CMPUT 675: Approximation Algorithms and Approximability **Winter 2018**

Lecture 13 (Feb 27th, 2018): Generalized Assignment Problem

Lecturer: Mohammad R. Salavatipour *Scribe: Based on older notes*

13.1 Generalized Assignment Problem (GAP)

Problem Description:
Recall the Generalized Assignment Problem (GAP) from last lecture. In this problem we are given a set J of n jobs, and m unrelated machines. Let p_{ij} be the processing time of job j on machine i and c_{ij} be the cost of running job j on machine i. Let T be the bound by which we want to finish all the jobs. Our goal is to find a scheduling of the jobs on the machines so that all the jobs are done before time T and we minimize the cost of processing these jobs. The following is a natural LP relaxation for this problem known as Generalized Assignment Problem (GAP):

$$\begin{array}{lrcll} \min & \sum_{i,j} c_{ij} \cdot x_{ij} & & & \\ & \sum_i x_{ij} & = & 1 & 1 \leq j \leq n \\ & \sum_j p_{ij} \cdot x_{ij} & \leq & T & 1 \leq i \leq m \\ & x_{ij} & \geq & 0 & \end{array}$$

Note that even to detect if there is a feasible schedule (respecting the deadline bound T) is NP-hard. Therefore, we cannot have an approximation algorithm. Instead, we present a bicriteria algorithm in the sense that it either detects that there is no feasible solution or finds a solution of cost at most OPT but violates the time constraint by a factor ≤ 2. Before we start talking about the algorithm for GAP we talk about the bipartite matching LP.

Bipartite Matching Polytope
Before presenting our algorithm we first present a classical result from combinatorial optimization. Consider a bipartite graph $G = (V \cup U, E)$ with $|U| \leq |V|$. We assume that each edge e has a given cost c_e. We say $M \leq E$ is a complete matching if saturates all of U, i.e. $\forall u \in U$, u has degree 1 in M and $\forall v \in V$, v has degree ≤ 1 and $\forall v \in V$ has degree ≤ 1. We say M is a perfect matching with every vertex of U and V has degree 1 in M (obviously we must have $|U| = |V| = |M|$). We can talk about minimum cost complete matching as well (one that has minimum total cost). We can write the minimum cost complete matching problem as the following integer program:

$$\begin{array}{lrcll} \text{minimize} & \sum_e c_e \cdot x_e & & & \\ & \sum_{u:uv \in E} y_{uv} & \leq & 1 & \forall v \in V \\ & \sum_{v:uv \in E} y_{uv} & = & 1 & \forall u \in U \\ & y_{uv} & \in & \{0,1\} & \end{array}$$

By relaxing the last constraint to $y_{uv} \geq 0$ we obtain an LP.

Theorem 1 *For any bipartite graph $G(U \cup V, E)$, any bfs of the above LP is integral. Also any feasible fractional solution can be turned to an integral solution of no more cost.*

Now back to the GAP, suppose $\bar{x}$ is an optimal solution with cost C^* to the LP presented. So we have a total of $\sum_{j=1}^{n} x_{ij}$ (fractional) jobs assigned to machine i. Suppose we allocate $\lceil \sum_{j=1}^{n} x_{ij} \rceil = k_i$ slots for machine i; think of each as a unit size bin. We build a bipartite graph $B = (J \cup S, E)$ (where S is the set of all slots over all machines) in the following way. For each job j we will have a node in J. We will have a node (i, s) in S for each i, s where i is the ith machine $(1 \leq i \leq m)$ and s is the sth slot $(1 \leq s \leq k_i)$. Consider the jobs assigned by LP to i. They are the only jobs that will have an edge to (i, s) (detailed below). Ideally we would like to have the following properties in our bipartite graph:

1. B has a fractional complete matching for J of cost at most C^*
2. Each integer complete matching on J corresponds to an assignment of jobs to machines of cost at most C and completion time $\leq 2T$.

