

Quicksort e Select Aleatorizados

CLRS Secs 7.3, 7.4 e 9.2

Particione

Rearranja $A[p..r]$ de modo que $p \leq q \leq r$ e $A[p..q-1] \leq A[q] < A[q+1..r]$

PARTICIONE (A, p, r)

```
1   $x \leftarrow A[r]$        $\triangleright x$  é o "pivô"
2   $i \leftarrow p-1$ 
3  para  $j \leftarrow p$  até  $r-1$  faça
4      se  $A[j] \leq x$ 
5          então  $i \leftarrow i+1$ 
6               $A[i] \leftrightarrow A[j]$ 
7   $A[i+1] \leftrightarrow A[r]$ 
8  devolva  $i+1$ 
```

Invariantes: no começo de cada iteração de 3–6,

(i0) $A[p..i] \leq x$ (i1) $A[i+1..j-1] > x$ (i2) $A[r] = x$

Particione

Rearranja $A[p..r]$ de modo que $p \leq q \leq r$ e $A[p..q-1] \leq A[q] < A[q+1..r]$

PARTICIONE (A, p, r)

```
1   $x \leftarrow A[r]$       ▷  $x$  é o "pivô"  
2   $i \leftarrow p-1$   
3  para  $j \leftarrow p$  até  $r-1$  faça  
4      se  $A[j] \leq x$   
5          então  $i \leftarrow i+1$   
6               $A[i] \leftrightarrow A[j]$   
7   $A[i+1] \leftrightarrow A[r]$   
8  devolva  $i+1$ 
```

Consumo de tempo: $\Theta(n)$ onde $n := r - p$.

Quicksort aleatorizado

PARTICIONE-ALEA(A, p, r)

1 $i \leftarrow \text{RANDOM}(p, r)$

2 $A[i] \leftrightarrow A[r]$

3 **devolva** PARTICIONE(A, p, r)

Quicksort aleatorizado

PARTICIONE-ALEA(A, p, r)

- 1 $i \leftarrow \text{RANDOM}(p, r)$
- 2 $A[i] \leftrightarrow A[r]$
- 3 devolva PARTICIONE(A, p, r)

QUICKSORT-ALE (A, p, r)

- 1 se $p < r$
- 2 então $q \leftarrow \text{PARTICIONE-ALEA}(A, p, r)$
- 3 QUICKSORT-ALE($A, p, q - 1$)
- 4 QUICKSORT-ALE($A, q + 1, r$)

Quicksort aleatorizado

PARTICIONE-ALEA(A, p, r)

- 1 $i \leftarrow \text{RANDOM}(p, r)$
- 2 $A[i] \leftrightarrow A[r]$
- 3 **devolva** PARTICIONE(A, p, r)

QUICKSORT-ALE(A, p, r)

- 1 **se** $p < r$
- 2 **então** $q \leftarrow \text{PARTICIONE-ALE}(A, p, r)$
- 3 QUICKSORT-ALE($A, p, q - 1$)
- 4 QUICKSORT-ALE($A, q + 1, r$)

Consumo esperado de tempo?

Quicksort aleatorizado

PARTICIONE-ALEA(A, p, r)

- 1 $i \leftarrow \text{RANDOM}(p, r)$
- 2 $A[i] \leftrightarrow A[r]$
- 3 devolva PARTICIONE(A, p, r)

QUICKSORT-ALE(A, p, r)

- 1 se $p < r$
- 2 então $q \leftarrow \text{PARTICIONE-ALEA}(A, p, r)$
- 3 QUICKSORT-ALE($A, p, q - 1$)
- 4 QUICKSORT-ALE($A, q + 1, r$)

Consumo esperado de tempo?

Basta contar o número esperado de comparações na linha 4 do PARTICIONE.

