IOE 691: Approximation Algorithms — Date: 03/15/2017

Lecture Notes: Generalized Assignment Problem

Instructor: Viswanath Nagarajan — Scribe: Hao Yuan

1 Generalized Assignment Problem

In this lecture, we consider a generalized assignment problem. There are m machines and n jobs. Each job must be assigned to some machine. If job $j \in [n]$ is assigned to machine $i \in [m]$, it requires processing time p_{ij} and incurs cost c_{ij}. Moreover, there is a common budget T such that the total processing time assigned to each machine has cannot exceed this budget. It is required to find an assignment in which no machine exceeds its budget and the total cost is minimized.

Let x_{ij} indicate the assignment of job j to machine i. A natural IP formulation is:

$$\begin{aligned} \min \quad & \textstyle\sum_{i\in[m],j\in[n]} c_{ij}x_{ij} & \\ s.t. \quad & \textstyle\sum_{j\in[n]} p_{ij}x_{ij} \le T, & \forall i \in [m] \\ & \textstyle\sum_{i\in[m]} x_{ij} = 1, & \forall j \in [n] \\ & x_{ij} \in \{0,1\}, & \forall i \in [m], j \in [n]. \end{aligned}$$

This is an NP-hard problem. Our goal is to find a bi-criteria approximation – achieving the optimal cost with a small processing time violation.

2 Iterative LP Based Algorithm

We consider an iterative LP based algorithm. In each iteration, we keep track of the current jobs $N \subseteq [n]$, current machines $M \subseteq [m]$, current set of active edges $E \subseteq [m] \times [n]$ and the current partial assignment $F \subseteq [m] \times [n]$. We solve the following LP in each such iteration.

$$\begin{aligned} \min \quad & \textstyle\sum_{i\in[m],j\in[n]} c_{ij}x_{ij} & \\ s.t. \quad & \textstyle\sum_{j\in[n]} p_{ij}x_{ij} \le T, & \forall i \in M \\ & \textstyle\sum_{i\in[m]} x_{ij} = 1, & \forall j \in N \\ & x_{ij} \in [0,1], & \forall (i,j) \in E \\ & x_{ij} = 1, & \forall (i,j) \in F \\ & x_{ij} = 0, & \forall (i,j) \in ([m] \times [n]) \setminus E \setminus F. \end{aligned}$$

Remark: In the LP above, x_{ij} is treated as a decision variable only if $(i,j) \in E$; if $(i,j) \notin E$ then x_{ij} is a fixed 0-1 value.

Now, we can describe our algorithm in detail.

Algorithm 1 LP based iterative algorithm

Set $N = [n], M = [m], E = [m] \times [n], F = \emptyset$
Repeat until $N = \emptyset$ **:**
  Find an extreme point solution $\mathbf{x}$ of problem $LP(N, M, E, F)$.
  If there exists $(i,j) \in E$ such that $x_{ij} = 0$**:** # Case 1
    assign $x_{ij} = 0$
    drop edge $E = E - \{(i,j)\}$
  If there exists $(i,j) \in E$ such that $x_{ij} = 1$**:** # Case 2
    assign $x_{ij} = 1$
    add edge to partial assignment $F = F \cup \{(i,j)\}$
    drop edge $E = E - \{(i,j)\}$
    drop assignment constraint $N = N - \{j\}$
  If there exists $i \in M$ such that $|\delta_E(i)| \leq 1$**:** # Case 3
    drop budget constraint $M = M - \{i\}$
  If there exists $i \in M$ such that $|\delta_E(i)| = 2$ and $\sum_{(i,j)\in\delta_E(i)} x_{ij} \geq 1$**:** # Case 4
    drop budget constraint $M = M - \{i\}$

3 Algorithm Analysis

Theorem 3.1 *[Bi-criteria Approximation] Our iterative LP based algorithm finds an integral assignment in polynomial time that violates each budget constraint by at most an additive $p_{max} = \max_{ij} p_{ij}$ and has cost at most OPT.*

The proof of Theorem 3.1 follows from the following two lemmas.

Lemma 3.1 *Assume that the algorithm terminates and produces assignment $\{\bar{x}_{ij}\}_{i\in[m],j\in[n]}$. Then,*

$$\begin{aligned} \sum_j p_{ij}\bar{x}_{ij} &\leq T + p_{\max}, \forall i \in [m] \\ \sum_{ij} c_{ij}\bar{x}_{ij} &\leq OPT. \end{aligned}$$

Proof: Note that we only remove a job j from N when it gets assigned in the partial solution F. So, if the algorithm terminates, the assignment $\{\bar{x}_{ij}\}_{i\in[m],j\in[n]}$ corresponding to F must have assigned each job (integrally) to some machine. Note that the first LP with $N = [n], M = [m], E = [m] \times [n], F = \emptyset$ is a relaxed version of the generalized assignment problem: so its optimal value is at most OPT. Moreover, in each iteration we only relax the current $LP(N, M, E, F)$ further, which shows that the optimal value of the final LP is also at most OPT. Now notice that the value of the final LP is just $\sum_{ij} c_{ij}\bar{x}_{ij}$. Hence $\sum_{ij} c_{ij}\bar{x}_{ij} \leq OPT$.