If we can obtain such a fractional complete matching, then using Theorem 1 we should be able to find an integer solution of cost at most C^* with completion time at most $2T$. Now we describe the edges of the bipartite graph B. Consider a machine i and suppose we sort the jobs in none-increasing order of their size on i, i.e. $p_{i_1} \geq p_{i_2} \geq \cdots p_{k_i}$. Now we consider slots (i, s) for $1 \leq s \leq k_i$ as unit size bins and x_{ij} as fractional pieces of the jobs to be packed in these bins. We go through the jobs in that order and fill slot $(i, 1)$ until it becomes full and we move on to the next slot. If at a point we have a capacity z is left in a bin and for job j we have $x_{ij} > z$ we fill that bin using $x_{ij} - z$ fraction of job j and the rest of that job goes to the next slot. Let $y_{j,(i,s)}$ be the fraction of job j assigned to bin/slot (i, s), $\forall j$. We will have an edge $j, (i, s)$ in B if $y_{j,(i,s)} > 0$, the cost of this edge is set to c_{ij}. Note that each job has fractional degree 1 (since $\sum_i x_{ij} = 1$). (see Figure 13.1).

So the $y_{j,(i,s)}$ constitute a fractional matching (covering all of J) in B and clearly the cost of the matching is at most $\sum_{i,j} c_{ij} x_{ij}$ since we are assigning the jobs fractionally in the same way as the LP does. Now we want to show that the second property mentioned above holds for B. Consider some slot (i, s) and let $\max(i, s)$ be defined to be longest job assigned to to slot (i, s). then if we consider any matching in B the total "load" (sum of processing time of jobs) assigned to machine i is at most:

$$\sum_{s=1}^{k_i} \max(i, s).$$

Also note that each job by itself is most T. Therefore if we show that $\sum_{s=1}^{k_i - 1} \max(i, s) \leq T$ then we have shown $\sum_{s=1}^{k_i} \max(i, s) \leq 2T$. Thus if we find a min-cost matching in B then each machine load is at most $2T$ and we are done. Below we complete this argument.

First note that all the slots except the last one for machine i is full, i.e. $1 \leq s \leq k_i - 1$: $\sum_j y_{j,(i,s)} = 1$. So $\sum_j p_{ij} y_{j,(i,s)}$ is a weighted average of processing times assigned to slot (i, s). Since the jobs are considered in non-increasing order of their processing times, the largest job assigned to slot $s + 1$ is no more than the average assigned to slot s, i.e. $\max(i, s + 1) \leq \sum_j y_{j,(i,s)} p_{ij}$, which implies

$$\sum_{s=1}^{k_i - 1} \max(i, s + 1) \leq \sum_{s=1}^{k_i - 1} \sum_j y_{j,(i,s)} p_{ij} \leq \sum_{s=1}^{k_i} \sum_j y_{j,(i,s)} p_{ij}.$$

Noting that $x_{ij} = \sum_s y_{j,(i,s)}$, and by changing the order of sums in the RHS, we can uppoer bound that expression by $\sum_j \sum_s y_{j,(i,s)} p_{ij} = \sum_j p_{ij} x_{ij} \leq T$.

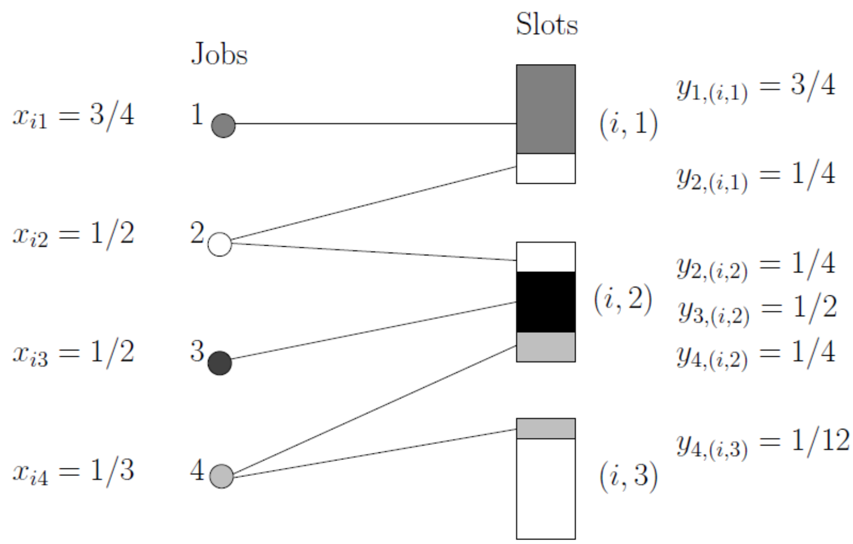

Figure 13.1: Example taken from WS book

13.2 Concentration bounds and Integer Multicommodity flow

In this section we see another problem called Integer Multicommodity flow with the goal of minimizing congestion. Before that, we review some of the basic bounds used to prove concentration of random variables around their mean.

