CS599 (Spring 2007) — Final Exam Solutions

Problem 1

The primal LP is

$$\begin{array}{ll} \text{Minimize} & \sum_e y_e \\ \text{subject to} & s_i + t_i \geq 1 \quad \text{for all } i \\ & y_e \geq s_i \quad \text{for all } i \text{ and } e \in S_i \\ & y_e \geq t_i \quad \text{for all } i \text{ and } e \in T_i \\ & s_i, t_i, y_e \geq 0, \end{array}$$

and the dual:

$$\begin{array}{ll} \text{Maximize} & \sum_i \gamma_i \\ \text{subject to} & \sum_{i:e \in S_i} \alpha_{e,i} + \sum_{i:e \in T_i} \beta_{e,i} \leq 1 \quad \text{for all } e \\ & \gamma_i \leq \sum_{e \in S_i} \alpha_{e,i} \quad \text{for all } i \\ & \gamma_i \leq \sum_{e \in T_i} \beta_{e,i} \quad \text{for all } i \\ & \gamma_i, \alpha_{e,i}, \beta_{e,i} \geq 0 \end{array}$$

We can interpret the goal as packing as much as possible into sets i (γ_i). Each i needs to acquire "two types" of room in S_i and T_i. Each element can only provide one unit of room total to all sets it belongs to.

In order to raise the objective function, we need to increase the γ_i. In order to do that, the $\alpha_{e,i}$ and $\beta_{e,i}$ need to be raised. It never helps to raise the sum of $\alpha_{e,i}$ higher than that of $\beta_{e,i}$ (or vice versa), as γ_i couldn't be raised more anyway. So the algorithm will be as follows:

All dual variables start at 0. For each i, in an arbitrary order (in parallel would work as well), we do the following. While at least one $e \in S_i$ and at least one $e' \in T_i$ is not tight, increase γ_i at unit rate. If currently, k_i elements $e \in S_i$ and k'_i elements $e' \in T_i$ are not tight, raise each such $\alpha_{e,i}$ at rate $1/k_i$, and each such $\beta_{e',i}$ at rate $1/k'_i$. Notice that this means that both sums will increase at the same rate as γ_i.

As soon as one (or more) such e or e' become tight (first dual constraint), all those elements are picked (their y_e set to 1). None of their duals $\alpha_{e,i'}$ or $\beta_{e,i'}$ will ever be raised again. If all $e \in S_i$ are now tight, we set $s_i = 1$, and if all $e \in T_i$ are tight, we set $t_i = 1$ (if both are set, either we keep both, or drop one arbitrarily). Otherwise, we continue raising as above. Once we are done with all i, the algorithm terminates.

This gives a feasible solution, because we only move on from an i once all of its $e \in S_i$ or all $e' \in T_i$ are tight, which means that the elements are picked.

To prove the approximation, notice that because each picked element was tight, we have $y_e = 1 = \sum_{i:e \in S_i} \alpha_{e,i} + \sum_{i:e \in T_i} \beta_{e,i}$ for all picked elements. And because $\gamma_i = \sum_{e \in S_i} \alpha_{e,i} = \sum_{e \in T_i} \beta_{e,i}$ for all i throughout the algorithm (by our raising strategy), we get

$$\sum_e y_e = \sum_e \left(\sum_{i:e \in S_i} \alpha_{e,i} + \sum_{i:e \in T_i} \beta_{e,i}\right) = \sum_i \left(\sum_{e \in S_i} \alpha_{e,i} + \sum_{e \in T_i} \beta_{e,i}\right) = \sum_i 2\gamma_i.$$

The latter is exactly twice the dual objective value, and hence at most twice the optimum cost by weak LP duality.

