Deterministic Rounding for Bipartite Matching and GAP[1]

- *Maximum Weight Bipartite Matching.* We are given a bipartite graph $G = (L \cup R, E)$ with weights w_{ij} on edges (i, j) The objective is to find a matching $M \subseteq E$ whose weight is maximized. This problem can be solved exactly. Below we see how a fractional solution of the natural LP relaxation can be rounded to an integral solution with the same cost, thus proving its integrality gap is 1.

$$\mathsf{lp}(G, w) := \text{maximize} \quad \sum_{(i,j)\in E} w_{ij}x_{ij} \tag{bipMWM-LP}$$

$$\sum_{j\in R} x_{ij} \leq 1, \qquad \forall i \in L \tag{1}$$

$$\sum_{i\in L} x_{ij} \leq 1, \qquad \forall j \in R \tag{2}$$

$$1 \geq x_{ij} \geq 0, \, \forall (i, j) \in E \tag{3}$$

Note that if $x_{ij} \in \{0, 1\}$, then the $x_{ij} = 1$ edges correspond to a matching. The above LP is a relaxation of the natural integer program capturing maximum weight matching.

- *Rounding by Rotation.* We now show a procedure which starts with *any* fractional matching x, and constructs a matching M with $w(M) \leftarrow \sum_{(i,j)\in E} w_{ij}x_{ij}$.

Theorem 1. For any bipartite graph G with weights w and any feasible solution x to $\mathsf{lp}(G, w)$, one can obtain a $\{0, 1\}$-solution x' with $\mathsf{lp}(x') \geq \mathsf{lp}(x)$.

Let $E_f(x) := \{(i, j) \in E : 0 < x_{ij} < 1\}$ be the *fractional* edges in the *support* of x. We now describe a procedure which takes x and converts it to x' such that two things occur: a) the number of edges in the corresponding $E_f(x')$ is strictly less than in $E_f(x)$, and b) $\sum_{i,j} w_{ij}x_{ij} \leq \sum_{i,j} w_{ij}x'_{ij}$. Continuing this till E_f becomes $\emptyset$, we end with a $\{0, 1\}$-solution x' with $\mathsf{lp}(x') \geq \mathsf{lp}(x)$, thus proving the theorem. See ROTATE below for precise definition.

Claim 1. Both $x^{(1)}$ and $x^{(2)}$ are feasible solutions to (bipMWM-LP), and $\mathsf{lp}(x') \geq \mathsf{lp}(x)$.

Proof. Let's prove $x^{(1)}$ is feasible and the proof for $x^{(2)}$ is analogous. If F is a cycle, then note that the "fractional load" on any vertex is unchanged in both x and $x^{(1)}$, and thus (1) and (2) are satisfied since they were satisfied in x. If F forms a path, then we need to concern ourselves with only end vertices of this path. Let $i \in L$ (or in R, doesn't matter) be be such a vertex and let (i, j) be the *unique* edge in $E_f(x)$. Since $x_{ij} > 0$, there cannot be any edge (i, j') with $x_{ij'} = 1$. In the end, by design $x^{(1)}_{ij} \leq 1$, and since all other edges incident on i have $x^{(1)}$ the same as x, we get that (1) is satisfied.

[1] *Lecture notes by Deeparnab Chakrabarty. Last modified : 7th Feb, 2022*
These have not gone through scrutiny and may contain errors. If you find any, or have any other comments, please email me at deeparnab@dartmouth.edu. Highly appreciated!