Consumo esperado de tempo

Basta contar o número esperado de comparações na linha 4 do **PARTICIONE**.

```
PARTICIONE ( $A, p, r$ )
1   $x \leftarrow A[r]$        $\triangleright x$  é o “pivô”
2   $i \leftarrow p-1$ 
3  para  $j \leftarrow p$  até  $r-1$  faça
4      se  $A[j] \leq x$ 
5          então  $i \leftarrow i+1$ 
6               $A[i] \leftrightarrow A[j]$ 
7   $A[i+1] \leftrightarrow A[r]$ 
8  devolva  $i+1$ 
```

Consumo de tempo esperado

Suponha $A[p..r]$ permutação de $1..n$.

X_{ab} = número de comparações entre a e b na
linha 4 do **PARTICIONE** do QUICKSORT-ALE;

Consumo de tempo esperado

Suponha $A[p..r]$ permutação de $1..n$.

X_{ab} = número de comparações entre a e b na linha 4 do PARTICIONE do QUICKSORT-ALE;

Queremos calcular

$$\begin{aligned} X &= \text{total de comparações "A[j] \le x"} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n X_{ab} \end{aligned}$$

Consumo de tempo esperado

Suponha $A[p..r]$ permutação de $1..n$.

X_{ab} = número de comparações entre a e b na linha 4 do PARTICIONE do QUICKSORT-ALE;

Queremos calcular

$$\begin{aligned} X &= \text{total de comparações "A[j] \leq x"} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n X_{ab} \end{aligned}$$

$$E[X] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n E[X_{ab}]$$

Consumo de tempo esperado

Suponha $A[p..r]$ permutação de $1..n$.

X_{ab} = número de comparações entre a e b na linha 4 do PARTICIONE do QUICKSORT-ALE;

Queremos calcular

$$\begin{aligned} X &= \text{total de comparações "A[j] \leq x"} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n X_{ab} \end{aligned}$$

$$E[X] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n E[X_{ab}] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n \Pr\{X_{ab}=1\}$$

Consumo de tempo esperado

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \dots, b\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

Consumo de tempo esperado

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \dots, b\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

$$\Pr \{X_{ab}=1\} = \frac{1}{b-a+1} + \frac{1}{b-a+1} = E[X_{ab}]$$

Consumo de tempo esperado

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \dots, b\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

$$\Pr \{X_{ab}=1\} = \frac{1}{b-a+1} + \frac{1}{b-a+1} = E[X_{ab}]$$

$$X = \sum_{a=1}^{n-1} \sum_{b=a+1}^n X_{ab}$$

$$E[X] = \text{????}$$

Consumo de tempo esperado

$$E[X] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n \Pr\{X_{ab}=1\}$$

Consumo de tempo esperado

$$\begin{aligned} E[X] &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n \Pr\{X_{ab}=1\} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n \frac{2}{b-a+1} \end{aligned}$$

Consumo de tempo esperado

$$\begin{aligned} E[X] &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n \Pr\{X_{ab}=1\} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n \frac{2}{b-a+1} \\ &= \sum_{a=1}^{n-1} \sum_{k=1}^{n-a} \frac{2}{k+1} \end{aligned}$$

Consumo de tempo esperado

$$\begin{aligned} E[X] &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n \Pr\{X_{ab}=1\} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n \frac{2}{b-a+1} \\ &= \sum_{a=1}^{n-1} \sum_{k=1}^{n-a} \frac{2}{k+1} \\ &< \sum_{a=1}^{n-1} 2 \left(\frac{1}{1} + \frac{1}{2} + \dots + \frac{1}{n} \right) \end{aligned}$$

Consumo de tempo esperado

$$E[X] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n \Pr\{X_{ab}=1\}$$

$$= \sum_{a=1}^{n-1} \sum_{b=a+1}^n \frac{2}{b-a+1}$$

$$= \sum_{a=1}^{n-1} \sum_{k=1}^{n-a} \frac{2}{k+1}$$

$$< \sum_{a=1}^{n-1} 2 \left(\frac{1}{1} + \frac{1}{2} + \dots + \frac{1}{n} \right)$$

$$< 2n \left(\frac{1}{1} + \frac{1}{2} + \dots + \frac{1}{n} \right) < 2n(1 + \ln n)$$

CLRS (A.7), p.1060

Conclusões

O consumo de tempo esperado do algoritmo QUICKSORT-ALE é $O(n \log n)$.

