1. Theory of Linear Inequalities

For each $k = 0, \ldots, n$, let U_k denote the set of all vectors $x \in \mathbf{R}^n$ such that x has exactly k coordinates equal to 1/2 and all other coordinates of x are either 0 or 1. Let $P \subseteq \{x \in \mathbf{R}^n : 0 \leq x \leq 1\}$ be a polyhedron and let P' denote its Chvátal closure. Show that if $U_k \subseteq P$ for some $k < n$, then $U_{k+1} \subseteq P'$.

Solution. A solution is available upon request.

2. Combinatorial Optimization

Let $G = (V, E)$ be a complete graph. For a vertex $v \in V$ let $\delta(v)$ denote the set of edges having v as an end; for $S \subset V$ let $\gamma(S)$ denote the set of edges having both ends in the set S; for a set $F \subset E$ let $x(F)$ denote $\sum(x_e : e \in F)$.

A Hamiltonian circuit in G is an integer solution to the following linear system:

$$\begin{aligned} x(\delta(v)) &= 2, \text{ for all } v \in V \\ x(\gamma(S)) &\leq |S| - 1, \text{ for all } S \neq V, |S| \geq 3 \\ 0 \leq x_e &\leq 1, \text{ for all } e \in E. \end{aligned}$$

By a *comb* we refer to a non-empty *handle* $H \subseteq V, H \neq V$ and $2k+1$ pairwise disjoint, non-empty *teeth* $T_1, T_2, ..., T_{2k+1} \subseteq V$, for k at least 1. We require each tooth T_i to have at least one vertex in common with the handle and at least one vertex not in the handle.

Show that the comb inequality

$$x(\gamma(H)) + \sum_{i=1}^{2k+1} x(\gamma(T_i)) \leq |H| + \sum_{i=1}^{2k+1} (|T_i| - 1) - (k+1).$$

is satisfied by all Hamiltonian circuits of G by deriving the inequality as a Chvátal cut for the above linear system.

NOTE: No credit is given for an alternative proof that comb inequalities are satisfied by Hamiltonian circuits. Such a proof is given, for example, on page 988 of Schrijver's *Combinatorial Optimization.*

Solution. A solution is available upon request.

3. Analysis of Algorithms

We say that a 4-CNF formula is *strongly satisfiable* if it has an assignment that satisfies at least 2 literals in each clause. Design a polynomial time randomized algorithm that given a 4-CNF formula Ψ which is strongly satisfiable, finds a satisfying assignment (in the usual sense) of Ψ. The algorithm is **not** required to find a satisfying assignment if the input formula is not strongly satisfiable (even if it is satisfiable in the usual sense).

Solution. The algorithm is basically the same random walk algorithm for 2-CNF we analyzed in class. That is, starting from any initial assignment (say, all 0) we do the following: if the current assignment satisfies Ψ we are done. Otherwise, we pick a clause that is not satisfied, randomly pick one of its 4 literals, and change the assignment to that literal.

We now claim that if Ψ is strongly satisfiable then the algorithm will find a satisfying assignment within $O(n^2)$ steps. Assume f is some strongly satisfying assignment of Ψ. Since f strongly satisfies Ψ, when we randomly flip a literal we have a probability of at least 1/2 to decrease the hamming distance between the current assignment and f. So we are exactly at the same situation we had for the 2-CNF algorithm, where a random walk starts at some $0 \le k \le n$ and at each step takes a random step to the left or to the right, where the probability of going left (that is, toward 0) is at least 1/2. Hence, as we saw in class, with high probability, within $O(n^2)$ steps we either find a satisfying assignment or we reach 0, meaning that we found f (which is also a satisfying assignment).

4. Graph Algorithms

Design a polynomial-time algorithm for the following problem and prove its correctness.
INPUT: A graph G and edges $e_1, e_2 \in E(G)$.
QUESTION: Do there exist disjoint cycles C_1, C_2 in G such that $e_i \in E(C_i)$?
Note. To receive full credit a complete proof is required; not just a statement of a theorem from the literature.