To see that $\mathsf{lp}(x') \geq \mathsf{lp}(x)$, note that the "increases" in $x^{(1)}$ and $x^{(2)}$ over x is precisely

$$\mathsf{lp}(x^{(1)}) - \mathsf{lp}(x) = \varepsilon_1 \left(\sum_{(i,j)\in M_2} w_{ij} - \sum_{(i,j)\in M_1} w_{ij} \right) ; \ \mathsf{lp}(x^{(2)}) - \mathsf{lp}(x) = \varepsilon_2 \left(\sum_{(i,j)\in M_1} w_{ij} - \sum_{(i,j)\in M_2} w_{ij} \right)$$

One of the terms in the RHS's above must be non-negative. □

procedure ROTATE($G = (L \cup R, E), x$):

▷ *Return a feasible solution x' with $|E_f(x')| < |E_f(x)|$ and $\mathsf{lp}(x') \geq \mathsf{lp}(x)$.*

Pick an arbitrary path or cycle in $G = (I, J, E_f)$. Call the edges picked F.

Decompose F into two matchings M_1 and M_2. ▷ *This is where bipartiteness is **crucially** used.*

Define

$$\varepsilon_1 := \min \left(\min_{(i,j)\in M_1} x_{ij}, \min_{(i,j)\in M_2} (1 - x_{ij}) \right)$$

$$\varepsilon_2 := \min \left(\min_{(i,j)\in M_2} x_{ij}, \min_{(i,j)\in M_1} (1 - x_{ij}) \right)$$

▷ *Note that both $\varepsilon_1, \varepsilon_2$ are strictly between 0 and 1.*

Define $x^{(1)}$ and $x^{(2)}$ as follows.

For each $(i, j) \in M_1$, $x^{(1)}_{ij} = x_{ij} - \varepsilon_1$, $x^{(2)}_{ij} = x_{ij} + \varepsilon_2$.

For each $(i, j) \in M_2$, $x^{(1)}_{ij} = x_{ij} + \varepsilon_1$, $x^{(2)}_{ij} = x_{ij} - \varepsilon_2$.

For all other edges $x^{(1)}_{ij} = x^{(2)}_{ij} = x_{ij}$

▷ *Note that $x^{(1)}$ and $x^{(2)}$ satisfy (3), $|E_f(x^{(1)})| < |E_f(x)|$ and $|E_f(x^{(2)})| < |E_f(x)|$.*

return $x^{(1)}$ or $x^{(2)}$ as x', whichever has higher lp-value.

- ***The Generalized Assignment Problem (GAP).*** In GAP, we are given m jobs J, and n machines I. The processing time of job j on machine i is p_{ij}, and if it is allocated on machine i, it generates a revenue of w_{ij} units. On the other hand, every machine i has a limit B_i of the maximum time it can run for. The goal is to find a feasible allocation of a subset of jobs to the machines such that the revenue generated is maximized. Formally, we need to find disjoint subsets $\mathcal{S} := (S_1, S_2, \ldots, S_n)$ of J such that so as to maximize $\mathsf{val}(\mathcal{S}) := \sum_{i=1}^{n} \sum_{j\in S_i} w_{ij}$ subject to $\sum_{j\in S_i} p_{ij} \leq B_i$ for all i. We let $\mathcal{I} := (I, J, w_{ij}, p_{ij})$ denote a GAP instance.

- *LP Relaxation.* The LP-relaxation looks very similar to the bipartite matching LP.

$$\mathsf{lp}(\mathcal{I}) := \text{maximize} \quad \sum_{i\in I, j\in J} w_{ij} x_{ij} \tag{GAP-LP}$$

$$\sum_{i\in I} x_{ij} \leq 1, \qquad \forall j \in J \tag{4}$$

$$\sum_{j\in J} p_{ij} x_{ij} \leq B_i, \qquad \forall i \in I \tag{5}$$

$$x_{ij} = 0, \qquad \forall j \in J, i \in I \ : \ p_{ij} > B_i \tag{6}$$

Indeed, if all B_i's and p_{ij}'s are 1, then it is precisely the same. x_{ij}'s indicate whether j is allocated to i. (5) asserts that the total processing times on any machine must be at most the machine's limit. (6) is the assertion that $p_{ij} > B_i$ implies j cannot be allocated to i. It's worth pointing out that (5) doesn't imply this and therefore (6) must explicitly be added to the LP.

