CMPUT 675: Approximation Algorithms **Fall 2015**

Lecture 13 (Oct 13th, 2015): GAP, MST

Lecturer: Mohammad R. Salavatipour *Scribe: Chenyang Huang*

13.1 Generalized Assignment Problem(GAP)

Problem Description:
Suppose we are given a set 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):

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

Clearly the problem is NP-hard since even the feasibility (whether there is a solution satisfying all the constraints is NP-hard). 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.

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 $|U| \le |V|$. We say $M \le E$ 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 write the complete matching problem as the following integer program:

$$\begin{array}{rcll} \sum_{u:uv \in E} y_{uv} & \le & 1 & \forall v \in V \\ \sum_{v:uv \in E} y_{uv} & = & 1 & \forall u \in U \\ y_{uv} & \in & \{0,1\} & \end{array}$$

By relaxing the last constraint to $y_{uv} \ge 0$ 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;

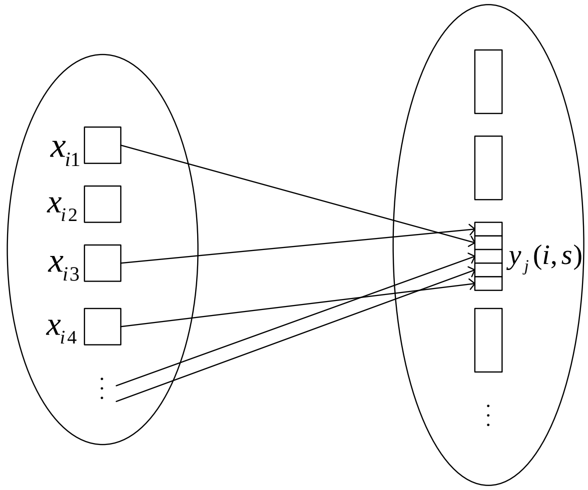

Figure 13.1: Blablabla

think of each as a unit size bin. We build a bipartite graph $B = (J \cup S, E)$ 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 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 $\sum_j \sum_s y_{j,(i,s)} p_{ij} = \sum_j p_{ij} x_{ij} \leq T$.

13.2 Minimum Spanning Tree

We know that the minimum spanning tree problem is solvable in polynomial time. Our goal (this and next lecture) is to prove that the spanning tree polytope is integral using an iterative argument. Let's start with the following LP relaxation of the problem.

$$\begin{array}{llccll} \min & \sum_{e \in E} c_e \cdot x_e & & & & \\ \text{s.t.} & \delta_x(S) & \geq & 1 & \forall S \subset V & (1) \\ & x_e & \geq & 0 & \forall e \in E & (2) \end{array}$$

It is easy to see that all the constraints of this LP are satisfied by any spanning tree. However this LP has integrality gap of almost 2.

Figure 13.2 shows such an example, a cycle of size n with each edge having factional value 1/2. So the cost of the LP will be $\frac{n}{2}$ but the integral solution will require $n-1$ edges. So the size of integrality gap is almost 2. We can model our problem in a way that we wouldn't encounter this problem. As an observation, it is obvious that in a graph G with a spanning tree T:

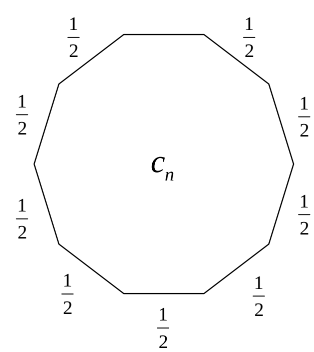

Figure 13.2: None integral example of LP(1)

$$\forall S \subseteq V : E(T) \cap E(S) \leq |S| - 1$$

where $E(T)$ is the edges of tree and $E(S)$ is the edges both end-points in S. As a matter of fact this condition is also sufficient for spanning trees. So we have:

$$\begin{array}{rl} \min & \sum w_e x_e \\ \text{s.t.} & x(E(S)) \leq |S| - 1, \forall S \subset V \\ & x(E(V)) = |V| - 1 \\ & x_e \geq 0. \end{array}$$

We call this LP_{MST}. Our goal is to prove that:

Theorem 2 *LP_{MST} is integral, i.e. every bfs of this LP is integral.*