Randomised Algorithms

Lecture 10: Approximation Algorithms: Set-Cover and MAX-CNF

Thomas Sauerwald (tms41@cam.ac.uk)

Lent 2024

UNIVERSITY OF CAMBRIDGE

Outline

Weighted Set Cover

MAX-CNF

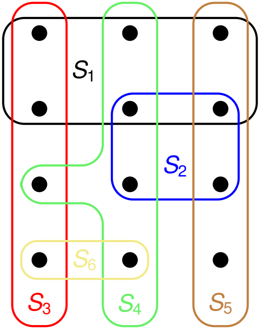
The Weighted Set-Cover Problem

Set Cover Problem

- Given: set X and a family of subsets $\mathcal{F}$, and a cost function $c : \mathcal{F} \to \mathbb{R}^+$
- Goal: Find a minimum-cost subset $\mathcal{C} \subseteq \mathcal{F}$

s.t. $$X = \bigcup_{S \in \mathcal{C}} S.$$

Sum over the costs of all sets in $\mathcal{C}$



	S_1	S_2	S_3	S_4	S_5	S_6
c :	2	3	3	5	1	2

Remarks:

- generalisation of the weighted Vertex-Cover problem
- models resource allocation problems

Setting up an Integer Program

Question: Try to formulate the integer program and linear program of the weighted SET-COVER problem
(solution on next slide!)

Setting up an Integer Program

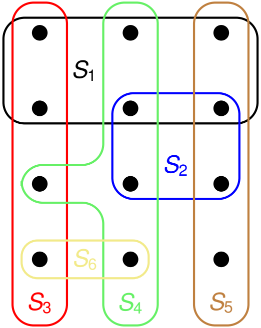
0-1 Integer Program

$$\begin{array}{llcll}
\text{minimize} & \displaystyle\sum_{S \in \mathcal{F}} c(S) y(S) & & & \\
\text{subject to} & \displaystyle\sum_{S \in \mathcal{F}\,:\, x \in S} y(S) & \geq & 1 & \text{for each } x \in X \\
& y(S) & \in & \{0, 1\} & \text{for each } S \in \mathcal{F}
\end{array}$$

Linear Program

$$\begin{array}{llcll}
\text{minimize} & \displaystyle\sum_{S \in \mathcal{F}} c(S) y(S) & & & \\
\text{subject to} & \displaystyle\sum_{S \in \mathcal{F}\,:\, x \in S} y(S) & \geq & 1 & \text{for each } x \in X \\
& y(S) & \in & [0, 1] & \text{for each } S \in \mathcal{F}
\end{array}$$

Back to the Example



	S_1	S_2	S_3	S_4	S_5	S_6
c :	2	3	3	5	1	2
$\overline{y}(.)$:	$1/2$	$1/2$	$1/2$	$1/2$	1	$1/2$

Cost equals 8.5

The strategy employed for Vertex-Cover would take all 6 sets!

Even worse: If all $\overline{y}$'s were below $1/2$, we would not even return a valid cover!

Randomised Rounding

	S_1	S_2	S_3	S_4	S_5	S_6
c :	2	3	3	5	1	2
$\overline{y}(.)$:	1/2	1/2	1/2	1/2	1	1/2

Idea: Interpret the $\overline{y}$-values as probabilities for picking the respective set.

Randomised Rounding

- Let $\mathcal{C} \subseteq \mathcal{F}$ be a random set with each set S being included independently with probability $\overline{y}(S)$.
- More precisely, if $\overline{y}$ denotes the optimal solution of the LP, then we compute an integral solution y by:

$$y(S) = \begin{cases} 1 & \text{with probability } \overline{y}(S) \\ 0 & \text{otherwise.} \end{cases} \qquad \text{for all } S \in \mathcal{F}.$$

- Therefore, $\mathbf{E}[\, y(S) \,] = \overline{y}(S)$.

Randomised Rounding

	S_1	S_2	S_3	S_4	S_5	S_6
c :	2	3	3	5	1	2
$\overline{y}(.)$:	$1/2$	$1/2$	$1/2$	$1/2$	1	$1/2$

Idea: Interpret the $\overline{y}$-values as probabilities for picking the respective set.