- ***Rounding.*** The algorithm starts with a solution $\mathbf{x}$ of (GAP-LP). It then uses this solution to construct a maximum weight bipartite matching instance (G, w) *and* a fractional solution $\mathbf{y}$ to $\mathsf{lp}(G, w)$ such that (a) $\mathsf{lp}(\mathbf{y}) = \mathsf{lp}(\mathbf{x})$, that is, the LP-value of $\mathbf{y}$ in the bipartite matching is at least that of $\mathbf{x}$ in the GAP instance, (b) given an *integral* matching M in G of weight $w(M)$, can construct a feasible solution $\sigma : J \to I$ of value $\mathsf{alg}(\sigma) \geq w(M)/2$. Together with Theorem 1, we get a 2-approximation since one can obtain a matching M with $w(M) \geq \mathsf{lp}(\mathbf{y})$. We now give details.

  New Bipartite Graph. For every $i \in I$, evaluate $n_i := \left\lceil \sum_{j \in J} \mathbf{x}_{ij} \right\rceil$. Thus, n_i counts the "number" of jobs that are assigned by the LP to machine i. We now construct a bipartite graph $G = (N \cup J, E)$, where one part of the bipartition is J, the set of jobs. The other part N is formed by taking n_i copies of each machine i in I. Let N_i denote this set of n_i copies; thus, $N = \bigsqcup_{i \in I} N_i$.

  We next describe the edges in E. Fix a machine $i \in I$. We now describe the edges between N_i and J. Consider the job vertices in J in ***decreasing*** processing time order w.r.t. i. Indeed, for simplicity rename the jobs such that $p_{i1} \geq p_{i2} \geq \cdots \geq p_{im}$. Now, consider the fractions $\mathbf{x}_{i1}, \mathbf{x}_{i2}, \ldots, \mathbf{x}_{im}$ in this order. Define $j_0 = 1$, and let $j_1, j_2, \ldots, j_{n_i - 1}$ be the "boundary" items defined as follows : $\mathbf{x}_{i,1} + \cdots + \mathbf{x}_{i,j_1} \geq 1$, and $\mathbf{x}_{i,1} + \cdots + \mathbf{x}_{i,j_1 - 1} < 1$; $\mathbf{x}_{i,1} + \cdots + \mathbf{x}_{i,j_2} \geq 2$, and $\mathbf{x}_{i,1} + \cdots + \mathbf{x}_{i,j_1 - 2} < 2$; and so on. Formally, for each $1 \leq \ell \leq n_i - 1$, we find j_ℓ such that

$$\sum_{j=1}^{j_\ell} \mathbf{x}_{ij} \geq \ell; \quad \sum_{j=1}^{j_\ell - 1} \mathbf{x}_{ij} < \ell$$

  Recall there are n_i copies of the machine i in N_i. The ℓth copy, call it i_ℓ, has an edge to job vertices $j_{\ell-1}$ to j_ℓ (j_0 is the item 1). The n_ith copy has an edge to job vertices $j_{n_i - 1}$ to m. We repeat this for every $i \in I$ to get all the edges E. For the edge (i_ℓ, j) we give weight $w_{i_\ell, j} := w_{ij}$. See Figure 1 for an illustration.

  New Fractional Matching. Now we define $\mathbf{y}_{ij}$ for $i \in N$ and $j \in J$. Once again, fix a machine i and we now define $\mathbf{y}_{i_\ell, j}$ for $1 \leq \ell \leq n_i$ as follows. It is so defined such that for every copy i_ℓ, the total fractional weight incident on it is at most 1 (in fact, it'll be exactly 1 for all copies but the n_ith copy). The total fractional weight incident on item j is the same as that induced by $\mathbf{x}$. It should be clear how to do it given the way edges are defined above. See Figure 1 for an illustration. Formally, it is

