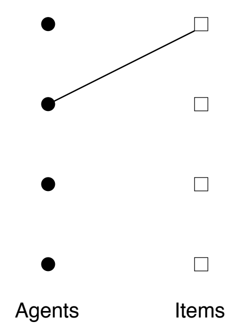
CS 369P: Polyhedral techniques in combinatorial optimization
Instructor: Jan Vondrák
Lecture date: November 4, 2010 **Scribe: David Tobin**

1 Maximum Budgeted Allocation



Given: n agents and m items. For each agent i and item j, b_{ij} is the amount agent i is willing to pay for item j. For each agent i, B_i is the budget of agent i.

Goal: Allocate a set of items S_i to each agent i (with $S_i \cap S_j = \emptyset \ \forall i, j$) to maximize

$$\sum_{i=1}^{n} \min \left\{ B_i, \sum_{j \in S_i} b_{ij} \right\}$$

The maximum budgeted allocation problem is NP-hard because the napsack problem can be reduced to maximum budgeted allocation with two agents. Hence we cannot expect an efficient algorithm that solves maximum budgeted allocation exactly. The goal is to find an approximation algorithm that produces a solution of value $\geq \alpha \cdot OPT$ for a fixed $\alpha \in [0, 1]$.

Theorem 1 (Chakrabarty, Goel) *There exists an effficient algorithm which finds an allocation of value* $\geq \frac{3}{4} OPT$.

It is not known if this approximation is optimal. It has been proven that it is NP-hard to achieve an approximation factor of $(\frac{15}{16} + \epsilon)$ for any fixed $\epsilon > 0$.

1.1 LP for Maximum Budgeted Allocation

Given variables $x_{ij} \in [0, 1]$, maximize

$$\sum_{i=1}^{n} \min \left\{ B_i, \sum_{j=1}^{m} b_{ij} x_{ij} \right\}$$

such that

$$\forall j; \sum_{i=1}^{n} x_{ij} \leq 1$$
$$\forall i, j; x_{ij} \geq 0$$

This objective function is not a linear constraint. It can be rewritten as $\sum_{i=1}^{n} y_i$ with the constraints $y_i \leq B_i$ and $y_i \leq \sum_j b_{ij} x_{ij}$ to make this a valid LP.

Given any solution to the above LP, if there is some agent i for which $\sum b_{ij} x_{ij} > B_i$, then we can decrease the x_{ij} so that $\sum b_{ij} x_{ij} \leq B_i$ without changing the value of the solution. So without loss of geneneralty we can assume that $\sum b_{ij} x_{ij} \leq B_i$. This gives us the following LP, which we will use for the remainder of this section:

$$\max \sum_{i=1}^{n} \sum_{j=1}^{m} b_{ij} x_{ij}$$
$$\forall i; \sum_{j} b_{ij} x_{ij} \leq B_i$$
$$\forall j; \sum_{i} x_{ij} \leq 1$$
$$\forall i, j; x_{ij} \geq 0$$

Lemma 2 *Let x be an optimal (vertex) solution to the LP, which in addition minimizes $\sum_{i,j} x_{ij}$ (among optimal solutions), then $G_x = (V, supp(\mathbf{x}))$ is a forest.*

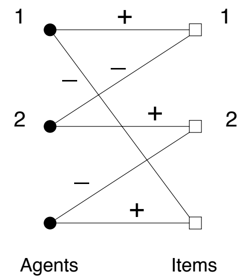
To find such a solution:

1. Solve the LP to find the optimum value.
2. Add a constraint that the value is equal to the optimum value, then change the objective function to $\sum x_{ij}$ and re-solve the LP.

Proof: Assume for the sake of contradiction that G_x has a cycle C. Decompose the cycle into matchings M_1, M_2 such that $C = M_1 \cup M_2$. Without loss of generality, assume that

$$\prod_{(i,j) \in M_1} b_{ij} \geq \prod_{(i,j) \in M_2} b_{ij}$$

Number the agents $1, \ldots, n$ and the items $1, \ldots, n$ in the order that the cycle traverses them. Our goal will be to assign ε_{ij} values to each edge of the cycle, alternating in sign, such that we can adjust the solution by those values.



Case 1: $\prod_{(i,j)\in M_1} b_{ij} = \prod_{(i,j)\in M_2} b_{ij}$

Choose $\varepsilon_1, \ldots, \varepsilon_k$ such that $b_{i(i-1)}\varepsilon_{i-1} = b_{i(i-1)}\varepsilon_{i-1}$.