Lemma

- The expected cost satisfies

$$\mathbf{E}\left[\, c(\mathcal{C}) \,\right] = \sum_{S \in \mathcal{F}} c(S) \cdot \overline{y}(S)$$

- The probability that an element $x \in X$ is covered satisfies

$$\mathbf{P}\left[x \in \bigcup_{S \in \mathcal{C}} S \right] \geq 1 - \frac{1}{e}.$$

Proof of Lemma

Lemma

Let $\mathcal{C} \subseteq \mathcal{F}$ be a random subset with each set S being included independently with probability $\overline{y}(S)$.

- The expected cost satisfies $\mathbf{E}[\, c(\mathcal{C})\,] = \sum_{S \in \mathcal{F}} c(S) \cdot \overline{y}(S)$.
- The probability that x is covered satisfies $\mathbf{P}[\, x \in \cup_{S \in \mathcal{C}} S\,] \geq 1 - \frac{1}{e}$.

Proof:

- **Step 1**: The expected cost of the random set $\mathcal{C}$

$$\mathbf{E}[\, c(\mathcal{C})\,] = \mathbf{E}\left[\sum_{S \in \mathcal{C}} c(S)\right] = \mathbf{E}\left[\sum_{S \in \mathcal{F}} \mathbf{1}_{S \in \mathcal{C}} \cdot c(S)\right]$$

$$= \sum_{S \in \mathcal{F}} \mathbf{P}[\, S \in \mathcal{C}\,] \cdot c(S) = \sum_{S \in \mathcal{F}} \overline{y}(S) \cdot c(S).$$

- **Step 2**: The probability for an element to be (not) covered

$$\mathbf{P}[\, x \notin \cup_{S \in \mathcal{C}} S\,] = \prod_{S \in \mathcal{F}:\, x \in S} \mathbf{P}[\, S \notin \mathcal{C}\,] = \prod_{S \in \mathcal{F}:\, x \in S} (1 - \overline{y}(S))$$

$$\leq \prod_{S \in \mathcal{F}:\, x \in S} e^{-\overline{y}(S)}$$

$1 + x \leq e^x$ for any $x \in \mathbb{R}$

$$= e^{-\sum_{S \in \mathcal{F}:\, x \in S} \overline{y}(S)} \leq e^{-1} \qquad \square$$

$\overline{y}$ solves the LP!

The Final Step

Lemma

Let $\mathcal{C} \subseteq \mathcal{F}$ be a random subset with each set S being included independently with probability $y(S)$.

- The expected cost satisfies $\mathbf{E}[\, c(\mathcal{C})\,] = \sum_{S \in \mathcal{F}} c(S) \cdot y(S)$.
- The probability that x is covered satisfies $\mathbf{P}[\, x \in \cup_{S \in \mathcal{C}} S\,] \geq 1 - \frac{1}{e}$.

Problem: Need to make sure that every element is covered!

Idea: Amplify this probability by taking the union of $\Omega(\log n)$ random sets $\mathcal{C}$.

```
WEIGHTED SET COVER-LP(X, F, c)
compute ȳ, an optimal solution to the linear program
C = ∅
repeat 2 ln n times
    for each S ∈ F
        let C = C ∪ {S} with probability ȳ(S)
return C
```

clearly runs in polynomial-time!

Analysis of Weighted Set Cover-LP

Theorem

- With probability at least $1 - \frac{1}{n}$, the returned set $\mathcal{C}$ is a valid cover of X.
- The expected approximation ratio is $2\ln(n)$.

