Chapter 5: Random Sampling and Randomized Rounding of Linear Programs

(cp. Williamson & Shmoys, Chapter 5)

Randomized Approximation Algorithm

Definition 5.1.

A randomized α-approximation algorithm is a polynomial-time randomized algorithm which always finds a feasible solution whose *expected* value is bounded by $\alpha \cdot \mathrm{OPT}$.

Remarks

- Often, a randomized α-approximation algorithm can be *derandomized*, i. e., turned into a deterministic α-approximation algorithm.
- It is usually simpler to state and analyze the randomized algorithm.
- Sometimes, the only known way of analyzing a deterministic approximation algorithm is to analyze a randomized version.
- Sometimes one can show that the performance guarantee of a randomized algorithm holds with high probability.

Maximum Satisfiability Problem (MAX SAT)

Given: Boolean variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$ with weights $w_1, \ldots, w_m \in \mathbb{R}_{\geq 0}$.

(Clause is disjunction of Boolean variables or negations, e. g., $x_1 \vee \overline{x_2} \vee x_3$)

Task: Find a truth assignment to $x_1, \ldots, x_n$.

Objective: Maximize the total weight of satisfied clauses.

Example: $(x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2}) \wedge (x_2 \vee x_3) \wedge (\overline{x_3})$

Remarks:

- A variable x_i or its negation $\overline{x_i}$ is a literal.
- The number of literals ℓ_j in clause C_j is its size or length.
- If $\ell_j = 1$, then C_j is a unit clause.
- W.l.o.g. no literal is repeated in a clause and clauses are distinct.
- W.l.o.g. at most one of x_i and $\overline{x_i}$ appears in a clause.

Randomized Truth Assignment

Theorem 5.2.

a. Setting each x_i to true independently with probability $1/2$ gives a randomized $1/2$-approximation algorithm for MAX SAT.

b. If $\ell_j \geq k$ for all $j = 1, \ldots, m$, then the above algorithm is a randomized $(1 - 1/2^k)$-approximation algorithm.

Proof:. . . □

Maximum Exactly 3SAT (MAX E 3SAT): The special case of MAX SAT where $\ell_j = 3$ for all $j = 1, \ldots, m$ is called MAX E 3SAT.

We state the following theorem without proof.

Theorem 5.3.

Unless $P = NP$, there is no $(7/8 + \varepsilon)$-approximation algorithm for MAX E 3SAT for any constant $\varepsilon > 0$.

Maximum Cut Problem (MAX CUT)

Given: Undirected Graph $G = (V, E)$ with edge weights $w_e \geq 0$, $e \in E$.

Task: Find $S \subset V$ maximizing $\sum_{e \in \delta(S)} w_e$.

Theorem 5.4.

Placing each node $v \in V$ into S independently at random with probability $1/2$ gives a randomized $1/2$-approximation algorithm for MAX CUT.

Proof:... □

Derandomization: Method of Conditional Expectations

Basic Idea:

- Consider random decisions sequentially one after another.
- Take next decision deterministically optimizing the expected solution value assuming that all remaining decisions are taken randomly.

Example: Derandomized version of randomized MAX SAT algorithm

Let W denote the total weight of satisfied clauses in final solution.

1. for $i = 1$ to n
2. if $\mathrm{E}[W \mid x_1 = b_1, \ldots, x_{i-1} = b_{i-1}, x_i = \text{true}] \geq \mathrm{E}[W \mid x_1 = b_1, \ldots, x_{i-1} = b_{i-1}, x_i = \text{false}]$
3. then set $b_i := \text{true}$;
4. else set $b_i := \mathit{false}$;
5. return x:=b;

Method of Conditional Expectations: Analysis

Theorem 5.5.

The value of the solution computed by the deterministic MAX SAT algorithm is at least the expected value of the randomized solution.

Proof:... □

Remarks.

- The crucial step of the derandomized algorithm is to compute the conditional expectations.
- Notice that $\mathrm{E}[W \mid x_1 = b_1, \ldots, x_i = b_i]$

$$= \sum_{j=1}^{m} w_j \cdot \Pr[C_j = \text{true} \mid x_1 = b_1, \ldots, x_i = b_i]$$

and $\Pr[C_j = \text{true} \mid x_1 = b_1, \ldots, x_i = b_i]$

$$= \begin{cases} 1 & \text{if } x_1 = b_1, \ldots, x_i = b_i \text{ satisfies } C_j, \\ 1 - 1/2^k & \text{else,} \end{cases}$$

where k is the number of remaining literals in clause C_j.

Flipping Biased Coins

We first restrict to MAX SAT instances with no negated unit clause.

Lemma 5.6.