In this case the ε_i values are consistent around the cycle.

$$\begin{aligned}
\varepsilon_2 &= \varepsilon_1 \cdot \frac{b_{21}}{b_{22}} \\
\varepsilon_3 &= \varepsilon_2 \cdot \frac{b_{32}}{b_{33}} = \varepsilon_1 \cdot \frac{b_{32}b_{21}}{b_{33}b_{22}} \\
&\cdots \\
\varepsilon_k &= \varepsilon_{k-1} \cdot \frac{b_{k(k-1)}}{b_{kk}} = \varepsilon_1 \cdot \frac{b_{k(k-1)}\ldots b_{21}}{b_{kk}\ldots b_{22}} \\
\varepsilon_k b_{1k} &= \varepsilon_1 b_{11} \cdot \frac{\prod_{(i,j)\in M_2} b_{ij}}{\prod_{(i,j)\in M_1} b_{ij}} = \varepsilon_1 b_{11}
\end{aligned}$$

With these ε_j values, define a vector ε where the values for the edges adjacent to item j are ε_j and $-\varepsilon_j$ (with signs as shown in the diagram above). We can move $\mathbf{x} \pm \varepsilon$ without changing any of the agent or item constraints, and if ε is chosen small enough, none of the edge constraints will be violated as well. This contradicts the assumption that x was a vertex.

Case 2: $\prod_{(i,j)\in M_1} b_{ij} < \prod_{(i,j)\in M_2} b_{ij}$

If we use the same procedure for choosing the ε values as in the previous part, we will end up with $\varepsilon_k b_{1k} > \varepsilon_1 b_{11}$.

Instead, define ε_{ij} such that

$$\begin{aligned}
&\varepsilon_{ij} > 0 \text{ for } (i,j) \in M_1 \\
&\varepsilon_{ij} < 0 \text{ for } (i,j) \in M_2 \\
&\varepsilon_{ij}b_{ij} + \varepsilon_{ij'}b_{ij'} = 0 \text{ for each agent } i \\
&\varepsilon_{ij} + \varepsilon_{i'j} \leq 0 \text{ for each item } j
\end{aligned}$$

If equality held for each $\varepsilon_{ij} + \varepsilon i'j \leq 0$ constraint, we would get exactly the definition of the ε values used in the previous case. But because $\prod_{(i,j)\in M_1} b_{ij} < \prod_{(i,j)\in M_2} b_{ij}$, there must be some item j for which $\varepsilon_{ij} + \varepsilon_{i'j} < 0$.

Then $\mathbf{x} + \varepsilon$ preserves the budget constraints for each item i and does not violate any of the item constraints, since $\sum_i x_{ij}$ can only decrease for each item j. $\mathbf{x} + \varepsilon$ decreases $\sum_{i,j} x_{i,j}$, so this contradicts the choice of x as the optimal solution with the smallest $\sum_{i,j} x_{ij}$.

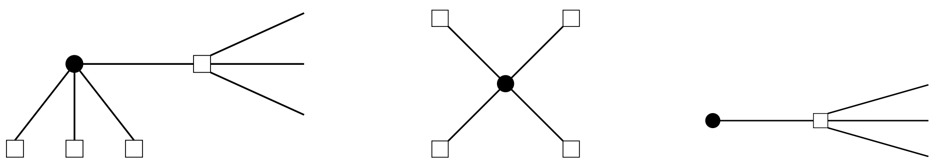
□

Definition 3 *A **leaf item** is an item that is a leaf of some tree in G_x.*

Definition 4 *An agent is called a **quasi-leaf** if:*

a) All neighbors are leaf items, with the exception of at most one neighbor.

b) At least one neighbor is a leaf.



(a) quasi-leaf (b) quasi-leaf (c) not a quasi-leaf (d) quasi-leaf

Figure 1: Examples of quasi-leaf agents

Lemma 5 *Every component of G_x has at most one agent which is not tight ($\sum_j b_{ij}x_{ij} < B_i$).*

Proof: By the Rank Lemma for vertices of a polytope, a vertex x has $|E|$ tight constraints. In fact, for a tree T_i in G_x with edges E_i, there must be at $|E_i|$ tight constraints for that tree. Otherwise thre is a direction $\mathbf{d} \in \mathbb{R}^{E_i}$ orthogonal to all the tight constraints. Then we could move $x \pm \mathbf{d}$, and x would not be a vertex. The constraints on T_i cannot be edge constraints because G_x is the graph of the support of x. There are $|E_i| + 1$ vertices in the tree and at least $|E_i|$ tight constraints, so at most 1 vertex can be non-tight. □

Lemma 6 *Every tree of size ≥ 2 in G_x has a quasi-leaf agent and either*

a) it's the only agent in the tree, or

b) the agent is tight.

Proof: Let T be a tree in G_x, and let T' be the tree resulting from removing all of the item leaves of T. Among the leaves of T' (all of which are agents), at most one can be non-tight by Lemma 5. So either there is only one agent or there are multiple agents and one is a tight agent that is a leaf in T'. If there is only one agent, we are done, so let i be a tight agent that is a leaf in T'. Let (i, j) be the edge adjacent to i in T'. We can assume without loss of generality that $b_{ij} \leq B_i$. Additionally, $x_{ij} < 1$ because $\sum_i x_{ij} \leq 1$ and item j has non-zero edges to other agents. Then $b_{ij}x_{ij} < B_i$. So in order for agent i to be tight, agent i must be adjacent to other (leaf) items in T. Thus agent i is a tight quasi-leaf agent. □

Algorithm. Our algorithm is the following.