Do exercício 7.4-4 do CLRS temos que

O consumo de tempo esperado do algoritmo QUICKSORT-ALE é $\Theta(n \log n)$.

Select aleatorizado

PARTICIONE-ALEA(A, p, r)

- 1 $i \leftarrow \text{RANDOM}(p, r)$
- 2 $A[i] \leftrightarrow A[r]$
- 3 **devolva** PARTICIONE(A, p, r)

SELECT-ALEA(A, p, r, k)

- 1 **se** $p < r$ **então**
- 2 $q \leftarrow \text{PARTICIONE-ALEA}(A, p, r)$
- 3 **se** $k = q - p + 1$
- 4 **então devolva** $A[q]$
- 5 **se** $k < q - p + 1$
- 6 **então** SELECT-ALEA($A, p, q - 1, k$)
- 7 **senão** SELECT-ALEA($A, q + 1, r, k - (q - p + 1)$)

Select aleatorizado

PARTICIONE-ALEA(A, p, r)

- 1 $i \leftarrow \text{RANDOM}(p, r)$
- 2 $A[i] \leftrightarrow A[r]$
- 3 devolva PARTICIONE(A, p, r)

SELECT-ALEA(A, p, r, k)

- 1 se $p < r$ então
- 2 $q \leftarrow \text{PARTICIONE-ALEA}(A, p, r)$
- 3 se $k = q - p + 1$
- 4 então devolva $A[q]$
- 5 se $k < q - p + 1$
- 6 então SELECT-ALEA($A, p, q - 1, k$)
- 7 senão SELECT-ALEA($A, q + 1, r, k - (q - p + 1)$)

Consumo esperado de tempo?

Consumo de tempo esperado

Suponha $A[p..r]$ permutação de $1..n$.

X_{ab} = número de comparações entre a e b na linha 4 do **PARTICIONE** do **SELECT-ALEA**.

Observe que X_{ab} não é a mesma de antes.

De novo, queremos calcular

$$X = \text{total de comparações "A[j] \le x"} = \sum_{a=1}^{n-1} \sum_{b=a+1}^n X_{ab}.$$

Consumo de tempo esperado

Suponha $A[p..r]$ permutação de $1..n$.

X_{ab} = número de comparações entre a e b na linha 4 do **PARTICIONE** do **SELECT-ALEA**.

Observe que X_{ab} não é a mesma de antes.

De novo, queremos calcular

$$X = \text{total de comparações "A[j] \leq x"} = \sum_{a=1}^{n-1} \sum_{b=a+1}^n X_{ab}.$$

Na verdade,

$$E[X] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n E[X_{ab}] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n \Pr\{X_{ab}=1\}$$

Consumo de tempo esperado

Vamos supor que $k = n$.

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 \end{cases}$$

Consumo de tempo esperado

Vamos supor que $k = n$.

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \dots, n\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

Consumo de tempo esperado

Vamos supor que $k = n$.

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \dots, n\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

$$\Pr\{X_{ab}=1\} = \frac{1}{n-a+1} + \frac{1}{n-a+1} = E[X_{ab}]$$

Consumo de tempo esperado

Vamos supor que $k = n$.

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \dots, n\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

$$\Pr\{X_{ab}=1\} = \frac{1}{n-a+1} + \frac{1}{n-a+1} = E[X_{ab}]$$

$$X = \sum_{a=1}^{n-1} \sum_{b=a+1}^n X_{ab}$$

Consumo de tempo esperado

Vamos supor que $k = n$.