If each x_i is independently set to true with probability $p > 1/2$, then the probability that a clause is satisfied is at least $\min\{p, 1 - p^2\}$.

Proof:... □

Theorem 5.7.

For $1/2 < p \leq 1$ this gives a randomized $\min\{p, 1 - p^2\}$-approximation algorithm for MAX SAT. □

Notice: For $p = (\sqrt{5} - 1)/2$ we get $\min\{p, 1 - p^2\} = (\sqrt{5} - 1)/2 \approx 0.618$.

Remark:

The initial assumption on the non-existence of negated unit clauses is not needed to get the randomized $(\sqrt{5} - 1)/2$-approximation algorithm!

Integer Programming Formulation for MAX SAT

For $j = 1, \ldots, m$ let $P_j := \{i \mid \text{literal } x_i \text{ occurs in } C_j\}$

and $N_j := \{i \mid \text{literal } \overline{x_i} \text{ occurs in } C_j\}$.

That is, $$C_j = \bigvee_{i \in P_j} x_i \vee \bigvee_{i \in N_j} \overline{x_i}.$$

IP formulation:

$$\begin{aligned} \max \quad & \sum_{j=1}^{m} w_j \cdot z_j \\ \text{s.t.} \quad & \sum_{i \in P_j} y_i + \sum_{i \in N_j} (1 - y_i) \geq z_j && \text{for all } j = 1, \ldots, m, \\ & y_i \in \{0, 1\} && \text{for all } i = 1, \ldots, n, \\ & 0 \leq z_j \leq 1 && \text{for all } j = 1, \ldots, m. \end{aligned}$$

LP relaxation: Replace $y_i \in \{0, 1\}$ with $0 \leq y_i \leq 1$ for all $i = 1, \ldots, n$.

Randomized Rounding

1. compute an optimal solution (y^*, z^*) to the LP relaxation;
2. for $i = 1$ to n do
3. set x_i to true independently at random with probability y_i^*;

Theorem 5.8.

Randomized rounding gives a randomized $(1 - 1/e)$-approximation algorithm for MAX SAT.

Proof:... □

Remark.

Algorithm can be derandomized by method of conditional expectations.

Choosing the Better of Two Solutions

Theorem 5.9.

Running either the unbiased randomized $1/2$-approximation algorithm or the randomized rounding algorithm, both with probability $1/2$, yields a randomized $3/4$-approximation algorithm.

Proof:... □

Derandomizing the initial coin flip yields:

Corollary 5.10.

Running both algorithms and choosing the better of the two solutions is a randomized $3/4$-approximation algorithm. □

Non-linear Randomized Rounding

Consider a function $f : [0,1] \to [0,1]$.

1. compute an optimal solution (y^*, z^*) to the LP relaxation;
2. for $i = 1$ to n do
3. set x_i to true independently at random with probability $f(y_i^*)$;

Theorem 5.11.

Let $f : [0,1] \to [0,1]$ with $1 - 4^{-x} \leq f(x) \leq 4^{x-1}$ for all $x \in [0,1]$. Then non-linear randomized rounding with function f is a randomized $3/4$-approximation algorithm.

Proof:... □

Remark:

- The integrality gap of the LP relaxation for MAX SAT is $3/4$.
- Thus, $3/4$ is best performance ratio one can prove based on the LP.

Randomized Algorithm for Prize-Collecting Steiner Trees

Idea:

- Obtain randomized variant of deterministic LP rounding algorithm from Chapter 4 by choosing α randomly.
- For some fixed $\gamma > 0$ choose α uniformly at random from $[\gamma, 1]$.
- That is, choose α from $[\gamma, 1]$ with constant density function $1/(1-\gamma)$.

Lemma 5.12.

The tree T returned by the randomized algorithm has expected cost

$$\mathrm{E}\left[\sum_{e \in E(T)} c_e\right] \leq \frac{2}{1-\gamma} \ln \frac{1}{\gamma} \sum_{e \in E} c_e \cdot x_e^* \ .$$

Proof:. . . □

Randomized Algorithm for Prize-Collecting Steiner Trees

Lemma 5.13.

The expected penalty costs are

$$\mathrm{E}\left[\sum_{i \in V \setminus V(T)} \pi_i\right] \leq \frac{1}{1-\gamma} \sum_{i \in V} \pi_i \cdot (1 - y_i^*) \ .$$

Proof:. . . □

Theorem 5.14.

For $\gamma := e^{-1/2}$ the expected cost of the solution is

$$\mathrm{E}\left[\sum_{e \in E(T)} c_e + \sum_{i \in V \setminus V(T)} \pi_i\right] \leq \frac{1}{1 - 1/\sqrt{e}} \cdot \mathrm{OPT}_{LP} \ .$$

Thus, we have a randomized 2.54-approximation algorithm. □

Derandomization and Integrality Gap

Derandomization.