Markov's inequality (first moment method): If X is a random variable taking non-negative values then $\Pr[X \geq a] \leq \frac{\mathrm{E}[X]}{a}$ for any $a > 0$.

For example, consider the problem of flipping a coin n times and let X be the number of heads. Then $\mathrm{E}[X] = n/2$ and using Markov's inequality, $\Pr[X \geq 3n/4] \leq \frac{2}{3}$.

Recall the definition of variance and standard deviation:

Definition 1

$$Var[X] = E[X^2] - (E[X])^2 = E[(X - E[X])^2].$$

$$\sigma(X) = \sqrt{Var[X]}.$$

We can upper bound the probability that a random variable is away from its mean using the variance.

Theorem 2 (Chebyshev's Inequality) *Let X be a random variable. Then for any $\lambda > 0$:*

$$\Pr[(X - E[X]) \geq \lambda] \leq \frac{Var[X]}{\lambda^2}.$$

Proof.

$$\Pr[(X - E[X]) \geq \lambda] = \Pr[(X - E[X])^2 \geq \lambda^2] \leq \frac{E[(X - E[X])^2]}{\lambda^2}$$

by Markov's Inequality. Thus:

$$\Pr[(X - E[X]) \geq \lambda] \leq \frac{Var[X]}{\lambda^2}.$$

■

Equivalently:

$$\Pr[|X - E[X]| \geq tE[X]] \leq \frac{Var[X]}{t^2(E[X])^2}.$$

The use of Markov's inequality is referred to as first moment method and the use of Chebyshev's inequality is referred to as second moment method.

For the example of flipping n coins, using Chebyshev's inequality:

$$\Pr[X > \frac{3n}{4}] \leq \Pr[(X - E[X]) \geq \frac{n}{4}] \leq \frac{n/4}{(n/4)^2} = \frac{4}{n}$$

Theorem 3 (Chernoff Bound) *Assume $X_1, X_2, \ldots, X_n$ are independent Poisson trials with $\Pr[X_i = 1] = p_i, for 0 < p_i < 1$. Then for $X = \sum_{i=1}^{n} X_i$, $\mu = E[X]$:*

1. *for any $0 < \delta$:* $\Pr[X \geq (1+\delta)\mu] \leq \left(\frac{e^\delta}{(1+\delta)^{(1+\delta)}}\right)^\mu$.
2. *for any $0 < \delta \leq 1$:* $\Pr[X \geq (1+\delta)\mu] \leq e^{-\mu\delta^2/3}$.
3. *for any $R \geq 6\mu$:* $\Pr[X \geq R] \leq 2^{-R}$.

Note that these bounds are independent of μ.

Similarly, we can show:

Theorem 4 *Under the same assumptions as in Theorem 3:*

1. *for any $0 < \delta < 1$:* $\Pr[X \leq (1-\delta)\mu] \leq \left(\frac{e^{-\delta}}{(1-\delta)^{(1-\delta)}}\right)^\mu$.
2. *for any $0 < \delta < 1$:* $\Pr[X \leq (1-\delta)\mu] \leq e^{-\mu\delta^2/2}$.

Corollary 1 *For $0 < \delta < 1$:*

$$\Pr[|X - \mu| \geq \delta\mu] \leq 2e^{-\mu\delta^2/3}.$$

13.2.1 Example : Flipping n unbiased coins

Suppose we flip n unbiased coins uniformly randomly and independently. Let $X_i = 1$ if coin i is heads and $X = \sum X_i$ is the number of heads, $E[X] = \frac{n}{2}$. Using the Chernoff bound,

$$\Pr\left[X \geq \frac{n}{2} + \lambda\right] = \Pr\left[X \geq \mu(1 + \frac{\lambda}{\mu})\right] \leq e^{-(\frac{\lambda}{\mu})^2 \frac{\mu}{3}} = e^{\frac{-\lambda}{3\mu}}.$$

Now try $\lambda = \sqrt{3n \ln n}$. Then $\frac{\lambda^2}{3\mu} = \frac{3n \ln n}{3\frac{n}{2}} = 2 \ln n$. So we have

$$\Pr[X \geq \mu + \sqrt{3n \ln n}] \leq e^{-2 \ln n} = n^{-2}.$$

This is a better bound than given by Chebyshev's inequality: $\sigma^2[X] = \sum \sigma^2[X_i] = \frac{n}{4}$ and

$$\Pr[X \geq \mu + \lambda] \leq \frac{\sigma^2}{\lambda^2} \in \frac{O(n)}{O(n \ln n)} = O(\frac{1}{\ln n}).$$