Supondo $a < b$,

$$X_{ab} = \begin{cases} 1 & \text{se primeiro pivô em } \{a, \dots, n\} \text{ é } a \text{ ou } b \\ 0 & \text{caso contrário} \end{cases}$$

Qual a probabilidade de X_{ab} valer 1?

$$\Pr\{X_{ab}=1\} = \frac{1}{n-a+1} + \frac{1}{n-a+1} = E[X_{ab}]$$

$$X = \sum_{a=1}^{n-1} \sum_{b=a+1}^n X_{ab}$$

$$E[X] = \text{????}$$

Consumo de tempo esperado

$$\begin{aligned} E[X] &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n E[X_{ab}] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n \Pr\{X_{ab}=1\} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n \frac{2}{n-a+1} \\ &= \sum_{a=1}^{n-1} \frac{2(n-a)}{n-a+1} \\ &< \sum_{a=1}^{n-1} 2 < 2n. \end{aligned}$$

Consumo de tempo esperado

$$\begin{aligned} E[X] &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n E[X_{ab}] = \sum_{a=1}^{n-1} \sum_{b=a+1}^n \Pr\{X_{ab}=1\} \\ &= \sum_{a=1}^{n-1} \sum_{b=a+1}^n \frac{2}{n-a+1} \\ &= \sum_{a=1}^{n-1} \frac{2(n-a)}{n-a+1} \\ &< \sum_{a=1}^{n-1} 2 < 2n. \end{aligned}$$

Exercício: Refaça os cálculos para um k arbitrário.

Conclusões

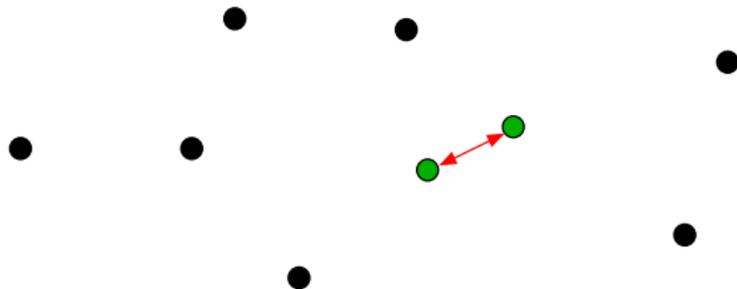
O consumo de tempo esperado do algoritmo **SELECT-ALEA** é $O(n)$.

Par de Pontos Mais Próximos

KT Sec 13.7

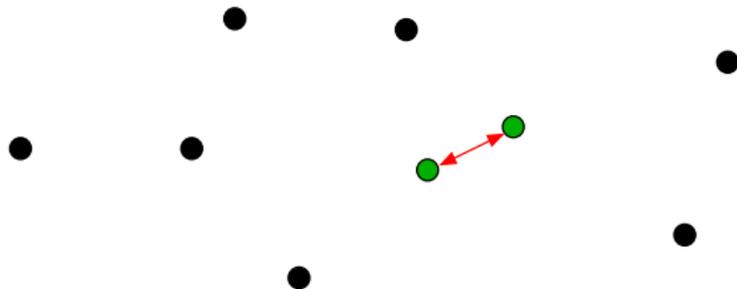
Par de pontos mais próximos

Problema: Dados n pontos no plano, determinar dois deles que estão à distância mínima.



Par de pontos mais próximos

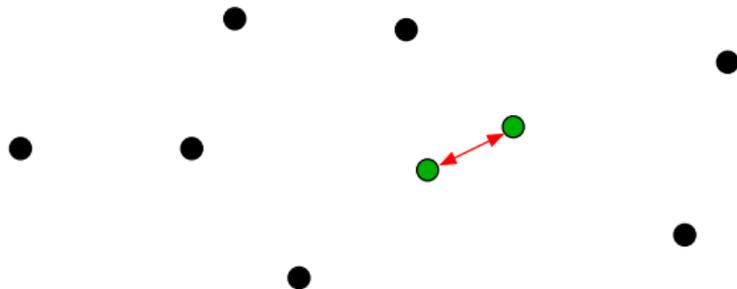
Problema: Dados n pontos no plano, determinar dois deles que estão à distância mínima.