1. Solve the LP and find a forest solution $\mathbf{x}$. Delete all edges with $x_{ij} = 0$ and call the remaining graph G_x.

2. If G_x contains any edges, find a quasi-leaf agent i as guaranteed by Lemma 6. Let L denote the leaf items adjacent to i. Assign all items in L to i and remove L from G_x. If there are no other items adjacent to i, remove also the agent i. Otherwise, there is exactly one additional item, call it j. Then modify the budget of i and its bid on j as follows: $B'_i = b'_{ij} = \frac{4}{3} b_{ij} x_{ij}$. Return to the first step.

3. If G_x does not contain any edges, return the current allocation.

Note that in case we have a quasi-leaf agent which has a non-leaf item, Lemma 6 guarantees that the agent is tight. In this case we modify its bid on the remaining item in a non-intuitive way - we call such an agent a "lying agent" in the following, because his bid does not correspond to a real value. Observe that next time we visit this agent, it must be the only agent in its component (otherwise we would not select it as a quasi-leaf) and then we will allocate the remaining item to him.

Analysis. What remains is a careful analysis of the value that we accumulate throughout the process. Let us denote by $Val(LP)$ the value of the LP at some point. Let S denote the partial solution at that point and $\tilde{Val}(S)$ denote its value in the following sense: for each agent who has been removed from the LP, we count the value of all its items, or its budget B_i, whichever is smaller. For a lying agent who still remains in the LP, we count $\sum_{\ell \in L} b_{i\ell} x_{i\ell}$ where L is the set of leaf items and x_{ij} their fractional values at the moment we allocated them (i.e., we count only the "fractional value" of these items; observe also that we have $\tilde{Val}(S) \leq Val(S)$ where $Val(S)$ is the actual value of the current allocation). If we count the value of a lying agent this way, and j is his remaining item, note that he still has $b_{ij} x_{ij}$ of budget remaining, because the solution at that moment satisfied the constraint $b_{ij} x_{ij} + \sum_{\ell \in L} b_{i\ell} x_{i\ell} \leq B_i$. Recall that we modify his budget in the LP to $B'_i = \frac{4}{3} b_{ij} x_{ij}$, i.e. possibly a factor of $4/3$ times the real remaining budget. This trick is essential in the analysis.

We claim that the following potential function never decreases throughout the algorithm:

$$\Phi = \frac{4}{3} \tilde{Val}(S) + Val(LP).$$

If we prove this, we are done, because at the beginning we have $\Phi = Val(LP) \geq OPT$, and at the end there are no variables left in the LP and hence $\Phi = \frac{4}{3} \tilde{Val}(S) \leq \frac{4}{3} Val(S)$ where $Val(S)$ is the actual value of our allocation.

So let us analyze each step of the algorithm. If we take a lying agent, there is only 1 item left and its contribution to the LP is at most $B'_i = \frac{4}{3} b_{ij} x_{ij}$, where b_{ij} and x_{ij} were the real bid and the fractional value at the time this agent started lying. Recall that the actual remaining budget of this agent is at least $b_{ij} x_{ij}$. Therefore, we gain at least $b_{ij} x_{ij} = \frac{3}{4} B'_i$ while the LP loses at most B'_i. So Φ does not decrease.

If we take a non-lying agent, and all his items are leaves, then the current solution gains at least $\sum_{\ell \in L} b_{i\ell} x_{i\ell}$ and the LP loses exactly this much, so we are also fine here.

Finally, we if take a non-lying agent who has a non-leaf item, we proceed as follows. $\tilde{Val}(S)$ gains at least $\sum_{\ell \in L} b_{i\ell} x_{i\ell} = B_i - b_{ij} x_{ij}$ where j is the remaining item, using the fact that this was a tight agent. Meanwhile, the LP loses

$$\sum_{\ell \in L} b_{i\ell} x_{i\ell} + (b_{ij} - b'_{ij}) x_{ij} = B_i - b'_{ij} x_{ij} = B_i - \frac{4}{3} b_{ij} x_{ij}^2.$$

Hence, the potential change can be written as follows:

$$\Delta\Phi = \frac{4}{3}(B_i - b_{ij} x_{ij}) - (B_i - \frac{4}{3} b_{ij} x_{ij}^2) = \frac{1}{3} B_i - \frac{4}{3} b_{ij} x_{ij}(1 - x_{ij}) \geq \frac{1}{3} B_i - \frac{1}{3} b_{ij} \geq 0$$

using the fact that $x_{ij}(1 - x_{ij}) \leq \frac{1}{4}$ and the bids b_{ij} are assumed to satisfy $b_{ij} \leq B_i$. This finishes the proof of Theorem 1.