- There are at most $n := |V|$ distinct values of y_i^*.
- Consider n sets $U_j := \{i \in V \mid y_i^* \geq y_j^*\}$.
- Any possible value of α corresponds to one of these n sets.
- Derandomize by trying each set U_j and choosing the best solution.

Integrality gap.

- There exist instances with integrality gap $2 - \frac{2}{n}$.
- By Theorem 5.14 that the integrality gap is at most $\frac{1}{1-1/\sqrt{e}} \approx 2.54$.
- We will prove later that the integrality gap is at most 2.

Randomized Algorithm for Uncapacitated Facility Location

In Chapter 4 we obtained an LP-based 4-approximation algorithm which computes a solution of cost at most

$$\sum_{i \in F} f_i \cdot y_i^* + 3 \cdot \sum_{j \in D} v_j^* \ .$$

Notation.

Let $C_j^* := \sum_{i \in F} c_{ij} \cdot x_{ij}^*$ denote the assignment cost of j paid by the LP, i. e.,

$$\mathrm{OPT}_{LP} = \sum_{i \in F} f_i \cdot y_i^* + \sum_{j \in D} C_j^* \ .$$

Idea:

- Include the assignment cost C_j^* in the analysis.
- Instead of bounding only the facility cost by OPT_{LP}, bound both the facility cost and part of the assignment cost by OPT_{LP}.

Randomized Algorithm for Uncapacitated Facility Location

Randomized algorithm for Uncapacitated Facility Location Problem

1. compute optimal LP solutions (x^*, y^*) and (v^*, w^*);
2. while $D \neq \emptyset$
3. choose $j := \operatorname{argmin}_{j' \in D}(v^*_{j'} + C^*_j)$;
4. choose $i \in N(j)$ according to probability distribution x^*_{ij};
5. assign all unassigned clients in $N^2(j)$ to facility i;
6. set $D := D \setminus N^2(j)$;

Theorem 5.15.

The algorithm above is a randomized 3-approximation algorithm for the Uncapacitated Facility Location Problem.

Proof:. . . □

Minimizing the Weighted Sum of Completion Times

Given: jobs with processing time $p_j \in \mathbb{Z}_{>0}$, weight $w_j \geq 0$, and release date $r_j \in \mathbb{Z}_{\geq 0}$, $j = 1, \ldots, n$.

Task: Schedule the jobs nonpreemptively on a single machine; minimize the total weighted completion time $\sum_{j=1}^{n} w_j \cdot C_j$.

Let $T := \max_j r_j + \sum_{j=1}^{n} p_j$ (upper bound on all completion times).

Consider an integer programming relaxation with variables

$$y_{jt} = \begin{cases} 1 & \text{if job } j \text{ is processed in time } [t-1, t), \\ 0 & \text{otherwise} \end{cases}$$

for $j = 1, \ldots, n$, $t = 1, \ldots, T$.

Integer Programming Relaxation

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{n} w_j \cdot C'_j & & \\
\text{s.t.} \quad & \sum_{j=1}^{n} y_{jt} \leq 1 & & \text{for } t = 1, \ldots, T, \\
& \sum_{t=1}^{T} y_{jt} = p_j & & \text{for } j = 1, \ldots, n, \\
& y_{jt} = 0 & & \text{for } j = 1, \ldots, n,\ t = 1, \ldots, r_j, \\
& C'_j = \frac{1}{p_j} \sum_{t=1}^{T} y_{jt}\left(t - \tfrac{1}{2}\right) + \tfrac{1}{2} p_j & & \text{for } j = 1, \ldots, n, \\
& y_{jt} \in \{0, 1\} & & \text{for } j = 1, \ldots, n,\ t = 1, \ldots, T.
\end{aligned}
$$

Remarks.

- Notice that in a feasible IP solution jobs might be preempted.
- In this case, C'_j underestimates the actual completion time of job j.

Randomized Rounding

1. compute optimal IP solution (y^*, C^*);
2. for $j = 1$ to n set random variable X_j to $t - \frac{1}{2}$ with probability y^*_{jt}/p_j;
3. sort the jobs such that $X_1 \leq X_2 \leq \cdots \leq X_n$;
4. schedule all jobs nonpreemptively and as early as possible in this order;

Lemma 5.16.

If the random variables X_j are independent, then $\mathrm{E}[C_j \mid X_j = x] \leq p_j + 2x$.

Proof:. . . □

Theorem 5.17.

The expected performance ratio of the randomized algorithm is at most 2.