Esta aula: algoritmo aleatorizado cujo consumo esperado de tempo é $O(n)$.

Par de pontos mais próximos

Problema: Dados n pontos no plano, determinar dois deles que estão à distância mínima.



Esta aula: algoritmo aleatorizado cujo consumo esperado de tempo é $O(n)$.

Hipótese simplificadora:

todos os pontos estão no quadrado $[0, 1] \times [0, 1]$.

Esboço do algoritmo

Problema: Dados n pontos no quadrado $[0, 1] \times [0, 1]$, determinar dois deles que estão à distância mínima.

Esboço do algoritmo

Problema: Dados n pontos no quadrado $[0, 1] \times [0, 1]$, determinar dois deles que estão à distância mínima.

Seja $\delta = \text{dist}(p_1, p_2)$ e $Q_{i,k} = \left(\frac{i\delta}{2}, \frac{(i+1)\delta}{2}\right) \times \left(\frac{k\delta}{2}, \frac{(k+1)\delta}{2}\right)$, para $i, k = 0, \dots, N$, onde $N = \lceil 2/\delta \rceil$.

Esboço do algoritmo

Problema: Dados n pontos no quadrado $[0, 1] \times [0, 1]$, determinar dois deles que estão à distância mínima.

Seja $\delta = \text{dist}(p_1, p_2)$ e $Q_{i,k} = \left(\frac{i\delta}{2}, \frac{(i+1)\delta}{2}\right) \times \left(\frac{k\delta}{2}, \frac{(k+1)\delta}{2}\right)$, para $i, k = 0, \dots, N$, onde $N = \lceil 2/\delta \rceil$.

Para $j \leftarrow 3$ até n faça:

Seja S o conjunto de pares (i, k) tais que existe um ponto p_ℓ com $\ell < j$ no quadrado $Q_{i,k}$.

Esboço do algoritmo

Problema: Dados n pontos no quadrado $[0, 1] \times [0, 1]$, determinar dois deles que estão à distância mínima.

Seja $\delta = \text{dist}(p_1, p_2)$ e $Q_{i,k} = \left(\frac{i\delta}{2}, \frac{(i+1)\delta}{2}\right) \times \left(\frac{k\delta}{2}, \frac{(k+1)\delta}{2}\right)$, para $i, k = 0, \dots, N$, onde $N = \lceil 2/\delta \rceil$.

Para $j \leftarrow 3$ até n faça:

Seja S o conjunto de pares (i, k) tais que existe um ponto p_ℓ com $\ell < j$ no quadrado $Q_{i,k}$.

Calcule (r, s) tal que p_j está em $Q_{r,s}$.

Para cada (i, k) em S tal que $|r - i| \leq 2$ e $|s - k| \leq 2$ calcule $\text{dist}(p_j, p_\ell)$ onde $\ell < j$ e $p_\ell \in Q_{i,k}$ e atualize δ quando necessário.

Esboço do algoritmo

Problema: Dados n pontos no quadrado $[0, 1] \times [0, 1]$, determinar dois deles que estão à distância mínima.

Seja $\delta = \text{dist}(p_1, p_2)$ e $Q_{i,k} = \left(\frac{i\delta}{2}, \frac{(i+1)\delta}{2}\right) \times \left(\frac{k\delta}{2}, \frac{(k+1)\delta}{2}\right)$, para $i, k = 0, \dots, N$, onde $N = \lceil 2/\delta \rceil$.

Para $j \leftarrow 3$ até n faça:

Seja S o conjunto de pares (i, k) tais que existe um ponto p_ℓ com $\ell < j$ no quadrado $Q_{i,k}$.