$$\begin{aligned}
\mathbf{y}_{i_\ell, j} &= \mathbf{x}_{ij}, && \text{for } j_{\ell-1} < j < j_\ell \text{ or } j_{n_i - 1} < j \leq m \\
\mathbf{y}_{i_\ell, j_{\ell-1}} &= \mathbf{x}_{i, j_{\ell-1}} - \mathbf{y}_{i_{\ell-1}, j_{\ell-1}} && \text{(if } \ell = 1\text{, then the second term is } 0\text{).} \\
\mathbf{y}_{i_\ell, j_\ell} &= 1 - \sum_{j = j_{\ell-1}}^{j_\ell - 1} \mathbf{y}_{i_\ell, j} && \forall 1 \leq \ell \leq n_i - 1
\end{aligned}$$

  The following proceeds from the definition and will be crucial later.

  Claim 2. For any machine i, for any $1 \leq \ell \leq n_i - 1$, we have $\sum_{j = j_{\ell-1}}^{j_\ell} \mathbf{y}_{i_\ell j} = 1$.

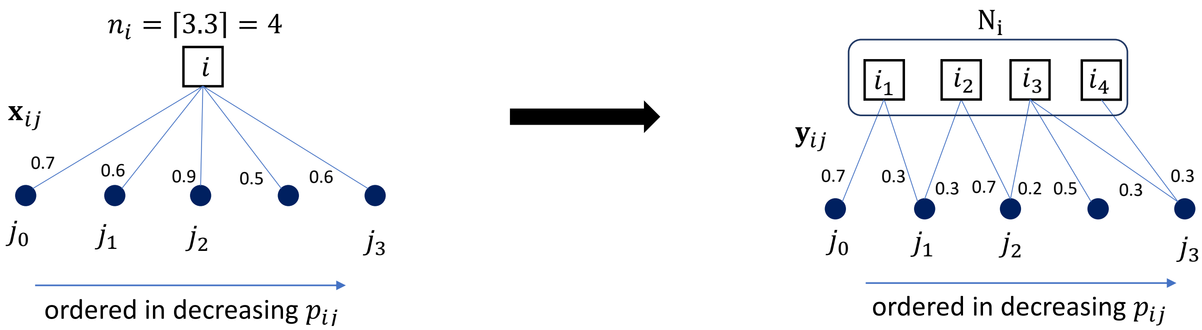

Figure 1: *Illustration of edges between N_i and J for one machine i, and also the description of $\mathbf{y}$-values on these edges. Note that the $\mathbf{y}$-load on jobs equal the $\mathbf{x}$-loads, and the $\mathbf{y}$-load on vertices i_1 to i_3 is 1, while on i_4 it is < 1.*

Proof. Since $\ell \leq n_i - 1$, $\mathbf{y}_{i_\ell j} > 0$ for $j_\ell \leq j \leq j_{\ell+1}$, and they sum to exactly 1. □

Claim 3. $\mathbf{y}$ is a valid feasible solution to (bipMWM-LP) with value $\mathsf{lp}(\mathbf{y}) = \mathsf{lp}(\mathbf{x})$.

Proof. The following can be inspected. For any job j and $i \in I$, we have $\sum_{i_\ell \in N_i} \mathbf{y}_{i_\ell,j} = \mathbf{x}_{ij}$. Thus, by design of $\mathbf{y}$, we have that $\mathbf{y}$ satisfies (2) where $R = J$. It also implies $\sum_{i_\ell \in N_i} \mathbf{w}_{i_\ell,j}\mathbf{y}_{i_\ell,j} = \mathbf{w}_{ij}\mathbf{x}_{ij}$ since $\mathbf{w}_{i_\ell,j} = \mathbf{w}_{ij}$. Thus, $\mathsf{lp}(\mathbf{y}) = \mathsf{lp}(\mathbf{x})$. By design, we have $\sum_{j \in J} \mathbf{y}_{i_\ell,j} \leq 1$ for all $i_\ell \in N_i$. □

Rounding and Pruning. From Theorem 1, we get that $G = (N \cup J, E, w)$ has a bipartite matching M with $w(M) \geq \mathsf{lp}(\mathbf{y})$. The GAP rounding ends by showing how to obtain $\sigma : J \to I$ using M. One idea is the following: for every $(i_\ell, j) \in M$ where $i_\ell \in N_i$, allocate job j to machine i. Let's call this allocation σ'.