Solution. This is the Two Disjoint Paths Algorithm of Seymour, Shiloah and Thomassen. It can be found in various places in the literature.

4. Approximation Algorithms

Consider the weighted vertex cover problem on a graph $G(V, E)$ over vertices $V = [n]$, with corresponding costs $c_i > 0$, $i \in [n]$. Show that, for any $\epsilon \in [0, 1)$, the following hold:
(a) The algorithm below is a factor $2/(1-\epsilon)$ approximation algorithm for weighted vertex cover.
(b) The analysis of (a) cannot be improved beyond factor $(2-\epsilon)/(1-\epsilon)$.
<u>Note:</u> You may answer the question for $\epsilon = 0$ for partial credit strictly greater than zero.

Algorithm

1. Initialization:
   - $U \leftarrow E$ (all edges are uncovered)
   - $\forall e \in E$, $y_e \leftarrow 0$
   - $C \leftarrow \emptyset$ (no vertices have been added to the vertex cover)
   - $\forall u \in V = [n]$, $\delta_u \leftarrow c_u$
2. While $U \neq \emptyset$ (thus while C is not a vertex cover) do:
   - Pick an uncovered edge $e \in U$, and let the endpoints of e be $e = (u, v)$
   - $\mu = \min(\delta_u, \delta_v)$
   - $y_e \leftarrow \mu$
   - $\delta_u \leftarrow \delta_u - \mu$
   - $\delta_v \leftarrow \delta_v - \mu$
   - Include in C all vertices having $\delta_i \le \epsilon\, c_i$ and update U:
     - $\forall i \in V = [n]$, if $\delta_i \le \epsilon\, c_i$ then $C \leftarrow C \cup \{i\}$

$U \leftarrow U \setminus \bigcup_{(i,j)\in E: i\in C}\{(i,j)\}$

3. Output C.

Solution.
(a)
Lemma 1 (Invariance): $\forall i \in V$, throughout the execution of the algorithm, the following holds:

$$\left(\sum_{e\in E: e=(i,j)} y_e\right) + \delta_i = c_i \tag{1}$$

Proof (of Lemma 1): Assume that (1) is true at the begining of an iteration of the while loop (it is obviously true at the begining of the first iteration, by the setting of variables at initialization). We can argue that (1) remains true at the end of the iteration of the while loop. In particular, if $e=(u,v)$ was the uncovered edge considered, prior to the updates of y_e, δ_u and δ_v we had:

$$\left(\sum_{e\in E: e=(i,j)} y_e\right) + \delta_i = c_i \quad ,$$

thus

$$\left(\sum_{e\in E: e=(i,j)} y_e\right) + \mu + (\delta_i - \mu) = c_i \quad .$$

This immediately implies that (1) is true after the updates $y_e \leftarrow \mu$, $\delta_u \leftarrow \delta_u - \mu$ and $\delta_v \leftarrow \delta_v - \mu$.

Corollary 1: $\forall i \in V$, at the end of the execution of the algorithm, the following holds:

$$\left(\sum_{e\in E: e=(i,j)} y_e\right) \leq c_i \tag{2}$$

Proof (of Corollary 1): Follows from (1) and the fact that all δ_i's are always non-negative.

Corollary 2: $\forall i \in C$, at the end of the execution of the algorithm, the following holds:

$$\left(\sum_{e\in E: e=(i,j)} y_e\right) \geq (1-\epsilon)c_i \tag{3}$$

Proof (of Corollary 2): Follows from (1) and the fact that vertices are included in C if and only if $\delta_i \leq \epsilon\, c_i$.

Lemma 2 (Duality): Let C_{OPT} be a vertex cover of minimum cost. Then

$$\left(\sum_{e\in E} y_e\right) \leq \sum_{u\in C_{\text{OPT}}} c_u \tag{4}$$

Proof (of Lemma 2):

$$\sum_{e \in E} y_e \leq \sum_{(u,v)=e \in E} |\{u,v\} \cap C_{\text{OPT}}|\, y_e = \sum_{u \in C_{\text{OPT}}} \left(\sum_{v:(u,v)=e \in E} y_e \right) \leq \sum_{u \in C_{\text{OPT}}} c_u \quad ,$$

where the first inequality follows from the fact that any cover, and hence also an optimal cover, includes at least one of the two endpoints of each edge: $1 \leq |\{u,v\} \cap C_{\text{OPT}}| \leq 2$, and the second inequality follows from (2) of the first corollary of Lemma 1.