Proof:. . . □

Computing an Optimum IP Solution

1. sort the jobs such that $w_1/p_1 \geq w_2/p_2 \geq \cdots \geq w_n/p_n$;
2. construct a preemptive schedule:
3. always schedule the first available job which is not yet completed;
4. implicitely assign the variables y_{jt} (and C_j) accordingly;

Lemma 5.18.
The algorithm finds an optimal IP solution in polynomial time.

Proof: Exchange argument... □

Remarks.

- There are at most n combinatorially different choices of X_j.
- Randomized rounding can be implemented to run in polynomial time.
- Derandomization by method of conditional expectations.

Minimum-Capacity Multicommodity Flow Problem

Given: Undirected graph $G = (V, E)$ and k pairs $s_i, t_i \in V$, $i = 1, \ldots, k$.

Task: Find single s_i-t_i-path in G, for $i = 1, \ldots, k$.

Objective: Minimize maximum number of paths containing same edge.

Path-based IP formulation:

$$
\begin{aligned}
\min \quad & W & & \\
\text{s.t.} \quad & \sum_{P \in \mathcal{P}_i} x_P = 1 & & \text{for all } i = 1, \ldots, k, \\
& \sum_{P : e \in P} x_P \leq W & & \text{for all } e \in E, \\
& x_P \in \{0, 1\} & & \text{for all } P \in \mathcal{P}_i,\ i = 1, \ldots, k.
\end{aligned}
$$

LP relaxation: Replace $x_P \in \{0, 1\}$ with $x_P \geq 0$.

- Despite exponential number of variables, LP relaxation can be solved in polynomial time!

Randomized Rounding

1. compute optimal LP solution (x^*, W^*);
2. for $i = 1$ to n
3. independently choose one path $P \in \mathcal{P}_i$ with probability x_P^*;

Theorem 5.19.

If $W^* \geq c \cdot \ln n$ for a large enough constant c, then with high probability, the total number of paths using any edge is at most $W^* + \sqrt{c \cdot W^* \ln n}$.

Proof:... □

Markov's Inequality and Chernoff Bound

Definition 5.20.

A probabilistic event happens with high probability if the probability that it does not occur is at most n^{-c} for some constant $c \geq 1$.

Lemma 5.21 (Markov's Inequality).

If $X \geq 0$ is a random variable, then $\Pr[X \geq a] \leq \mathrm{E}[X]/a$ for $a > 0$. □

Theorem 5.22 (Chernoff Bound).

Let $X_1, \ldots, X_k$ be independent 0-1 random variables. Then for $X := \sum_{i=1}^{k} X_i$, $\mu \geq \mathrm{E}[X]$, and $0 < \delta \leq 1$

$$\Pr[X \geq (1+\delta) \cdot \mu] < \left(\frac{e^{\delta}}{(1+\delta)^{1+\delta}} \right)^{\mu} \leq e^{-\mu \cdot \delta^2/3} .$$

□

Performance Guarantees

Corollary 5.23.

a. If $W^* \geq c \cdot \ln n$, then randomized rounding with high probability produces a solution of value at most $2W^*$.

b. If $W^* \geq 1$, then with high probability the total number of paths using any edge is $O(\log n) \cdot W^*$.

Proof:. . . □

Coloring Dense 3-Colorable Graphs

Definition 5.24.

i. A graph on n nodes is dense if for some constant $\alpha > 0$ the number of edges is at least $\alpha \cdot \binom{n}{2}$.

ii. For $0 \leq \delta < 1$, a graph on n nodes is δ-dense if every node has at least $\delta \cdot n$ neighbors.

Remark: Any δ-dense graph is dense.

We state the following Theorem without proof.

Theorem 5.25.

a. It is *NP*-hard to decide if a graph can be colored with only 3 colors, or needs at least 5 colors.

b. Assuming a variant of the *Unique Games Conjecture*, for any constant $k > 3$, it is *NP*-hard to decide if a graph can be colored with only 3 colors, or needs at least k colors. □

Random Sampling

Theorem 5.26.

With high probability, we can properly color any δ-dense 3-colorable graph.

Random Sampling Algorithm

1. select random subset $S \subseteq V$ of $O(\ln n/\delta)$ nodes such that each node in $V \setminus S$ has a neighbor in S;
2. enumerate all possible 3-colorings of S;
3. try to extend each 3-coloring of S to a 3-coloring of G;

Lemma 5.27.

With high probability, the subset S in Step 1 can be obtained by including each node with probability $(3c \ln n)/(\delta n)$ for some constant c.

Proof:... □

Lemma 5.28.

Let $X_1, \ldots, X_n$ independent 0-1 random variables. For $X := \sum_{i=1}^{n} X_i$ and $\mu \leq \mathrm{E}[X]$ it holds that $\Pr[X = 0] < e^{-\mu}$. □