Calcule (r, s) tal que p_j está em $Q_{r,s}$.

Para cada (i, k) em S tal que $|r - i| \leq 2$ e $|s - k| \leq 2$ calcule $\text{dist}(p_j, p_\ell)$ onde $\ell < j$ e $p_\ell \in Q_{i,k}$ e atualize δ quando necessário.

Devolva δ .

Consumo de tempo estimado

Como fazer para o consumo esperado de tempo ser $O(n)$?

Consumo de tempo estimado

Como fazer para o consumo esperado de tempo ser $O(n)$?

Que ED usar para armazenar S ?

Consumo de tempo estimado

Como fazer para o consumo esperado de tempo ser $O(n)$?

Que ED usar para armazenar S ?

Que operações sofre S ?

Consumo de tempo estimado

Como fazer para o consumo esperado de tempo ser $O(n)$?

Que ED usar para armazenar S ?

Que operações sofre S ?

Por iteração, uma inserção e algumas consultas.

Em algumas iterações, S muda totalmente...

Consumo de tempo estimado

Como fazer para o consumo esperado de tempo ser $O(n)$?

Que ED usar para armazenar S ?

Que operações sofre S ?

Por iteração, uma inserção e algumas consultas.

Em algumas iterações, S muda totalmente...

Seria bom se...

inserções e buscas consumissem tempo (esperado) $O(1)$.

Consumo de tempo estimado

Como fazer para o consumo esperado de tempo ser $O(n)$?

Que ED usar para armazenar S ?

Que operações sofre S ?

Por iteração, uma inserção e algumas consultas.

Em algumas iterações, S muda totalmente...

Seria bom se...

inserções e buscas consumissem tempo (esperado) $O(1)$.

Que ED atinge isso?

Consumo de tempo estimado

Como fazer para o consumo esperado de tempo ser $O(n)$?

Que ED usar para armazenar S ?

Que operações sofre S ?

Por iteração, uma inserção e algumas consultas.

Em algumas iterações, S muda totalmente...

Seria bom se...

inserções e buscas consumissem tempo (esperado) $O(1)$.

Que ED atinge isso? Hashing!

Primeira tentativa

Distância(p, n)

- 1 $\delta \leftarrow \text{dist}(p_1, p_2)$
- 2 seja (i_ℓ, k_ℓ) tal que $p_\ell \in Q_{i_\ell, k_\ell}$ para $\ell = 1, 2$
- 3 **CRIE-HASHING**(H) **INSIRA**(H, i_1, k_1, p_1) **INSIRA**(H, i_2, k_2, p_2)

Primeira tentativa

Distância(p, n)

- 1 $\delta \leftarrow \text{dist}(p_1, p_2)$
- 2 seja (i_ℓ, k_ℓ) tal que $p_\ell \in Q_{i_\ell, k_\ell}$ para $\ell = 1, 2$
- 3 **CRIE-HASHING**(H) **INSIRA**(H, i_1, k_1, p_1) **INSIRA**(H, i_2, k_2, p_2)
- 4 **para** $j \leftarrow 3$ **até** n **faça**
- 5 seja (r, s) tal que $p_j \in Q_{r, s}$
- 6 **para** $t \leftarrow -2$ **até** 2 **faça**
- 7 **para** $u \leftarrow -2$ **até** 2 **faça**
- 8 se **PERTENCE**($H, r + t, s + u$)

Primeira tentativa

Distância(p, n)

- 1 $\delta \leftarrow \text{dist}(p_1, p_2)$
- 2 seja (i_ℓ, k_ℓ) tal que $p_\ell \in Q_{i_\ell, k_\ell}$ para $\ell = 1, 2$
- 3 **CRIE-HASHING**(H) **INSIRA**(H, i_1, k_1, p_1) **INSIRA**(H, i_2, k_2, p_2)
- 4 **para** $j \leftarrow 3$ **até** n **faça**
- 5 seja (r, s) tal que $p_j \in Q_{r, s}$
- 6 **para** $t, u \leftarrow -2$ **até** 2 **faça**
- 7 se **PERTENCE**($H, r + t, s + u$)