Lemma 3 (Slackness): Let C be the vertex cover output by the algorithm. Then

$$\sum_{u \in C} c_u \leq \frac{2}{1-\epsilon} \sum_{e \in E} y_e \tag{5}$$

Proof (of Lemma 3):

$$(1-\epsilon) \sum_{u \in C} c_u \leq \sum_{u \in C} \left(\sum_{v:(u,v)=e \in E} y_e \right) = \sum_{(u,v)=e \in E} |\{u,v\} \cap C|\, y_e \leq 2 \sum_{e \in E} y_e \quad ,$$

where the first inequality follows from (3) of the second corollary of Lemma 1, and the second inequality follows from the fact that any cover, and hence also the cover C constructed by the algorithm, includes at least one of the two endpoints of each edge: $1 \leq |\{u,v\} \cap C| \leq 2$.

Finally, combining (5) of Lemma 3 and (4) of Lemma 2 we establish the approximation factor:

$$\sum_{u \in C} c_u \leq \frac{2}{1-\epsilon} \sum_{e \in E} y_e \leq \frac{2}{1-\epsilon} \sum_{u \in C_{\text{OPT}}} c_u \quad .$$

(b) Consider a bipartite graph with vertices $v_1, \ldots, v_n$ on the left, vertices $u_1, \ldots, u_n$ on the right, and n edges $(v_1, u_1), \ldots (v_n, u_n)$. All v_i's have the same cost, say 1. All u_i's have the same cost $1/(1-\epsilon) = 1 + \frac{\epsilon}{1-\epsilon}$.

Clearly, the optimal cover is $C_{\text{OPT}} = \{u_1, \ldots, u_n\}$ which is of cost n.

On the other hand, the algorithm will consider all n edges (v_i, u_i) successively, set $\mu = 1$ in each iteration, update the corresponding $\delta_{v_i} \leftarrow 0$ and $\delta_{u_i} \leftarrow \frac{\epsilon}{1-\epsilon}$, and consequently include in C both v_i and u_i. The final cover C will consist of all vertices v_i and u_i, $1 \leq i \leq n$, and will thus be of cost $n + n/(1-\epsilon) = \frac{2-\epsilon}{1-\epsilon} n$.

5. Graph Theory

Let G be a simple graph on n vertices and m edges. Prove that it has at least $\frac{m}{3n}(4m - n^2)$ triangles.

Solution. For adjacent vertices $u, v \in V(G)$ let $t(u,v)$ denote the number of triangles containing u, v, and let t be the number of triangles in G. Then $\deg(u) + \deg(v) \leq n + t(u,v)$. By summing over all edges $uv \in E(G)$ we obtain $\sum_{uv \in E(G)} (\deg(u) + \deg(v)) \leq nm + 3t$. But

$$\sum_{uv \in E(G)} (\deg(u) + \deg(v)) = \sum_{v \in V(G)} \deg^2(v) \geq \left(\sum_{v \in V(G)} \deg(v) \right)^2 / n = 4m^2/n$$

by the Cauchy-Schwarz inequality. The result follows.

6. Probability

A **superinversion** of a permutation σ on $\{1, \ldots, n\}$ is a pair (i, j) satisfying the following two conditions:

$$\begin{aligned} (\mathbf{i}) \quad j - i &> \frac{n}{4} \\ (\mathbf{ii}) \quad \sigma(i) - \sigma(j) &> \frac{n}{4}. \end{aligned}$$

