INF554: Using randomness in algorithms	**Autumn 2014**

Cours 4 — October 6th

Enseignant : Frédéric Magniez — *Rédacteur : Marc Sanselme*

4.1 Exercises

4.1.1 Max-SAT

Problem

Input: A SAT formula φ with n variables $X_1, \ldots, X_n$ and m clauses $\varphi = C_1 \wedge \ldots \wedge C_m$
Output: $a \in \{0,1\}^n$ which maximizes the number of satisfied clauses in φ

Strategy

1. Write the problem as a integer linear program
2. (a) Is it possible to find an optimal solution in polynomial time?
   (b) How to link it with the initial problem?
3. Find a random algorithm to calculate a solution in $a \in 0, 1^n$ such that
$$\mathbb{P}_a(C(a) = 1) \geq \beta_k z^*_C,$$
where C has exactly k variables and $\beta_k = 1 - (1 - \frac{1}{k})^k$
4. (a) Find an algorithm with approximation factor $(1 - \frac{1}{e})$ in average
   (b) Try to derandomize it

Solution

1.
$$\begin{aligned} w &= \max \sum_{C \in \varphi} z_C \\ \text{such that} \quad & \begin{cases} \forall C \in \varphi : 0 \leq z_C \leq 1 \\ \forall i \in 1, \ldots, n : 0 \leq x_i \leq 1 \\ \forall C \in \varphi : \sum_{i:X_i \in C} x_i + \sum_{i:\overline{X_i} \in C} (1 - x_i) \geq z_C \end{cases} \\ \text{and} \quad & \forall C \in \varphi : z_C \in \mathbb{Z} \\ & \forall i \in \{1, \ldots, n\} : x_i \in \mathbb{Z} \end{aligned}$$

This integer program characterizes the problem:

- $\Rightarrow$ Let $a \in \{0,1\}^n$ be any assigment. Set $x = a$, $z_C = C(a)$. Then $\sum_{C \in C} z_C$ equals the number of clauses that a satisfies in φ.
- $\Leftarrow$ Conversely, let $x \in \{0,1\}^n$ and let z be some optimal solution to the integer program with given x. Set $a = x$. Then

$$C(a) = \sum_{i:X_i \in C} x_i + \sum_{i:\overline{X_i} \in C} (1 - x_i) \geq z_C.$$

  The last inequality is in fact here an equality since z is optimal for x, and therefore a satisfies exactly $\sum_{C \in \varphi} Z_C$ in φ.

2. (a) It is possible to compute the optimal solution of a linear program (if variables are all reals and not integers) in time polynomial in the program size, that is here in n and m.

   (b) Let (x^*, z^*) be any solution maximizing the linear program, and let w^* be its value. Then $w^* \leq$ Max-SAT(φ) since the linear program has ben relaxed to real variables.

3. Chose independently at random each bit a_i such that $P(a_i = 1) = x_i$. Then it follows that for any C having exactly k variables:

$$\begin{aligned}
\mathbb{P}_a(C(a) = 0) &= \prod_{i:X_i \in C} (1 - x_i^*) \times \prod_{i:\overline{X_i} \in C} (x_i^*) \\
&\leq \left(\sum_{i:X_i \in C} (1 - x_i^*) + \sum_{i:\overline{X_i} \in C} (x_i^*) \right)^k \\
&\leq \left(1 - \frac{1}{k} \left(\sum_{i:X_i \in C} x_i^* + \sum_{i:\overline{X_i} \in C} (1 - x_i^*) \right) \right)^k \\
&\leq (1 - \frac{z_C^*}{k})^k \\
&\leq 1 - \beta_k z_C^*,
\end{aligned}$$

   where $\beta_k = 1 - (1 - \frac{1}{k})^k$, and because $t \mapsto 1 - (1 - \frac{t}{k})^k$ is an increasing and concave function.

4. (a)

$$\begin{aligned}
\mathbb{E}_a(\#\text{satisfied clauses}) &\geq \sum_C \beta_{k_C} Z_C^* \\
&\geq (1 - \tfrac{1}{e}) \sum_C Z_C^* \\
&\geq (1 - \tfrac{1}{e}) w^* \\
&\geq (1 - \tfrac{1}{e}) \text{Max-SAT}(\varphi).
\end{aligned}$$

(b) Since we have

$$\begin{aligned}\mathbb{E}_a(\#\text{satisfied clauses}) &= \mathbb{E}_a(\#\text{satisfied clauses}|X_1 = 0) \times P(a_1 = 0)\\ &\quad +\mathbb{E}_a(\#\text{satisfied clauses}|X_1 = 1) \times P(a_1 = 1),\end{aligned}$$

there must be a value of $a_1 \in \{0, 1\}$ such that

$$\mathbb{E}(\#\text{satisfied clauses}|X_1 = a_1) \geq \mathbb{E}(\#\text{satisfied clauses}).$$

