Mm steps then, by the Pigeonhole Principle, some clause C must have its variables reassigned at least M times. Consider constructing a tree T as described, with C_1 the first occurence of C and C_t the last occurence of C. Then, since T is rooted at C, every occurence of C will be in T, and hence T will contain at least M nodes. Furthermore, it is clear from the algorithm and part (i) that T will be legally labeled and consistent with R.

(iv) Consider the following procedure for creating a directed graph on M nodes:

(1) Choose a clause to be the root node.

(2) From the set of ordered pairs of clauses which intersect, choose

(3) For each pair, add a (directed) edge from the first clause to the second.

Since edges in legally labeled trees can only join intersecting clauses, every legally labeled tree can be constructed in exactly one way by the above procedure by identifying the graph as a legally labeled tree when there is a unique path from the chosen root node to each of the other $M-1$ (although the procedure will also construct many graphs which are *not* legally labeled trees) Recalling that a graph on M nodes is a tree (if and) only if it has $M-1$ edges and there is a unique path between any two nodes.

Step (1) can can be done in m ways. Since each clause intersects at most Δ other clauses, there are at most ΔM ordered pairs from which the $M-1$ edges are chosen, so that Step (2) can be done in at most $\binom{\Delta M}{M-1}$ ways, giving an upper bound of

$$\binom{\Delta M}{M-1} \leq \binom{\Delta M}{M} \leq \frac{(\Delta M)^M}{M!} \leq \left(\frac{\Delta M e}{M}\right)^M = (\Delta e)^M,$$

where the last inequality follows from Stirling's Approximation. ■

(v) Define, for notational convenience, $\alpha := \Delta 2^{-k} e$. Since a random assignment fails to satisfy a k-clause with probability 2^{-k}, the probability over R that a legally labeled tree on M nodes is consistent with R is $\left(2^{-k}\right)^M = 2^{-kM}$. By the union bound, the probability that there exists a legally labeled tree on M nodes that is consistent with R is, by part (iv), at most $m(\Delta e)^M 2^{-kM} = m\alpha^M$, so that the desired probability is, by the geometric series formula, assuming $\Delta < 2^k/e$, at most

$$\sum_{i=M}^{\infty} m\alpha^M = \frac{m\alpha^M}{1-\alpha}. \quad ■$$

(vi) Let W be a random variable denoting the number of steps taken by the algorithm. By part (iii), $\mathsf{Pr}\,(W = mM)$ is at most the probability that there is a legally labelled tree with at least M clauses that is consistent with R, so that, by part (v),

$$\begin{aligned}
E[W] = \sum_{M=0}^{\infty} mM \cdot \mathsf{Pr}\,(W = mM) &\leq m \sum_{M=0}^{\infty} M \cdot \frac{m\alpha^M}{1-\alpha} \\
&= \frac{m^2\alpha}{1-\alpha} \sum_{M=0}^{\infty} M\alpha^{M-1} \\
&= \frac{m^2\alpha}{1-\alpha} \frac{d}{d\alpha} \sum_{M=0}^{\infty} \alpha^M \\
&= \frac{m^2\alpha}{1-\alpha} \frac{d}{d\alpha} \frac{1}{1-\alpha} \\
&= \frac{m^2\alpha}{1-\alpha} \frac{1}{(1-\alpha)^2} \in O(m^2).
\end{aligned}$$

I couldn't figure out how to tighten this into $O(m \log m)$...

Problem 4

(a) Suppose non-zero $(v_1, v_2) \in \{0,1\}^{2n}$ with $v_1, v_2 \in \{0,1\}^n$. If $\alpha_1, \alpha_2 \in \{0,1\}^n$ with $\alpha_1 \cdot v_1 = v_2 = \alpha_2 \cdot v_2$, by the properties of multiplication and the fact that any element is its own inverse under XOR,

$$0 = \alpha_1 \cdot v_1 + \alpha_2 \cdot v_1 = (\alpha_1 \cdot \alpha_2) \cdot v_1,$$

so that either $v_1 = 0$ (which can't be the case, as $v \neq 0$) or $\alpha_1 \cdot \alpha_2 = 0$, and so $\alpha_1 = \alpha_2$. Thus, each non-zero code is associated with at most one code in $\mathcal{W}$.

Let $Z := \{(\underline{0}, x) : x \in \{0,1\}^n\}$, and, for each $\alpha \in \{0,1\}^n$, let $S_\alpha := \{(x, \alpha \cdot x) : x \in \{0,1\}^n\}$.

To show that each non-zero vector is associated with at least one code in $\mathcal{W}$, it suffices to show that the sets S_α together with Z cover $\{0,1\}^{2n}$, which follows from observing that

$$\left| Z \cup \bigcup_{\alpha \in \{0,1\}^n} S_\alpha \right| = |Z| + \left(\sum_{\alpha \in \{0,1\}^n} |S_\alpha| - 1 \right) = 2^n + 2^n \cdot (2^n - 1) = 2^{2n} = |\{0,1\}^{2n}|,$$

(the first equality holds because we showed that Z and S_α sets intersect precisely on $\underline{0}$). ■

(b) Let $C_{\alpha,J}$ denote the code resulting from using C_α and then erasing the values in positions indexed by J. Since C_α is a linear code, $C_{\alpha,J}$ is also linear. Hence, $C_{\alpha,J}$ is injective if and only if its null-space is trivial (i.e., $\mathcal{N}(C_{\alpha,J}) = \{\underline{0}\}$), which is the case precisely when C_α contains no codewords supported entirely on J. ■

(c) By part (b), it suffices to find the fraction of codes in $\mathcal{W}$ for which no codeword is entirely supported on J. The number of code works supported entirely on J is $2^{|J|} = 2^{n(1-\gamma)}$, so that, since each codeword is associated with at most one code in $\mathcal{W}$, for $c := \gamma$, the fraction in question is at most

$$\frac{2^{n(1-\gamma)}}{2^n} = 2^{-\gamma n} = 2^{-cn}. \quad ■$$

(d) Chernoff bounds give that

$$\mathsf{Pr}\left(|J| \geq (1+\delta)pn\right) < e^{-\frac{\delta^2 np}{3}} = 2^{-c_1 \delta^2 n},$$

for $c_1 := (p/3)\log 2$.

Also, by Markov's inequality,

$$\begin{aligned}\mathsf{Pr}_\alpha\left(\mathsf{Pr}_J\left(\mathrm{err}(J,\alpha)\right) \geq 2^{-c'n}\right) &\leq 2^{c'n} \mathop{\mathbb{E}}_J\left[\mathsf{Pr}_\alpha\left(J \text{ causes an error for code } \alpha\right)\right] \\ &\leq 2^{c'n} \sum_{i=1}^{2n} 2^{2n-i} \mathop{\mathbb{E}}_J\left[\mathsf{Pr}_\alpha\left(J \text{ causes an error for code } \alpha\right)\right]\end{aligned}$$

(e) As with the original construction from class, when a block is correctly decoded, all bits in that block are correctly decoded, and hence it suffices to show that the probability of incorrectly decoding a block is at most $(f(\epsilon))^2$, assuming the outer code can correct some fraction $f(\epsilon)$ of errors. By choosing

$$n \geq -2 \log_{2^{c'}}(f(\epsilon)),$$

it follows from part (d) that, for each block, the probability of a decoding error is at most

$$2^{-c'n} \cdot 1 + (1 - 2^{-c'n}) \cdot 2^{-c'n} \leq (f(\epsilon))^2. \quad \blacksquare$$