Problem 2

We write $x_v = 1$ if v is on the s-side of the cut, and $x_v = 0$ otherwise. Also, $y_e = 1$ if edge e is cut, and $y_e = 0$ otherwise. The LP is then:

$$\begin{array}{ll} \text{Minimize} & \sum_e c_e y_e \\ \text{subject to} & x_s = 1, x_t = 0 \\ & y_e \geq x_u - x_v \quad \text{for all } e = (u,v) \\ & y_e \geq x_v - x_u \quad \text{for all } e = (u,v) \\ & \sum_v x_v \leq k \\ & x_v, y_e \geq 0 \end{array}$$

The fractional y_e can be interpreted as edge lengths, and the x_u as distances from t. To round, we pick a radius $\rho \in [\frac{1}{2}, 1]$ uniformly at random, and define $S := \{v \mid x_v \geq \rho\}$. This defines an s-t cut, because $s \in S$ and $t \notin S$.

Because all nodes v in S had $x_v \geq \rho \geq \frac{1}{2}$, there can be at most $2k$ of them (or the last constraint would have been violated, as $\sum_v x_v > 2k \cdot \frac{1}{2} = k$). For each edge e, the probability that it is cut is the probability that ρ is between x_u and x_v, which is at most $2|x_u - x_v|$, because the probability density over $[0, 1]$ is bounded above by 2. Thus, the total expected cost is $\sum_e c_e \cdot \text{Prob}[e \text{ is cut}] \leq \sum_e c_e \cdot 2y_e$, which is twice the LP cost.

Problem 3

(a) It is true that each cycle can be embedded into sufficiently high-dimensional ℓ_1 space. There are several ways to prove this. One could give a cut packing, or prove something about integrality gaps for sparsest cuts. Or one could just give the embedding. Again, there are different possible embeddings.

For an even n, we can embed the cycle into $\mathbb{R}^{n/2}$. Nodes are numbered from $0, \ldots, n-1$. Nodes $i = 0, \ldots, n/2 - 1$ are embedded to $1^i 0^{n/2-i}$ (meaning: the first i coordinates are 1, the remaining $n/2 - i$ are 0.) Nodes $i = n/2, \ldots, n-1$ are embedded to $0^{i-n/2} 1^{n-i}$. Clearly, if both nodes i and j are less than $n/2$, or both are at least $n/2$, then they differ exactly in the coordinates between i and j (or $i - n/2$ and $j - n/2$), hence their distance is $|i - j|$, which is also their distance on the cycle. Otherwise, w.l.o.g., $i < n/2 \leq j$. Then, if $j - n/2 \geq i$, the embeddings agree in coordinates between i and $j - n/2$ (which are 0 in both), and disagree in all others (the coordinates before i are 1 in the embedding of i and 0 in the embedding for j; the coordinates greater than $j - n/2$ are 0 for i, and 1 for j.) Hence, the distance in the embedding is $n/2 - (j - n/2 - i) = n - j + i$, which by assumption is also the distance on the cycle. If $j - n/2 < i$, the embeddings agree in the same coordinates (which are now one for both), and the distance in the embedding is $n/2 - (i - (j - n/2)) = j - i$, again the distance on the cycle.

For odd n, we first subdivide each edge by adding a vertex in the middle. This gives an even cycle of length $2n$, which we now embed as before. Now, we remove all of the artificial vertices. The distances are just scaled up by a factor of 2, so we simply multiply each coordinate by $\frac{1}{2}$. This gives an embedding into $\mathbb{R}^n$.

(b) It is not true that each complete bipartite graph can be embedded into ℓ_1. In fact, it fails for $K_{3,2}$, with vertex sets $X = \{x_1, x_2, x_3\}$ and $Y = \{y_1, y_2\}$. The simple and elegant solution is to define a demands MCF instance with demand 1 between the four pairs $(x_1, x_2), (x_2, x_3), (x_3, x_1), (y_1, y_2)$. The sparsest cut has sparsity 1, but the largest MCF is 3/4: each unit of demand routed uses up at least two units of edge capacity (since all paths have length at least 2). Since the 6 edges have total capacity 6, at most 3 units total can be routed, so at most 3/4 per commodity. Thus, the integrality gap is strictly larger than 1. On the other hand, an isometric embedding would imply an integrality gap of 1.

