HW 4: due Tue, May 2

1. We considered n jobs $\{j_1, \ldots j_n\}$ that take processing times $\{p_1, \ldots, p_n\}$ each of which must be scheduled, without interuption on one of a on a set of m identical machines. We wish to find the schedule that completes the final job in the soonest amount of time. We considered the greedy scheduling algorithm, which orders the jobs arbitrarily, and then schedules the next job on the next available machine, and showed that it gave a 2-approximation to the optimal schedule. Show that in fact it gives a $2 - 1/m$-approximation to the optimal schedule, where m is the number of machines.

2. The *fault-tolerant* version of the k-center problem with triangle inequality has an additional input $\alpha \leq k$ which specifies the number of centers that each vertex must be connected to. In other words, we assume that up to $\alpha - 1$ centers might be closed, and so the fault-tolerant cost for a vertex is its distance to its αth closest center. The problem is to pick k centers so that the maximum fault-tolerant cost of a vertex is minimized. A set $S \subseteq V$ in an undirected graph $H = (V, E)$ is an α-*dominating set* if each vertex $v \in V$ is adjacent to at least α vertices in S (we consider a vertex to be adjacent to itself). Let $dom_\alpha(H)$ denote the size of a minimum cardinality α-dominating set in H.

   (a) Let I be an independent set in H^2. Show that $\alpha|I| \leq dom_\alpha(H)$.

   (b) Give a factor 3 approximation algorithm for the fault-tolerant k-center problem (Hint: Compute a maximal independent set M_i in G_i^2, for $1 \leq i \leq m$. Find the smallest index i such that $|M_i| \leq \lfloor \frac{k}{\alpha} \rfloor$, and moreover, the degree of each vertex of M_i in G_i is $\geq \alpha - 1$.)

3. Consider a bipartite graph $G = (U, V, E)$ on $2n$ vertices that contains a perfect matching. Suppose the vertices in U arrive in an online fashion and the edges incident to each vertex $u \in U$ are revealed when u arrives. When this happens, the algorithm may match u to a previously unmatched adjacent vertex in V, if there is one. Such a decision,

once made is irrevocable. The objective is to maximize the size of the resulting matching.

(a) Consider the algorithm that always matches a vertex in U if a match is possible. Show that this algorithm achieves a competitive ratio of $1/2$.

(b) (optional) Consider the following randomized online matching algorithm: 1) Randomly rank all the vertices in V. 2) As each vertex in U arrives, match it with the highest rank vertex remaining to which it has an edge. Show that this algorithm does better in expectation than the deterministic algorithm above: you might want to read: Birnbaum, B. and Mathieu, C., 2008. On-line bipartite matching made simple. ACM SIGACT News, 39(1), pp.80-87.