- 教材讨论
  - JH第5章第4节

问题1：去随机

- 去随机追求的目标是什么？
  - To obtain a required approximation ratio or correct results efficiently for all input instances rather than merely with a high probability.
  - Without any essential increase in the amount of computational resources.

问题2：reduction of the probability space size

- 通过降低变量独立性来减小概率空间的基本思路是什么？
- 定理5.4.2.2是如何具体实现这一思路的？

Theorem 5.4.2.2. *Let $(\Omega, Prob)$ be a probability space and $X_1, \ldots, X_n$ be random variables over $(\Omega, Prob)$ such that $\Omega = \{(X_1 = \alpha_1, \ldots, X_n = \alpha_n) \,|\, \alpha_i \in \{0,1\}$ for $i = 1, \ldots, n\}$. Let k be a positive integer, $2 \leq k < n$, and let q be a prime power such that $q \geq n$. Let $\mathrm{GF}(q) = \{r_1, r_2, \ldots, r_q\}$ be the finite field of q elements. Let*

$$A_{i,1} = \{r_1, r_2, \ldots, r_{d_i}\} \text{ and } A_{i,0} = \mathrm{GF}(q) - A_{i,1},$$

where $d_i = \lceil q \cdot \mathrm{Prob}\,[X_i = 1] - \frac{1}{2} \rceil$.
Then, the probability space (S, Pr), where $S = \{p \,|\, p$ is a polynomial over $\mathrm{GF}(q)$ with degree at most $k-1\}$ and Pr is the uniform distribution over S, has the following properties:

(i) $|S| = q^k$,
(ii) for $i = 1, \ldots, n$, the random variables $X'_i : S \to \{0,1\}$ defined by

$$X'_i(p) = 1 \text{ iff } p(r_i) \in A_{i,1}$$

and

$$X'_i(p) = 0 \text{ iff } p(r_i) \in A_{i,0}$$

satisfy the following properties:
a) $X'_1, \ldots, X'_n$ *are k-wise independent,*
b) $|Pr[X'_i = \alpha] - Prob[X_i = \alpha]| \leq \frac{1}{2q}$ *for every* $\alpha \in \{0,1\}$.

问题2：reduction of the probability space size (续)

- 如何利用定理5.4.2.2来减少一个随机算法的random bits并最终去随机？
- 对于MAX-E*k*SAT是如何具体实现的？
- 如何评价这套方法的理论和实际意义？

问题2：reduction of the probability space size (续)

Algorithm 5.4.2.5. RED(A)

Input: An input w as for A.
Step 1: Choose uniformly at random an element p from the probability space (S, Pr) described in Theorem 5.4.2.2.
{Observe that this can be performed by $\lceil \log_2 |S| \rceil = O(k \cdot \log_2 q)$ random bits.}
Step 2: Compute $X_1', X_2', \ldots, X_n'$ as described in (ii) of Theorem 5.4.2.2.
Step 3: Run the algorithm A on w with the sequence of k-wise independent random bits $X_1', X_2', \ldots, X_n'$.
Output: The output of A computed in Step 3.

Deterministic simulation of A by probability space reduction, PSR(A)

Input: An input w consistent as an input of A.
Step 1: Create the probability space (S, Pr) as described in Theorem 5.4.2.2.
Step 2: **for** every $p \in S$ **do**
simulate RED(A) on w with the random choice p and save the output $Result(p)$.
Step 3: Estimate the "right" output from all outputs computed in Step 2.
{Obviously, Step 3 depends on the kind of computing problem. If one considers an optimization problem then the output with the best cost is chosen. If A has been designed for a decidability problem, one has to look on the probabilities of the answers "accept" and "reject".}

问题2：reduction of the probability space size (续)

Algorithm 5.4.3.1. DERAND-RSMS-3

Input: A formula Φ in 3-CNF over a set of variables $\{x_1, \ldots, x_n\}$.
Step 1: Find a positive integer r such that

$$q := 2^r \geq n$$

and r is the smallest integer with the property $2^r \geq n$.