Proof:

- **Step 1**: The probability that $\mathcal{C}$ is a cover
  - By previous Lemma, an element $x \in X$ is covered in one of the $2\ln n$ iterations with probability at least $1 - \frac{1}{e}$, so that

$$\mathbf{P}\left[\, x \notin \cup_{S \in \mathcal{C}} S \,\right] \leq \left(\frac{1}{e}\right)^{2\ln n} = \frac{1}{n^2}.$$

  - This implies for the event that all elements are covered:

$$\mathbf{P}\left[\, X = \cup_{S \in \mathcal{C}} S \,\right] = 1 - \mathbf{P}\left[\bigcup_{x \in X} \{x \notin \cup_{S \in \mathcal{C}} S\}\right]$$

$\mathbf{P}\left[\, A \cup B \,\right] \leq \mathbf{P}\left[\, A \,\right] + \mathbf{P}\left[\, B \,\right]$

$$\geq 1 - \sum_{x \in X} \mathbf{P}\left[\, x \notin \cup_{S \in \mathcal{C}} S \,\right] \geq 1 - n \cdot \frac{1}{n^2} = 1 - \frac{1}{n}.$$

- **Step 2**: The expected approximation ratio
  - By previous lemma, the expected cost of one iteration is $\sum_{S \in \mathcal{F}} c(S) \cdot \overline{y}(S)$.
  - Linearity $\Rightarrow \mathbf{E}\left[\, c(\mathcal{C}) \,\right] \leq 2\ln(n) \cdot \sum_{S \in \mathcal{F}} c(S) \cdot \overline{y}(S) \leq 2\ln(n) \cdot c(\mathcal{C}^*)$ $\square$

Analysis of WEIGHTED SET COVER-LP

Theorem

- With probability at least $1 - \frac{1}{n}$, the returned set $\mathcal{C}$ is a valid cover of X.
- The expected approximation ratio is $2\ln(n)$.

By Markov's inequality, $\mathbf{P}[\, c(\mathcal{C}) \leq 4\ln(n) \cdot c(\mathcal{C}^*) \,] \geq 1/2$.

Hence with probability at least $1 - \frac{1}{n} - \frac{1}{2} > \frac{1}{3}$, solution is valid and within a factor of $4\ln(n)$ of the optimum.

probability could be further increased by repeating

Typical Approach for Designing Approximation Algorithms based on LPs

[Exercise Question (9/10).10] gives a different perspective on the amplification procedure through non-linear randomised rounding.

Outline

Weighted Set Cover

MAX-CNF

MAX-CNF

Recall:

MAX-3-CNF Satisfiability

- Given: 3-CNF formula, e.g.: $(x_1 \vee x_3 \vee \overline{x_4}) \wedge (x_2 \vee \overline{x_3} \vee \overline{x_5}) \wedge \cdots$
- Goal: Find an assignment of the variables that satisfies as many clauses as possible.

MAX-CNF Satisfiability (MAX-SAT)

- Given: CNF formula, e.g.: $(x_1 \vee \overline{x_4}) \wedge (x_2 \vee \overline{x_3} \vee x_4 \vee \overline{x_5}) \wedge \cdots$
- Goal: Find an assignment of the variables that satisfies as many clauses as possible.

Why study this generalised problem?

- Allowing arbitrary clause lengths makes the problem more interesting (we will see that simply guessing is not the best!)
- a nice concluding example where we can practice previously learned approaches

Approach 1: Guessing the Assignment

Assign each variable true or false uniformly and independently at random.

Recall: This was the successful approach to solve MAX-3-CNF!

Analysis

For any clause i which has length ℓ,

$$\mathbf{P}[\text{clause } i \text{ is satisfied}] = 1 - 2^{-\ell} := \alpha_\ell.$$

In particular, the guessing algorithm is a randomised 2-approximation.

Proof:

- First statement as in the proof of Theorem 35.6. For clause i not to be satisfied, all ℓ occurring variables must be set to a specific value.
- As before, let $Y := \sum_{i=1}^{m} Y_i$ be the number of satisfied clauses. Then,

$$\mathbf{E}[Y] = \mathbf{E}\left[\sum_{i=1}^{m} Y_i\right] = \sum_{i=1}^{m} \mathbf{E}[Y_i] \geq \sum_{i=1}^{m} \frac{1}{2} = \frac{1}{2} \cdot m. \qquad \square$$

Approach 2: Guessing with a "Hunch" (Randomised Rounding)

First solve a linear program and use fractional values for a **biased** coin flip.

