Pontificia Universidad Católica de Chile
Departamento de Ciencia de la Computación
IIC2283 - Diseño y Análisis de Algoritmos
Profesor: Nicolás Van Sint Jan
Ayudante: Dante Pinto

Ayudantía 8

Monte Carlo

Problema 1: Lema de Schwartz-Zippel

Sea $p(x_1, ..., x_n)$ un polinomio, no nulo, de grado k y sea A un subconjunto finito y no vacío de $\mathbb{Q}$. Demuestre que si $a_1, ..., a_n$ son elegidos de manera uniforme e independiente desde A, entonces:

$$Pr(p(a_1, ..., a_n) = 0) \leq \frac{k}{|A|}$$

Solución: La demostración de este lema se encuentra en las slides de clases.

Problema 2: Multiplicación de Matrices

Sean A, B, $C \in \mathbb{Q}^{n \times n}$. Queremos determinar si $A \cdot B = C$.

1. Diseñe un algoritmo determinista que resuelva el problema y caracterice su tiempo de ejecución.

   Solución: El algoritmo más simple que podemos utilizar es simplemente multiplicar las matrices A y B y luego comparar el resultado, entrada por entrada, con la matriz C. Considerando la suma y multiplicación de racionales como operación a contar y utilizando la forma más simple de multiplicación de matrices, este algoritmo tomará tiempo cúbico respecto a n.

2. Diseñe un algoritmo aleatorizado que resuelva el problema con un mejor tiempo que el algoritmo anterior.

   Solución: En lugar de multiplicar directamente las dos matrices podemos generar, con distribución uniforme, un vector cualquiera v, calcular $A \cdot B \cdot v - C \cdot v$ y luego comparar este valor con el vector 0.

   La multiplicación de una matriz por un vector toma tiempo cuadrático, por lo que si calculamos $B \cdot v$ y luego multiplicamos el resultado de esta operación por A, el algoritmo tomará tiempo cuadrático.

3. Calcule la probabilidad de error del algoritmo 2.

   Solución: Formalizando el algoritmo anterior, tenemos:

Function `Compare`(A, B, C)
  `generate`$(v) \in \{0, 1\}^n$
  $c = C \cdot v$
  $b = B \cdot v$
  $a = a \cdot b$
  $r = a - c$
  if $r = \vec{0}$ **then**
    return `true`
  return `false`

En caso que $r \neq \vec{0}$, es imposible que $A \cdot B = C$, sin importar qué valor tenga el vector v, por tanto el algoritmo solo cometerá errores cuando retorne `true` y $A \cdot B \neq C$, por lo que solo debemos analizar este caso.

En otras palabras, queremos calcular la probabilidad de que $r = \vec{0}$ cuando $A \cdot B \neq 0$. Si definimos $D = A \cdot B - C$, es claro que $r = D \cdot v$ y además que alguna de las entradas de D, d_{i_0,j_0} debe ser no nula, por lo que podemos acotar la probabilidad que buscamos por la probabilidad de que la entrada r_{i_0} sea 0.

El valor de r_{i_0} está dado por:

$$r_{i_0} = \sum_{i=1}^{n} d_{i,j} \cdot v_i = d_{i_0,j_0} \cdot r_{i_0} + k \quad \wedge \quad k \in \mathbb{N}$$

$$\begin{aligned} \Rightarrow Pr[r_{i_0} = 0] = Pr[r_{i_0} = 0 | k = 0] \cdot Pr[k = 0] \\ + Pr[r_{i_0} = 0 | k \neq 0] \cdot Pr[k \neq 0] \end{aligned}$$

Calculando las probabilidades que buscamos podemos ver que:

$$Pr[r_{i_0} = 0 | k = 0] = Pr[v_{i_0} = 0] = \frac{1}{2}$$
$$Pr[r_{i_0} = 0 | k \neq 0] \leq Pr[v_{i_0} = 0] = \frac{1}{2}$$

Luego:

$$\begin{aligned} Pr[r = 0] &\leq Pr[r_{i_0} = 0] \\ &\leq \frac{1}{2} \cdot Pr[k = 0] + \frac{1}{2} \cdot Pr[k \neq 0] \\ &= \frac{1}{2} \left(Pr[k = 0] + Pr[k \neq 0] \right) \\ &= \frac{1}{2} \end{aligned}$$

Y por lo tanto la probabilidad de error del algoritmo será menor a un medio.