Step 2: **for** $c_0 = 0$ **to** $q - 1$ **do**
  for $c_1 = 0$ **to** $q - 1$ **do**
    for $c_2 = 0$ **to** $q - 1$ **do**
      begin
      for $i = 1$ **to** n **do**
        if $c_2 r_i^2 + c_1 r_i + c_0 \in \{r_1, r_2, \ldots, r_{q/2}\} \in \mathrm{GF}(q)$
          then $x_i = 1$
          else $x_i = 0$;
      count the number of satisfied clauses of Φ by $x_1, \ldots, x_n$, and save the assignment $(\alpha_1, \ldots, \alpha_n)$ with the maximal number of satisfied clauses up till now.
      end

Output: $\alpha_1, \ldots, \alpha_n$.

问题2： reduction of the probability space size (续)

Finally, we observe that the derandomization method by the reduction of the size of the probability space is quite general and very powerful. But the complexity of the resulting deterministic algorithm may be too high. Already $O(n^4)$ for a formula in 3-CNF of n variables may be too large. Thus, from the practical point of view, the possibility of essentially reducing the number of random bits (choices) in a randomized algorithm may be the main current contribution of this method.

问题3：conditional probabilities

- 通过条件概率来去随机的基本思路是什么？
  - 面向什么样的问题？
  - 目标是什么？
  - 具体方法/算法是什么？为什么可以达成目标？
  - 算法的关键步骤是什么？
- 对于MAX-E*k*SAT和RSMS是如何具体实现这一关键步骤的？
- 算法5.4.5.3、5.4.5.4、5.4.5.6分别在做什么？

问题3：conditional probabilities (续)

Lemma 5.4.4.1. *Let $(\Omega, Prob)$ be a probability space, and $X_1, \ldots, X_n$, and Z be random variables as described above. If, for a given input w, $\beta = \beta_1\beta_2\ldots\beta_n \in \{0,1\}^n$ is computed by the method of pessimistic estimators, then*

$$\begin{aligned} E[Z] \le E[Z|X_1 = \beta_1] \le E[Z|X_1 = \beta_1, X_2 = \beta_2] \le \cdots \\ \le E[Z|X_1 = \beta_1, \ldots, X_n = \beta_n] = cost(A_\beta(w)). \end{aligned}$$

Proof. The fact that $E[Z|X_1 = \beta_1, \ldots, X_n = \beta_n] = cost(A_\beta(w))$ is obvious. In what follows we prove for every $i = 0, 1, \ldots, n-1$ that

$$E[Z|X_1 = \beta_1, \ldots, X_i = \beta_i] \le E[Z|X_1 = \beta_1, \ldots, X_i = \beta_i, X_{i+1} = \beta_{i+1}]. \tag{5.38}$$

Since $X_1, X_2, \ldots, X_n$ are considered to be independent, it can be easily observed that

$$\begin{aligned} &E[Z|X_1 = \alpha_1, \ldots, X_i = \alpha_i] = \\ &\quad Prob[X_{i+1} = 1] \cdot E[Z|X_1 = \alpha_1, \ldots, X_i = \alpha_i, X_{i+1} = 1] + \\ &\quad Prob[X_{i+1} = 0] \cdot E[Z|X_1 = \alpha_1, \ldots, X_i = \alpha_i, X_{i+1} = 0] \end{aligned}$$

for every $\alpha_1, \ldots, \alpha_i \in \{0,1\}^i$. Since $Prob[X_{i+1} = 1] = 1 - Prob[X_{i+1} = 0]$ and the weighted mean of two numbers cannot be larger than their maximum we obtain

$$\begin{aligned} &E[Z|X_1 = \beta_1, \ldots, X_i = \beta_i] \le \\ &\max\{E[Z|X_1 = \beta_1, \ldots, X_i = \beta_i, X_{i+1} = 1], \\ &\qquad E[Z|X_1 = \beta_1, \ldots, X_i = \beta_i, X_{i+1} = 0]\}. \end{aligned} \tag{5.39}$$

Since our choice for β_{i+1} corresponds to the choice of the maximum of the conditional probabilities in (5.39), (5.39) directly implies (5.38). □

问题3：conditional probabilities (续)

Algorithm 5.4.4.2. COND-PROB(A)

Input: A consistent input w for A.
Step 1: **for** $i := 1$ **to** n **do**
  if $E[Z|X_1 = \beta_1, \ldots, X_{i-1} = \beta_{i-1}, X_i = 1] \geq$
   $E[Z|X_1 = \beta_1, \ldots, X_{i-1} = \beta_{i-1}, X_i = 0]$
  then $\beta_i := 1$
  else $\beta_i := 0$
Step 2: Simulate the work of A_β on w, where $\beta = \beta_1\beta_2 \ldots \beta_n$.
Output: $A_\beta(w)$.

