Approximation Algorithms via LP rounding

Recipe: 1. Encode problem as integer LP
2. Relax to fractional LP and use an LP solver to obtain an optimal fractional solution x^*
3. ROUND fractional solution to an integer solution x^*_{INT}

$$\min c^T x \quad Ax \geq b \quad x \geq 0$$

1. $X^* \leq X_{OPT}$
2. $X^*_{INT} \leq \alpha \; x^*$

$\Rightarrow$ X^*_{INT} is an α-approximation

Examples

1. Vertex cover

$$\min \sum_{v \in V} x(v)$$
$$\text{s.t.} \quad x(u) + x(v) \geq 1 \quad \forall (u,v) \in E$$
$$x(v) \geq 0 \quad \forall v \in V$$

$$x^*_{INT}(v) = \begin{cases} 1 & \text{if } x^*(v) \geq \frac{1}{2} \\ 0 & \text{o.w.} \end{cases}$$

Approx factor $= 2$

2. Set cover

$$\min \sum_{s \in \mathcal{S}} c(s) x(s)$$
$$\sum_{s : e \in s} x(s) \geq 1 \quad \forall e \in U$$
$$x(s) \geq 0 \quad \forall s \in \mathcal{S}$$

$$x^*_{INT}(s) = \begin{cases} 1 & \text{w/p } x^*(s) \\ 0 & \text{o.w.} \end{cases}$$

$$\mathbb{E}\left[c(s)\, x^*_{INT}(s)\right] = c(s)\, x^*(s)$$

Too good! Exact algorithm!

Problem: $\Pr\left[\sum_{s : e \in s} x(s) \geq 1\right] \geq 1 - \prod_{s : e \in s} \left(1 - x(s)\right) \geq 1 - \frac{1}{e}$.

i.e. in expectation, a constant fraction of elements are not covered by x^*_{INT}.

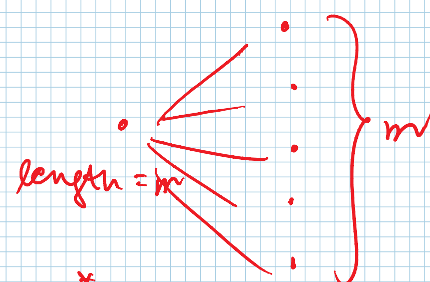
Solution: Repeat $\log n$ times (or boost rounding probability by $\log n$).

⇒ Monte Carlo algorithm with approx factor $\log n$.

How do we convert to Las Vegas? For each element that is not covered, add cheapest set covering the element.

Load balancing on Unrelated Machines [Lenstra-Shmoys-Tardös]

$$\min t$$
$$\text{s.t.} \sum_{j \in J} x_{ij} p_{ij} \leq t \quad \forall i \in M$$
$$\sum_{i \in M} x_{ij} \geq 1 \quad \forall j \in J$$
$$x_{ij} \geq 0$$



$t^* = 1$; Integer OPT $= m$

Defn: The integrality gap of an LP is the worst-case ratio (over all instances) of the integer optimal to the fractional optimal objective.

The above example shows that this LP has an integrality gap of m.

Stronger LP: Let t_{OPT} be optimal makespan (can be guessed up to $(1+\varepsilon)$ error). Define

$$\tilde{p}_{ij} = \begin{cases} p_{ij} & \text{if } p_{ij} \leq t_{OPT} \\ \infty & \text{if } p_{ij} > t_{OPT} \end{cases}$$

New LP: Check feasibility of

$$\sum_{i \in M} x_{ij} \geq 1 \quad \forall j \in J$$

guess for t_{OPT}

$$\sum_{j \in J} x_{ij} \tilde{p}_{ij} \leq \tilde{t}_{OPT} \quad \forall i \in M$$

If infeasible, update $\tilde{t}_{OPT}$ to $(1+\varepsilon)\tilde{t}_{OPT}$.

If feasible, we need to round the solution.

Rounding: In a basic feasible solution, at most $(\underset{\#\text{ of machines}}{m} + \underset{\#\text{ of jobs}}{n})$ variables

Rounding: In a basic feasible solution, at most $(m+n)$ variables x_{ij} are non-zero. If $n \geq m$, at least $n-m$ jobs are assigned integrally. Let H be the graph of edges with $0 < x_{ij} < 1$. This is an $n' \times m$ graph containing $\leq n'+m$ edges, but each job has degree ≥ 2 and $n' \leq m$.

Lemma: H is a pseudo-tree, i.e. each connected component on k vertices has at most k edges.

Proof: x^*, restricted to the jobs and machines in a connected component, must be a BFS. If not, it can be written as a convex combination of feasible points which can be extended to the full LP.

Lemma: A pseudo-tree in a bipartite graph has a perfect matching.

Proof: Leaves are machines since each job has degree ≥ 2. Match leaves to their unique neighbors and recurse. If there are no leaves, then the pseudo-tree is a cycle, which being bipartite, must be even and hence contain a perfect matching // Thus, each machine gets at most one job more than those it got integrally in the fractional solution. By parametric pruning, this is a 2-approximation.