Let X_n be the number of superinversions of a permutation chosen uniformly at random from all $n!$ permutations on n elements.
(a): Compute, up to first order, $\mathbf{E}(X_n)$.
(b): Let ϵ be fixed and arbitrary. Show that

$$\mathbf{P}\left((1-\epsilon)\mathbf{E}(X_n) < X_n < (1+\epsilon)\mathbf{E}(X_n)\right) = 1 - o(1)$$

as n tends to infinity. **Solution.** (a): Assume n is divisible by 4 (this only effects the lower order terms)

The number of pairs (i, j) with $j > i + n/4$ is

$$\sum_{i=1}^{\frac{3n}{4}} \left(n - i - \frac{n}{4}\right) = \sum_{i=1}^{\frac{3n}{4}} \left(\frac{3n}{4} - i\right) = \left(\frac{9}{32} + o(1)\right) n^2 .$$

By symmetry, this is also the number of pairs (k, l) with $k - l > n/4$. It follows that for any individual pair (i, j) satisfying (**i**), the probability it satisfies (**ii**) is $(9/32 + o(1))$. By linearity of expectation, we have

$$\mathbf{E}(X_n) = \left(\frac{9}{32} + o(1)\right)^2 n^2 = \left(\frac{81}{1024} + o(1)\right) n^2 .$$

(b): **Method 1:** By Chebyshev's inequality, it suffices to show $Var(X_n) = o(n^4)$. We write

$$\mathbf{E}(X_n^2) = \sum_{(i,j),(k,l)} \mathbf{P}((i,j) \text{ and } (k,l) \text{ both superinversions })$$

and split the sum up into two parts.

Class 1: Pairs where $i = k$ or $j = l$. There are n^3 such pairs, and each can contribute at most 1 to the sum. So this part is $o(n^4)$.

Class 2: Pairs where $i \neq k$ and $j \neq l$. Then the probability that both form a superinversion is zero unless both pairs satisfy (**i**). If both pairs do satisfy (**i**) it follows from independence and direct computation that the probability both satisfy (**ii**) is $(9/32 + o(1))^2$. It follows that the total contribution from this class is $(9/32 + o(1))^4 n^4$.

This gives $\mathbf{E}(X_n^2) = \mathbf{E}(X_n)^2(1 + o(1))$, which is what we needed.

Method 2: We view σ as being formed by putting first choosing $x_1, \ldots, x_n$ uniformly and independently from $[0, 1]$, then ordering them from smallest to largest. Under this new model, changing x_i does not impact the relative order of the x_j for $j \neq i$, so can only create or destroy at most n superinversions. It follows from Azuma's inequality (the method of bounded differences) that

$$\mathbf{P}\left(|X_n - E(X_n)| \geq \lambda n^{3/2}\right) \leq e^{-\lambda^2/2}$$

(NOTE : The second method is slicker than the first method, but depends heavily on the knowledge of the trick of making the x_i independent.)

7. Algebra

Let R be an integral domain and suppose that $R[x]$ is a principal ideal domain. Show that R is a field.

Solution. Let $a \in R$ be nonzero. We need to show that a is a unit. Consider the ideal (a, x) in $R[x]$. It consists of polynomials whose constant term lies in (a). Since $R[x]$ is a principal ideal domain, there is a $p(x) \in R[x]$ so that $(a, x) = (p(x))$. The elements of $(p(x))$ are all of the form $rp(x)$ for $r \in R$. So in particular, $p(x)$ is nonzero and divides both a and x. Since $p(x)$ divides a, it must be that $p(x) = c$ for some nonzero $c \in R$, that is, $p(x)$ is a nonzero constant polynomial. Since $p(x) = c$ also divides x, there is a degree one polynomial $q(x) = bx + d$ such that $p(x)q(x) = x$, that is, $c(bx + d) = x$. Rewriting, we have $(cb)x + cd = x$. Thus, $cb = 1$. This means that c is a unit in R, and so $1 \in (c)$. Since $(a, x) = c$, we then have $1 \in (a, c)$. This means

$$s(x)a + t(x)x = 1$$

for some $s(x), t(x) \in R[x]$. By considering degrees, we find that $t(x) = 0$ and $s(x)$ is a constant polynomial $s(x) = a'$. Rewriting, we have $a'a + 0x = a'a = 1$, and so a is a unit.