We can then proceed the other variables inductively, leading to the following algorithm:

Algorithm:
For $i = 1 \ldots n$
Try $a_i = 0$
Compute $\mathbb{E}_a(\#\text{satisfied clauses}|X_1 = a_1 ... X_i = a_i)$
If $\leq \mathbb{E}_a(\#\text{satisfied clauses})$ then set $a_i := 1$
Return a

We conclude by observing that we can combine this algorithm with the one we have seen in class. More precisely, given a random assignment a chosen uniformly at random in $\{0, 1\}^n$, we have seen that any clauses with exactly k variables is satisfied with the following probability:

$$\mathbb{P}_{\text{uniform } a}(C(a) = 1) \geq (1 - 2^{-k}) = \alpha_k.$$

Observe first that for any value Z^*_k we have $\alpha_k \geq \alpha_k Z^*_k$. Moreover, one can prove that for all $k \geq 1$:

$$\frac{\alpha_k + \beta_k}{2} \geq \frac{3}{4}$$

Thus, considering the sampling procedure, which first flip a random bit, and according to this bit either sample $a \in \{0, 1, \}^n$ uniformly at random or according to the previous distribution (each bit a_i are sample independently such that $P(a_i = 1) = x^*_i$). Then any clause becomes satisfiable with probability at least 3/4, leading to an randomized algorithm for Max-SAT with approximation ratio $\frac{4}{3}$. This one can again be derandomized by returning the best value from the two derandomized underlying algorithms.

4.1.2 Min cut

Problem

Input: $G : (V, E)$ a connected graph with n vertices and m edges
Output: $C \subset E$ a cut (i.e. removing C from G creates at least 2 disjoint connected components) such that the size of C is minimal.

Algorithm

Select a random edge e uniformly at random
Contract e
Repeat this process until only two vertices a, b remeain Return the set C of remaining edges between a and b

Analysis

Let C be any cut of minimal size k. coupe minimale quelconque de taille k. First we show that if the algorithm never choses an edge in C, then after $(n-2)$ iterations, it returns C. Indeed, let C_a be the connected component of a in $G \setminus C$, and let similarly C_b be the connected component of a in $G \setminus C$. Then all removed edges are within $G_{|C_a}$ or $G_{|C_b}$ since C is a cut.

We now bound the probability that the algorithm never choses an edge in C. Let E_i be the event "the contracted edge at step i is not in C". Then define $F_i = \cap_{j=1}^{i} E_j$. We will lower bound by induction $\mathbb{P}(F_{n-2})$, which is the probability we want to estimate.

First, when $i = 1$, vertices in the original graph G have all degree at least k, otherwise there would be a cut with smaller size. Therefore

$$\mathbb{P}(F_1) = \frac{k}{m} \leq \frac{2}{n}$$

, since the fact that all vertices have degree at least k implies $m \geq \frac{kn}{2}$.

Then, for $i \geq 2$ and assuming that F_{i-1} occurs, the set C is still a cut of minimal size k in the reduced graph (that is G where selected edges has been contracted). But now the graph has now only $(n - i + 1)$ vertices remaining, each of degree at least k. Therefore, as before, we get donc

$$\mathbb{P}(E_i|F_{i-1}) \geq 1 - \frac{2}{n+1-i}.$$

We can now compute $\mathbb{P}(F_{n-2})$ using $F_i = E_i \cap F_{i-1}$ and conditional probabilities as follows:

$$\begin{aligned}
\mathbb{P}(F_{n-2}) &= \mathbb{P}(E_{n-2}|F_{n-3})\mathbb{P}(F_{n-3}) \\
&= \mathbb{P}(E_{n-2}|F_{n-3})\mathbb{P}(E_{n-3}|F_{n-4})\ldots\mathbb{P}(E_2|F_1)\mathbb{P}(F_1) \\
&\geq \left(1-\frac{2}{3}\right)\left(1-\frac{2}{4}\right)\ldots\left(1-\frac{2}{n-1}\right)\left(1-\frac{2}{n}\right) \\
&= \frac{1 \times 2 \times \ldots \times (n-3) \times (n-2)}{3 \times 4 \ldots \times (n-1) \times n} \\
&= \frac{2}{n(n-1)}.
\end{aligned}$$

This probability can be posted to any success probability $(1-\delta)$ by executing the algorithm $\log(n/\delta)$ times, and taking the best cut. This number of execution is enough since we are in a case similar to the one of one-sided error algorithms: the probability to get a better cut, if the best current computed cut is not optimal, is at least $\frac{2}{n(n-1)}$ at each execution of the algorithm.