The same as randomised rounding!

0-1 Integer Program

These auxiliary variables are used to reflect whether a clause is satisfied or not

$$\begin{array}{llcll}
\text{maximize} & \displaystyle\sum_{i=1}^{m} z_i & & & \\
\text{subject to} & \displaystyle\sum_{j \in C_i^+} y_j + \sum_{j \in C_i^-} (1 - y_j) & \geq & z_i & \text{for each } i = 1, 2, \ldots, m \\
 & z_i & \in & \{0, 1\} & \text{for each } i = 1, 2, \ldots, m \\
 & y_j & \in & \{0, 1\} & \text{for each } j = 1, 2, \ldots, n
\end{array}$$

C_i^+ is the index set of the un-negated variables of clause i.

- In the corresponding LP each $\in \{0, 1\}$ is replaced by $\in [0, 1]$
- Let $(\overline{y}, \overline{z})$ be the optimal solution of the LP
- Obtain an integer solution y through randomised rounding of $\overline{y}$

Analysis of Randomised Rounding

Lemma

For any clause i of length ℓ,

$$\mathbf{P}\left[\text{clause } i \text{ is satisfied}\right] \geq \left(1 - \left(1 - \frac{1}{\ell}\right)^{\ell}\right) \cdot \overline{z}_i.$$

Proof of Lemma (1/2):

- Assume w.l.o.g. all literals in clause i appear non-negated (otherwise replace every occurrence of x_j by $\overline{x_j}$ in the whole formula)
- Further, by relabelling assume $C_i = (x_1 \vee \cdots \vee x_\ell)$

$$\Rightarrow \quad \mathbf{P}\left[\text{clause } i \text{ is satisfied}\right] = 1 - \prod_{j=1}^{\ell} \mathbf{P}\left[\ y_j \text{ is false }\ \right] = 1 - \prod_{j=1}^{\ell} (1 - \overline{y}_j)$$

Arithmetic vs. geometric mean:

$$\frac{a_1 + \ldots + a_k}{k} \geq \sqrt[k]{a_1 \times \ldots \times a_k}.$$

$$\geq 1 - \left(\frac{\sum_{j=1}^{\ell}(1 - \overline{y}_j)}{\ell}\right)^{\ell}$$

$$= 1 - \left(1 - \frac{\sum_{j=1}^{\ell} \overline{y}_j}{\ell}\right)^{\ell} \geq 1 - \left(1 - \frac{\overline{z}_i}{\ell}\right)^{\ell}.$$

Analysis of Randomised Rounding

Lemma

For any clause i of length ℓ,

$$\mathbf{P}\left[\text{clause } i \text{ is satisfied}\right] \geq \left(1 - \left(1 - \frac{1}{\ell}\right)^{\ell}\right) \cdot \overline{z}_i.$$

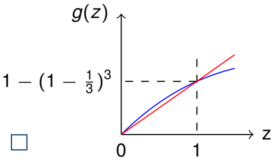
Proof of Lemma (2/2):

- So far we have shown:

$$\mathbf{P}\left[\text{clause } i \text{ is satisfied}\right] \geq 1 - \left(1 - \frac{\overline{z}_i}{\ell}\right)^{\ell}$$

- For any $\ell \geq 1$, define $g(z) := 1 - \left(1 - \frac{z}{\ell}\right)^{\ell}$. This is a concave function with $g(0) = 0$ and $g(1) = 1 - \left(1 - \frac{1}{\ell}\right)^{\ell} =: \beta_\ell$.

$$\Rightarrow \quad g(z) \geq \beta_\ell \cdot z \qquad \text{for any } z \in [0,1]$$



- Therefore, $\mathbf{P}\left[\text{clause } i \text{ is satisfied}\right] \geq \beta_\ell \cdot \overline{z}_i$. $\square$

Analysis of Randomised Rounding

Lemma

For any clause i of length ℓ,

$$\mathbf{P}[\,\text{clause } i \text{ is satisfied}\,] \geq \left(1 - \left(1 - \frac{1}{\ell}\right)^{\ell}\right) \cdot \overline{z}_i.$$

Theorem

Randomised Rounding yields a $1/(1 - 1/e) \approx 1.5820$ randomised approximation algorithm for MAX-CNF.