Primeira tentativa

Distância(p, n)

- 1 $\delta \leftarrow \text{dist}(p_1, p_2)$
- 2 seja (i_ℓ, k_ℓ) tal que $p_\ell \in Q_{i_\ell, k_\ell}$ para $\ell = 1, 2$
- 3 **CRIE-HASHING**(H) **INSIRA**(H, i_1, k_1, p_1) **INSIRA**(H, i_2, k_2, p_2)
- 4 **para** $j \leftarrow 3$ **até** n **faça**
- 5 seja (r, s) tal que $p_j \in Q_{r, s}$
- 6 **para** $t, u \leftarrow -2$ **até** 2 **faça**
- 7 se **PERTENCE**($H, r + t, s + u$)
- 8 então seja $p_\ell \in Q_{r+t, s+u}$ com $\ell < j$
- 9 se $\delta > \text{dist}(p_j, p_\ell)$ então $\delta \leftarrow \text{dist}(p_j, p_\ell)$

Primeira tentativa

Distância(p, n)

- 1 $\delta \leftarrow \text{dist}(p_1, p_2)$
- 2 seja (i_ℓ, k_ℓ) tal que $p_\ell \in Q_{i_\ell, k_\ell}$ para $\ell = 1, 2$
- 3 **CRUE-HASHING**(H) **INSIRA**(H, i_1, k_1, p_1) **INSIRA**(H, i_2, k_2, p_2)
- 4 **para** $j \leftarrow 3$ **até** n **faça**
- 5 seja (r, s) tal que $p_j \in Q_{r, s}$
- 6 **para** $t, u \leftarrow -2$ **até** 2 **faça**
- 7 se **PERTENCE**($H, r + t, s + u$)
- 8 então seja $p_\ell \in Q_{r+t, s+u}$ com $\ell < j$
- 9 se $\delta > \text{dist}(p_j, p_\ell)$ então $\delta \leftarrow \text{dist}(p_j, p_\ell)$
- 10 se δ foi alterado nessa iteração
- 11 então **RECONSTRUA**(H, p, j)
- 12 senão **INSIRA**(H, r, s, p_j)
- 13 **devolva** δ

Primeira tentativa

Distância(p, n)

- 1 $\delta \leftarrow \text{dist}(p_1, p_2)$
- 2 seja (i_ℓ, k_ℓ) tal que $p_\ell \in Q_{i_\ell, k_\ell}$ para $\ell = 1, 2$
- 3 **CRUE-HASHING**(H) **INSIRA**(H, i_1, k_1, p_1) **INSIRA**(H, i_2, k_2, p_2)
- 4 **para** $j \leftarrow 3$ até n **faça**
- 5 seja (r, s) tal que $p_j \in Q_{r, s}$
- 6 **para** $t, u \leftarrow -2$ até 2 **faça**
- 7 se **PERTENCE**($H, r + t, s + u$)
- 8 então seja $p_\ell \in Q_{r+t, s+u}$ com $\ell < j$
- 9 se $\delta > \text{dist}(p_j, p_\ell)$ então $\delta \leftarrow \text{dist}(p_j, p_\ell)$
- 10 se δ foi alterado nessa iteração
- 11 então **RECONSTRUA**(H, p, j)
- 12 senão **INSIRA**(H, r, s, p_j)
- 13 **devolva** δ

Consumo de tempo esperado: $O(n)$ exceto pela linha 11.