Here is the main lemma which implies 2-approximation.

Lemma 1. For any machine i, $\sum_{j \in J : \sigma'(j) = i} p_{ij} \leq B_i + \Delta_i$, where $\Delta_i := \max_{j \in J} p_{ij}$.

Proof. This is where we use the fact that the items (for machine i) were ordered in decreasing order of processing times when we formed the graph. Let J_ℓ be the set of jobs from $j_{\ell-1}$ to j_ℓ, and let J_{n_i} be the jobs from j_ℓ to m. Note that the vertex i_ℓ can be matched to a vertex only from J_ℓ. Let σ'_ℓ be this job, and we let it be $\perp$ if i_ℓ was unmatched; in this case we define $p_{i_\ell,\sigma'_\ell} := 0$.

Since $\mathbf{x}$ is a feasible solution to (GAP-LP), we get

$$B_i \geq \sum_{j \in J} p_{ij}\mathbf{x}_{ij} = \sum_{\ell=1}^{n_i} \sum_{j \in J_\ell} p_{ij}\mathbf{y}_{i_\ell,j} \tag{7}$$

Now, since the p_{ij}'s are in *decreasing order*, for any $j \in J_\ell$ and $j' \in J_{\ell+1}$, we have $p_{ij} \geq p_{ij'}$. In particular, we have $p_{ij} \geq p_{i,\sigma'_{\ell+1}}$ for all $1 \leq \ell \leq n_i - 1$, $j \in J_\ell$. Therefore,

$$\text{For } 1 \leq \ell \leq n_i - 1, \quad \sum_{j \in J_\ell} p_{ij}\mathbf{y}_{i_\ell,j} \geq p_{i,\sigma'_{\ell+1}} \sum_{j \in J_\ell} \mathbf{y}_{i_\ell,j} \underbrace{=}_{\text{Claim 2}} p_{i,\sigma'_{\ell+1}}$$

Substituting in (7), we get

$$B_i \geq \sum_{\ell=1}^{n_i-1} p_{i,\sigma'_{\ell+1}} = \mathsf{load}_{\sigma'}(i) - p_{i,\sigma'_1}$$

Since $p_{i,\sigma'_1} \leq \Delta_i$, by definition, the lemma follows. $\square$

In particular, if we define $S'_i := \{j \in J : \sigma'(j) = i\}$ and let $\mathcal{S}' = (S'_1, \ldots, S'_n)$, then $\mathsf{val}(\mathcal{S}') = w(M) \geq \mathsf{lp}(\mathbf{x})$, but for the load on a machine i we can only say $\mathsf{load}_i \leq B_i + \Delta_i$.

To obtain a valid approximation algorithm, for every machine i, we partition S'_i into two : $S'_{i,1}$ which contains the job $j \in S'_i$ with the largest p_{ij} and the rest which we call $S'_{i,2}$. We define S_i to be the one among these with the *largest* weight. That is, $S_i = S'_{i,1}$ if $\sum_{j \in S'_{i,1}} w_{ij} \geq \sum_{j \in S'_{i,2}} w_{ij}$, and $S_i = S'_{i,2}$ otherwise. Note that by design (a) $\sum_{j \in S_i} w_{ij} \geq \frac{1}{2} \cdot \sum_{j \in S'_i} w_{ij}$, and (b) $\sum_{j \in S_i} p_{ij} \leq B_i$. The reason for (b) is that $\sum_{j \in S'_{i,1}} p_{ij} \leq B_i$ since j is a single job where $\mathbf{x}_{ij} > 0$, and thus p_{ij} for this job is $\leq B_i$, and $\sum_{j \in S'_{i,2}} p_{ij} \leq B_i$ due to Lemma 1. Therefore, at the end we end with a feasible allocation $\mathcal{S}$ with total value $\mathsf{val}(\mathcal{S}) \geq \frac{w(M)}{2} \geq \mathsf{lp}(\mathbf{x})/2$.

