400 Reject determinism, embrace probability

Francesco Virgolini

Ñifáuuuuu

Donarem un algorisme que, per qualsevol $\delta > 0$ fixat, troba la solució en temps $\mathcal{O}(n \log(n/\delta))$ i amb probabilitat d'error menor que δ.

L'algorisme és el següent:

---

input : La mida del vector $n \in \mathbb{Z}^+$ i la probabilitat d'error permesa $\delta > 0$.
output: Una estimació x del vector ocult.

$x \leftarrow (0, \dots, 0)$;
for $i \leftarrow 1$ **to** n **do**
  $c_0 \leftarrow 0$;
  $c_1 \leftarrow 0$;
  $m \leftarrow \frac{1}{4\varepsilon^2} \log(n/\delta)$;
  for $j \leftarrow 1$ **to** m **do**
    $z \leftarrow \mathrm{Unif}(\mathbb{Z}_2^n)$;
    $z_i \leftarrow 0$;
    $r_0 \leftarrow A_x(z)$;
    $z_i \leftarrow 1$;
    $r_1 \leftarrow A_x(z)$;
    if $r_0 = r_1$ **then** $c_0 \leftarrow c_0 + 1$;
    else $c_1 \leftarrow c_1 + 1$;
  if $c_0 > c_1$ **then**
    $x_i \leftarrow 0$
  else
    $x_i \leftarrow 1$

Està clar que l'algorisme farà $\mathcal{O}(nm) = \mathcal{O}(n \log(n/\delta))$ crides a A_x. Anem ara a provar la seva correctesa.

La idea de l'algorisme és, per cada posició $i \in [n]$, samplejar m vectors z uniformement a l'atzar i independents els uns dels altres, i calcular quan és $A_x(z_0)$ i $A_x(z_1)$, on z_0 és z amb la posició i-èssima canviada a 0, i z_1 és z amb la posició i-èssima canviada a 1. Si A_x ens donés sempre el valor correcte del producte escalar, tindríem que $A_x(z_0) = A_x(z_1) \iff x_i = 0$, i podríem inferir-ne el valor de x.

No obstant, tenim que A_x a vegades ens pot donar una resposta errònia. Ara bé, la probabilitat de rebre informació incorrecta la podem fitar per

$$p \leq \Pr[A_x(z_0) \text{ erroni}] + \Pr[A_x(z_1) \text{ erroni}] \leq \frac{1}{2} - 2\varepsilon$$

Així doncs, si només féssim una única iteració ($m = 1$) tindríem que la probabilitat d'error és $\leq 1/2 - 2\varepsilon$. Per reduir-la exponencialment, fem el truc típic de repetir el càlcul m vegades i quedar-nos amb la resposta que surti més vegades.

Sigui Y_j la variable indicadora de què la j-èssima repetició ens ha donat la resposta correcta. Sigui $Y = Y_1 + \cdots + Y_m$. El nostre algorisme donarà la resposta incorrecta en una certa posició si $Y < m/2$. Com cada Y_j val 1 amb probabilitat major o igual a $1/2 + 2\varepsilon$, tenim que $\Pr[Y < m/2] \leq \Pr[Z < m/2]$, on $Z \sim \mathrm{Bin}(m, 1/2 + 2\varepsilon)$. Per fitar la probabilitat d'error, utilitzem la desigualtat de Chernoff, que ens garantitza la concentració de Z al voltant de la seva mitja:

$$\Pr[Z < m/2] < \Pr[Z < \mathbb{E}[Z](1 - 4\varepsilon)] \leq e^{-4^2\varepsilon^2\mathbb{E}[Z]/2} < e^{-4\varepsilon^2 m}$$

Això ens dona una cota de la probabilitat que una certa x_i estigui malament. Fent el union-bound, tenim que la probabilitat que alguna de les x_i estigui malament és menor a $n \cdot \Pr[x_i \text{ malament}]$. Així doncs, hem de prendre un valor de m que satisfaci

$$ne^{-4\varepsilon^2 m} \leq \delta \implies -4\varepsilon^2 m \leq \log(\delta/n) \implies m \geq \frac{1}{4\varepsilon^2}\log(n/\delta)$$

Donat que ε és una constant, el nombre total de crides a A_x serà $\mathcal{O}(\log(n/\delta))$ per component de x, és a dir, $\mathcal{O}(n\log(n/\delta))$ en total.

Observem que si volguessim que l'error decreixés exponencialment en n, necessitaríem $\delta = e^{-cn}$, de manera que $m = \mathcal{O}(n)$ i el temps total seria $\mathcal{O}(n^2)$.