Proof of Theorem:

- For any clause $i = 1, 2, \ldots, m$, let ℓ_i be the corresponding length.
- Then the expected number of satisfied clauses is:

$$\mathbf{E}[\,Y\,] = \sum_{i=1}^{m} \mathbf{E}[\,Y_i\,] \geq \sum_{i=1}^{m} \left(1 - \left(1 - \frac{1}{\ell_i}\right)^{\ell_i}\right) \cdot \overline{z}_i \geq \sum_{i=1}^{m} \left(1 - \frac{1}{e}\right) \cdot \overline{z}_i \geq \left(1 - \frac{1}{e}\right) \cdot \text{OPT}$$

By Lemma

Since $(1 - 1/x)^x \leq 1/e$

LP solution at least as good as optimum

Approach 3: Hybrid Algorithm

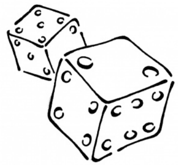
Summary

- Approach 1 (Guessing) achieves better guarantee on longer clauses
- Approach 2 (Rounding) achieves better guarantee on shorter clauses

Idea: Consider a hybrid algorithm which interpolates between the two approaches

HYBRID-MAX-CNF(φ, n, m)

Let $b \in \{0, 1\}$ be the flip of a fair coin
If $b = 0$ **then** perform random guessing
If $b = 1$ **then** perform randomised rounding
return the computed solution



Algorithm sets each variable x_i to TRUE with prob. $\frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \overline{y}_i$.
Note, however, that variables are **not** independently assigned!

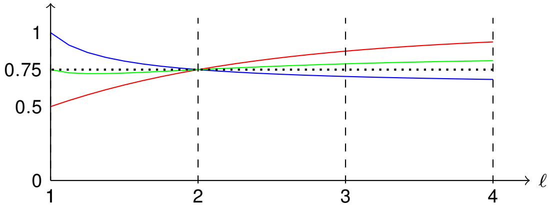
Analysis of Hybrid Algorithm

Theorem

HYBRID-MAX-CNF(φ, n, m) is a randomised 4/3-approx. algorithm.

Proof:

- It suffices to prove that clause i is satisfied with probability at least $3/4 \cdot \overline{z}_i$
- For any clause i of length ℓ:
  - Algorithm 1 satisfies it with probability $1 - 2^{-\ell} = \alpha_\ell \geq \alpha_\ell \cdot \overline{z}_i$.
  - Algorithm 2 satisfies it with probability $\beta_\ell \cdot \overline{z}_i$.
  - HYBRID-MAX-CNF(φ, n, m) satisfies it with probability $\frac{1}{2} \cdot \alpha_\ell \cdot \overline{z}_i + \frac{1}{2} \cdot \beta_\ell \cdot \overline{z}_i$.
- Note $\frac{\alpha_\ell+\beta_\ell}{2} = 3/4$ for $\ell \in \{1, 2\}$, and for $\ell \geq 3$, $\frac{\alpha_\ell+\beta_\ell}{2} \geq 3/4$ (see figure)
- $\Rightarrow$ HYBRID-MAX-CNF(φ, n, m) satisfies it with prob. at least $3/4 \cdot \overline{z}_i$ □



MAX-CNF Conclusion

Summary

- Since $\alpha_2 = \beta_2 = 3/4$, we cannot achieve a better approximation ratio than $4/3$ by combining Algorithm 1 & 2 in a different way
- The $4/3$-approximation algorithm can be easily derandomised
  - Idea: use the conditional expectation trick for both Algorithm 1 & 2 and output the better solution
- The $4/3$-approximation algorithm applies unchanged to a weighted version of MAX-CNF, where each clause has a non-negative weight
- Even MAX-2-CNF (every clause has length 2) is NP-hard!