The pedestrian way is to do the following. Assume that we have an isometric embedding in $\mathbb{R}^d$ for some d. Consider the three points at which x_1, y_1, y_2 are embedded. By rotating and translating the space (which does not affect edge lengths), we can assume w.l.o.g. that x_1 maps to the origin, y_1 to $(1, 0, 0, \ldots, 0)$, and y_2 lies in the plane spanned by the first two coordinates (three points always lie in a 2-D plane). Again, by rotating the space, we can assume w.l.o.g. that y_2 lies on the line segment $(-1, 0, 0, \ldots, 0) - (0, 1, 0, 0, \ldots, 0)$ (that line segment, and its mirror image, are the only points at distance 1 from x_1 and at distance 2 from y_1).

If y_2 were anywhere but at $(0, 1, 0, 0, \ldots, 0)$, there would be no point at distance 1 from y_2 and y_1, but at distance at least 1 from x_1. (Any such candidate point must lie on a shortest path from y_1 to y_2 (hence in the same plane), but also at distance 1 from x_1.) Thus, y_2 must be at $(0, 1, 0, 0, \ldots, 0)$. Still, both x_2 and x_3 must lie on shortest y_1-y_2 paths (with respect to ℓ_1 distance). In particular, they cannot differ from y_1, y_2 in any coordinates but the first two, and must lie in the square with corners y_1, y_2. But there is only one point in that square at distance 2 from x_1, so we cannot embed both x_2 and x_3 at distance 2 from each other.

Problem 4

The problem is really just MAX 3-WAY CUT, rather than a coloring problem.

(a) The algorithm simply assigns each node an independent and uniformly random color from among the three choices. Each edge has differently colored endpoints with probability 2/3, so by linearity of expectation, at least $2m/3$ edges are satisfied in expectation. The OPT is bounded by m, so we have a 2/3 approximation.

(b) For the hardness, we reduce approximation-preservingly from MAX 3-SAT. We proved in class that there is a $\gamma(= 1/8)$ such that MAX 3-SAT is hard to approximate within $1 - \gamma$. The reduction is the standard NP-hardness reduction for 3-SAT with some duplicated edges. You can look up the gadgets in CLRS or Kleinberg/Tardos, among others.

We start with three nodes B, T, F, which will have m edges between each pair. For each variable x_i, we will have nodes $x_i, \overline{x}_i$, each connected to B and to each other with k_i parallel edges (where k_i is the sum of the number of negated and unnegated appearances of x_i in the formula). Finally, for each clause C_j, we have one copy of the standard clause gadget with 13 edges. Thus, the reduction is polynomial. The total number of edges between B, T, F is $3m$, the total number of edges between variable nodes and B is $9m$ (the total number of all variable occurrences is $3m$, the number of literals in clauses, and we have three edges for each). The total number of edges in clause gadgets is $13m$, so we have $25m$ edges.

If the formula was satisfiable, then all $25m$ edges can be satisfied, by producing a valid coloring (all true literals get the same color as the node T, all false literals the same color as F, and the clause gadgets can be pretty easily colored).

Conversely, suppose that at most $7m/8$ clauses can be satisfied, and look at the best possible coloring of the graph. First, it is easy to satisfy more than $24m$ edges, by choosing an arbitrary variable assignment (not necessarily satisfying at all), coloring all variable nodes (and B, T, F) accordingly, and simply violating at most one edge for each clause gadget. Thus, the optimum solution must satisfy all edges between B, T, F (otherwise, it would satisfy at most $24m$ edges). Similarly, suppose the optimum solution does not satisfy the edges between $x_i, \overline{x}_i$ and B (any one of the three). It loses k_i edges (at least). By coloring them consistently and arbitrarily, and recoloring the clause gadgets, it will gain those k_i, and lose at most k_i edges in clause gadgets (one each), so w.l.o.g. , all variables are consistent. Thus, we can associate with the coloring an assignment: each variable x_i colored with the same color as T is true, the others are false. For this variable assignment, if a clause is satisfied, then all edges in the gadgets can be satisfied; otherwise, at most 12 of the 13 edges can be satisfied. Thus, if at most $7m/8$ clauses can be satisfied for the formula, at least $m/8$ edges must remain unsatisfied, and the fraction of correct edges is at most $\frac{24+7/8}{25} = \frac{199}{200}$. Thus, we proved that this problem cannot be approximated better than $\frac{199}{200}$ unless $P = NP$.