Versão final

Distância(p, n)

- 1 **EMBARALHE**(p, n) ▷ permutação aleatória dos pontos dados
- 2 $\delta \leftarrow \text{dist}(p_1, p_2)$
- 3 seja (i_ℓ, k_ℓ) tal que $p_\ell \in Q_{i_\ell, k_\ell}$ para $\ell = 1, 2$
- 4 **CRIE-HASHING**(H) **INSIRA**(H, i_1, k_1, p_1) **INSIRA**(H, i_2, k_2, p_2)
- 5 **para** $j \leftarrow 3$ **até** n **faça**
- 6 seja (r, s) tal que $p_j \in Q_{r, s}$
- 7 **para** $t, u \leftarrow -1$ **até** 1 **faça**
- 8 se **PERTENCE**($H, r + t, s + u$)
- 9 então seja $p_\ell \in Q_{r+t, s+u}$ com $\ell < j$
- 10 se $\delta < \text{dist}(p_j, p_\ell)$ então $\delta \leftarrow \text{dist}(p_j, p_\ell)$
- 11 se δ foi alterado nessa iteração
- 12 então **RECONSTRUA**(H, p, j)
- 13 senão **INSIRA**(H, r, s, p_j)
- 14 **devolva** δ

Versão final

Distância(p, n)

- 1 **EMBARALHE**(p, n) ▷ permutação aleatória dos pontos dados
- 2 $\delta \leftarrow \text{dist}(p_1, p_2)$
- 3 seja (i_ℓ, k_ℓ) tal que $p_\ell \in Q_{i_\ell, k_\ell}$ para $\ell = 1, 2$
- 4 **CRIE-HASHING**(H) **INSIRA**(H, i_1, k_1, p_1) **INSIRA**(H, i_2, k_2, p_2)
- 5 **para** $j \leftarrow 3$ **até** n **faça**
- 6 seja (r, s) tal que $p_j \in Q_{r, s}$
- 7 **para** $t, u \leftarrow -1$ **até** 1 **faça**
- 8 se **PERTENCE**($H, r + t, s + u$)
- 9 então seja $p_\ell \in Q_{r+t, s+u}$ com $\ell < j$
- 10 se $\delta < \text{dist}(p_j, p_\ell)$ então $\delta \leftarrow \text{dist}(p_j, p_\ell)$
- 11 se δ foi alterado nessa iteração
- 12 então **RECONSTRUA**(H, p, j)
- 13 senão **INSIRA**(H, r, s, p_j)
- 14 **devolva** δ

Qual é o consumo de tempo esperado?

Consumo de tempo

Seja X o número de inserções em H .

O consumo de tempo é proporcional a $E[X]$.

Consumo de tempo

Seja X o número de inserções em H .

O consumo de tempo é proporcional a $E[X]$.

Seja X_j a variável binária que vale 1 sse a linha 12 foi executada na iteração j .

$$X = \sum_{j=1}^n (1 + (j-1)X_j) \leq n + \sum_{j=1}^n j X_j$$

Consumo de tempo

Seja X o número de inserções em H .

O consumo de tempo é proporcional a $E[X]$.

Seja X_j a variável binária que vale 1 sse a linha 12 foi executada na iteração j .

$$X = \sum_{j=1}^n (1 + (j-1)X_j) \leq n + \sum_{j=1}^n j X_j$$

Mas então

$$E[X] \leq n + \sum_{j=1}^n j E[X_j] = n + \sum_{j=1}^n j \Pr\{X_j = 1\}.$$

Consumo de tempo

Seja X o número de inserções em H .

O consumo de tempo é proporcional a $E[X]$.

Seja X_j a variável binária que vale 1 sse a linha 12 foi executada na iteração j .

$$X = \sum_{j=1}^n (1 + (j-1)X_j) \leq n + \sum_{j=1}^n j X_j$$

Mas então

$$E[X] \leq n + \sum_{j=1}^n j E[X_j] = n + \sum_{j=1}^n j \Pr\{X_j = 1\}.$$

Note que $\Pr\{X_j = 1\} \leq 2/j$, logo $E[X] \leq n + 2n = 3n$.