13.2.2 Integer Multicommodity Flow

Consider the following multicommodity network problem. Given a graph $G = (V, E)$, and k pairs of vertices (source/sink) $\{(s_1, t_1), (s_2, t_2), \ldots, (s_n, t_n)\}$. The problem is to find a feasible solution which consists of a set of n paths $p_1, p_2, ..., p_n$ such that p_i is a path from s_i to t_i. The congestion of each edge e is the number of paths using the edge. Our goal is to minimize the maximum number of paths that use any edge, that is find a feasible solution with minimum maximum congestion.

Denote the set of all paths between s_i and t_i by P_i and let $x_{i,p}$ be the indicator variable that is 1 if we pick path $p \in \mathrm{P}_i$.

$$\begin{array}{ll} \text{minimize} & C \\ \text{subject to} & \sum_{p \in \mathrm{P}_i} x_{i,p} = 1 \quad \forall i \\ & \sum_{p \in \mathrm{P}_i : e \in p} x_{i,p} \leq C \quad \forall e \in E \\ & x_{i,p} \in \{0, 1\} \end{array}$$

This is an integer program formulation of our problem. If we relax the constraint on $x_{i,p}$ to be real value between 0 and 1, i.e. $0 \leq x_{i,p} \leq 1$ (instead of $x_{i,p} \in \{0, 1\}$) we obtain a linear programming relaxation of the problem. Note that the solution to linear program (LP) is no more than the solution to the integral program (IP).

Exercise: We can solve this LP using LP solvers in polynomial time.

One property of the solution to this LP is that it has only a polynomially many number of nonzero variables $x_{i,p}$.

Denote by vector x^* the optimal fractional solution and denote by C^* the value of the optimal fractional solution. For every i, we are going to choose exactly one path from P_i. The probability that we choose $p \in \mathrm{P}_i$ is exactly equal to $x^*_{i,p}$ (note that the sum of these values is 1). Let random variable $Y^i_e = 1$ if and only if the selected path for s_i, t_i contains edge e. Thus, the congestion of edge e is $Y_e = \sum_{i=1}^{n} Y_e^i$.

$$\begin{aligned} \mathrm{E}[Y_e] &= \mathrm{E}\left[\sum_{i=1}^{n} Y_e^i\right] \\ &= \sum_{i=1}^{n} \mathrm{E}[Y_e^i] \\ &= \sum_{i=1}^{n} \Pr[Y_e^i = 1] \\ &= \sum_{i=1}^{n} \sum_{p \in \mathrm{P}_i : e \in p} x_p^* \\ &\leq C^* \end{aligned}$$

If $\mu = E[Y_e] \geq 1$ then let $1 + \alpha = \frac{d \ln n}{\ln \ln n}$ for some constant $d > 0$ where $n = |V|$. So $(1+\alpha)\ln(1+\alpha) - \alpha \geq 3 \ln n$ if d is large enough, which implies:

$$\Pr[Y_e \geq (1+\alpha)\mu] \leq e^{-(3 \ln n)} \leq \frac{1}{n^3}. \tag{13.1}$$

Since $\mu \leq C^* \leq OPT$, (13.1) implies that $\Pr[Y_e \geq (1+\alpha)OPT] \leq \frac{1}{n^3}$. If $\mu < 1$ then let $\alpha\mu = \frac{d \ln n}{\ln \ln n}$ for large enough d. Then $((1+\alpha)\ln(1+\alpha) - \alpha)\mu \geq 3 \ln n$ for large enough d. Therefore:

$$\Pr[Y_e \geq (1+\alpha)\mu] \leq e^{-(3 \ln n)} \leq \frac{1}{n^3}. \tag{13.2}$$

Note that in this case the congestion in optimal (integral) solution is at least 1. So the probability that congestion of e is larger than $O(\frac{\ln n}{\ln \ln n} OPT)$ is at most $\frac{1}{n^3}$ by (13.2). In either case, the probability that for edge e, the congestion is larger than OPT by a factor of $O(\frac{\ln n}{\ln \ln n})$ is at most $\frac{1}{n^3}$. Summing this probability over all edges, we see that with probability at least $1 - n^2 \frac{1}{n^3} = 1 - o(1)$ every edge has congestion at most $O(\frac{\ln n}{\ln \ln n} OPT)$.