Now, we show that each budget constraint is violated by at most p_{max}. Consider any machine $i \in [m]$. At the termination time, let $load_{\mathsf{TRM}}(i)$ denote the total processing time on machine i. If at that time $i \in M$ (i.e., the budget constraint i is active), then $load_{\mathsf{TRM}}(i) = \sum_{j\in[n]} p_{ij}\bar{x}_{ij} \leq T$. Otherwise, i has been dropped at either case 3 or case 4 in some iteration. Consider the point when i is dropped: let $load_{\mathsf{DRP}}(i)$ denote the load on i due to jobs currently assigned to it in F (the partial solution), and let E denote the current active edges. If i is dropped at case 3, we have $|\delta_E(i)| \leq 1$ and $load_{\mathsf{DRP}}(i) \leq T$. They together imply $load_{\mathsf{TRM}}(i) \leq T + p_{max}$. If i is instead dropped at case 4, since $|\delta_E(i)| = 2$ let's assume $\delta_E(i) = \{j, k\}$. The LP budget constraint implies that:

$$load_{\mathsf{DRP}}(i) + p_{ij}x_{ij} + p_{ik}x_{ik} \leq T,$$

Also because $x_{ij} + x_{ik} \geq 1$, we then have, by simple algebra,

$$load_{\mathsf{TRM}}(i) \quad \leq \quad load_{\mathsf{DRP}}(i) + p_{ij} + p_{ik} \leq T + p_{max}.$$

■

Lemma 3.2 *The algorithm terminates in polynomial time.*

Proof: Since each case 1 – 4 makes our problem size strictly smaller, it is enough to show that, in each iteration, some case 1 – 4 must apply. We proceed by contradiction. We assume at some iteration that none of cases 1 – 4 apply to the extreme point solution $\mathbf{x}$ of $LP\left(N, M, E, F\right),$. So we must have

1. $0 < x_{ij} < 1$ for all $(i, j) \in E$,
2. $|\delta_E\left(i\right)| \geq 2$ for all $i \in M$,
3. $|\delta_E\left(j\right)| \geq 2$ for all $j \in N$ (otherwise, constraint $\sum_{i\in[m]} x_{ij} = 1$ forces some $x_{ij} = 1$ for $i \in \delta_E\left(j\right)$).

Note (2) and (3) implies that

$$|E| \geq \frac{1}{2}\left(2M + 2N\right) = M + N. \quad (\dagger)$$

On the other hand, note $\mathbf{x} \in \mathbb{R}^{|E|}$ as an extreme point solution has $|E|$ active/tight constrains (refer to Proposition 3.1 below), and (1) implies that constraints

$$\begin{array}{ll} \sum_{j\in[n]} p_{ij}x_{ij} \leq T & ,\forall i \in M \\ \sum_{i\in[m]} x_{ij} = 1 & ,\forall j \in N \end{array}$$

are the only possible $M + N$ active constrains, which shows that

$$|E| \leq M + N. \quad (*)$$

Therefore, from $(\dagger)$ and $(*)$, we get $|E| = M + N$, which, together with (2) and (3), forces that $|M| = |N|$ and $\delta_E\left(i\right) = \delta_E\left(j\right) = 2$ for all $i \in M, j \in N$. Now consider

$$\sum_{i\in M}\sum_{j:(ij)\in E} x_{ij} = \sum_{j\in N}\sum_{i:(ij)\in E} x_{ij} = |N| = |M|,$$

which implies $x(\delta_E\left(i_0\right)) \geq 1$ for some $i_0 \in M$. Combined with $|\delta_E\left(i_0\right)| = 2$, it follows that case 4 applies for i_0, which leads to contradiction. ■

Propersition 3.1 *[Extreme Point] For any linear program with bounded and non-empty feasible region $\{x : Ax \geq b\} \subseteq \mathbb{R}^n$, there exists an optimal solution $\bar{x}$ such that $\bar{A}\bar{x} = \bar{b}$ where $\bar{A} \in \mathbb{R}^{n\times n}$ is a rank-n sub-matrix of A and $\bar{b} \in \mathbb{R}^n$ is the corresponding sub-vector of b.*