问题3：conditional probabilities (续)

Algorithm 5.4.5.1. CCP

Input: Φ and $\alpha_1, \ldots, \alpha_i \in \{0,1\}^i$ for some positive integer i.

Step 1: **for** $j = 1$ **to** m **do**

begin replace the variables $x_1, \ldots, x_i$ by the constants $\alpha_1, \ldots, \alpha_i$, respectively, in the clause C_j and denote by $C_j(\alpha_1, \ldots, \alpha_i)$ the resulting simplified clause;

if $C_j \equiv 0$

then set $E[Z_j | X_1 = \alpha_1, \ldots, X_i = \alpha_i] := 0$

else if $C_j \equiv 1$

then set $E[Z_j | X_1 = \alpha_1, \ldots, X_i = \alpha_i] := 1$

else set $E[Z_j | X_1 = \alpha_1, \ldots, X_i = \alpha_i] := 1 - \frac{1}{2^l}$

where l is the number of different variables appearing in $C_j(\alpha_1, \ldots, \alpha_i)$.

end

Step 2: $E[Z | X_1 = \alpha_1, \ldots, X_i = \alpha_i] := \sum_{j=1}^{m} E[Z_j | X_1 = \alpha_1, \ldots, X_i = \alpha_i]$.

Output: $E[Z | X_1 = \alpha_1, \ldots, X_i = \alpha_i]$.

问题3：conditional probabilities (续)

Algorithm 5.4.5.3. CCP-LP

Input: Φ and $\alpha_j = \alpha(x_j)$ for $j = 1, \ldots, n$, where $\alpha(x_j)$ is the solution of $\mathrm{LP}(\Phi)$ for the Boolean variable x_j, and $\beta_1 \ldots \beta_i \in \{0,1\}^i$ for some integer i, $1 \le i \le n$.

Step 1: **for** $j = 1$ **to** m **do**

begin replace the variables $x_1, \ldots, x_i$ by the constants $\beta_1, \ldots, \beta_i$, respectively, in the clause C_j and let $C_j(\beta_1, \ldots, \beta_i) = x_{l_1}^{\gamma_1} \vee x_{l_2}^{\gamma_2} \vee \cdots \vee x_{l_r}^{\gamma_r}$ be the resulting simplified clause;

if $C_j \equiv \delta$ for some $\delta \in \{0,1\}$ {i.e., $r = 0$}

then set $E[Z_j | X_1 = \beta_1, \ldots, X_i = \beta_i] := \delta$

else set $E[Z_j | X_1 = \beta_1, \ldots, X_i = \beta_i] := 1 - \prod_{i=1}^{r} |\gamma_i - \alpha(x_{l_i})|$

end

Step 2: $E[Z | X_1 = \beta_1, \ldots, X_i = \beta_i] := \sum_{j=1}^{m} E[Z_j | X_1 = \beta_1, \ldots, X_i = \beta_i]$.

Output: $E[Z | X_1 = \beta_1, \ldots, X_i = \beta_i]$.

Algorithm 5.4.5.4. DER-RRRMS

Input: A formula Φ over $X = \{x_1, \ldots, x_n\}$ in CNF, $n \in \mathbb{N}$.

Step 1: Formulate the instance Φ of MAX-SAT as the instance $\mathrm{LP}(\Phi)$ of the problem of linear programming.

Step 2: Solve the relaxed version of $\mathrm{LP}(\Phi)$.

Step 3: Compute $\beta_1, \ldots, \beta_n$ such that $E[Z] \le E[Z | X_1 = \beta_1, \ldots, X_n = \beta_n]$ by the strategy described in COND-PROB() and using CCP-LP to compute the conditional probabilities.

Output: An assignment $\beta_1 \ldots \beta_n$ to X.

Algorithm 5.4.5.6.

Input: A formula Φ over X in CNF.

Step 1: Compute an assignment γ to X by COND-PROB(RSMS). Estimate $I(\gamma)$:=the number of clauses of Φ satisfied by γ.

Step 2: Compute an assignment δ to X by the algorithm DER-RRRMS. Estimate $I(\delta)$:=the number of clauses of Φ satisfied by δ.

Step 3: **if** $I(\gamma) \ge I(\delta)$ **then output**(γ)

else output(δ).