To summarize,

procedure GAP ROUNDING($\mathcal{I} = (I \cup J, p_{ij}, w_{ij})$):
  Solve (GAP-LP) to get $\mathbf{x}$.
  Form bipartite graph $G = (N \cup J, E, w)$ with fractional solution $\mathbf{y}$. ▷ $\mathsf{lp}(\mathbf{y}) = \mathsf{lp}(\mathbf{x})$.
  Find matching M in G with $w(M) \geq \mathsf{lp}(\mathbf{y}) \geq \mathsf{lp}(\mathbf{x})$.
  Find tentative assignment σ' of *all* $j \in J$ with $\mathsf{val}(\sigma) \geq \mathsf{lp}(\mathbf{x})$.
  For each $i \in I$ either assign job with max processing time among jobs allocated to it by σ', or the remaining, whichever gives more value.

Theorem 2. GAP ROUNDING is a $\frac{1}{2}$-approximation for the Generalized Assignment Problem.

Remark: *Recall the load balancing problem from a previous lecture : n jobs with processing times p_j, m machines, goal was to find an assignment which minimizes the maximum load on a machine. This problem has a PTAS, and indeed, an EPTAS. A generalization of the problem, called* makespan minimization on unrelated machines *is the same input as above except job i takes a different p_{ij} time on machine i, and the p_{ij}'s for different i's may not be related. This is a much harder problem, and in fact, there can be no 1.499-approximation algorithm unless $P = NP$. On the other hand, note that the algorithm described here gives a 2-approximation. In particular, in (GAP-LP) replace B_i in (5) by T, and find (using binary search, say) the smallest T for which the LP returns a feasible solution. Lemma 1 shows that if the LP returns a feasible solution, then we can assign all jobs with maximum load $\leq 2T$, since $\Delta_i \leq T$ due to (6).*

Notes

The algorithm for GAP described here is from the paper [2] by Shmoys and Tardos. This generalized a result from an earlier paper [5] by Lenstra, Shmoys, and Tardos which gave the first 2-approximation for the unrelated makespan minimization problem. This later paper also contained the $\frac{3}{2} - \varepsilon$ hardness, and closing this gap has resisted effort and is an outstanding open problem. For GAP, the version we study, better approximation algorithms are possible. There is an $(1 - \frac{1}{e})$-approximation algorithm described in the paper [4] by Fleischer, Goemans, Mirrokni, and Sviridenko, and this factor was improved to $(1 - \frac{1}{e} + \varepsilon_0)$ for a (very) small constant ε_0 in the paper [3] by Feige and Vondrák. The best known hardness of approximation for GAP is $\frac{10}{11}$ which can be found in the paper [1] by Chakrabarty and Goel.

References

[1] D. Chakrabarty and G. Goel. On the approximability of budgeted allocations and improved lower bounds for submodular welfare maximization and GAP. *SIAM Journal on Computing (SICOMP)*, 39(6):2189–2211, 2010.

[2] David B. Shmoys and Eva Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming*, 62:461–474, 1993.

[3] U. Feige and J. Vondrák. Approximation algorithms for allocation problems: Improving the factor of $1 - \frac{1}{e}$. *Proceedings of FOCS*, 2006.

[4] L. Fleischer, M. X. Goemans, V. Mirrokini, and M. Sviridenko. Tight Approximation Algorithms for Maximum General Assignment Problems,. *Proc., ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2006.

[5] J. K. Lenstra, D. B. Shmoys, and E. Tardos. Approximation Algorithms for Scheduling Unrelated Parallel Machines. *Math. Programming*, 46:259–271, 1990.