Análise probabilística

CLRS 5.1, 5.2, C.1 a C.3, 7.1 e 7.2

Máximo

Problema: Encontrar o elemento máximo de um vetor $A[1 \mathinner{.\,.} n]$ de números inteiros positivos distintos.

```
MAX (A, n)
1  max ← 0
2  para i ← 1 até n faça
3       se A[i] > max
4            então max ← A[i]
5  devolva max
```

Quantas vezes a linha 4 é executada?
Melhor caso, pior caso, caso médio?

Suponha que $A[1 \mathinner{.\,.} n]$ é permutação aleatória uniforme de $1, \ldots, n$.

Cada permutação tem probabilidade $1/n!$.

Um pouco de probabilidade

$(S, \Pr)$ espaço de probabilidade
$S =$ conjunto finito (eventos elementares)
$\Pr\{\} =$ função de S em $[0, 1]$ tal que
(distribuição de probabilidades)

p1. $\Pr\{s\} \geq 0$, *e*
p2. $\Pr\{S\} = 1$, *onde* $\Pr\{U\}$ *é abreviação de* $\sum_{u \in U} \Pr\{u\}$.

Note que

p3. $R, T \subseteq S, R \cap T = \emptyset \Rightarrow \Pr\{R \cup T\} = \Pr\{R\} + \Pr\{T\}$.

No problema do máximo:

- S é o conjunto das permutações dos números de 1 a n;
- na distribuição uniforme, para cada $s \in S$, $\Pr\{s\} = 1/n!$.

Mais um pouco de probabilidade

Um evento é um subconjunto de S.

No problema do máximo,
eventos são subconjuntos de permutações de 1 a n.

Exemplo.

$$U := \{\text{permutações } A \text{ de } 1 \text{ a } n \text{ em que } A[n] \text{ é máximo}\}$$

é um evento de S.

Se $\Pr\{\}$ é distribuição uniforme, então

$$\Pr\{U\} = ???.$$

Se $\Pr\{\}$ é distribuição uniforme, então

$$\Pr\{U\} = \frac{1}{n}.$$

Mais um pouco de probabilidade

Uma variável aleatória é uma função númerica definida sobre os eventos elementares.

Exemplo de variável aleatória

$$X(A) := \text{número de execuções da linha 4 em MAX}(A, n)$$

"$X = k$" é uma abreviação de $\{s \in S : X(s) = k\}$

Esperança $\mathrm{E}[X]$ de uma variável aleatória X

$$\mathrm{E}[X] = \sum_{k \in X(S)} k \cdot \Pr\{X = k\} = \sum_{s \in S} X(s) \cdot \Pr\{s\}$$

Linearidade da esperança: $\mathrm{E}[\alpha X + Y] = \alpha \mathrm{E}[X] + \mathrm{E}[Y]$

Variedade aleatória binária

É uma variável aleatória cujos valores são só 0 e 1.

Fácil calcular a esperança:

$$\begin{aligned} E(X) &= \sum_{\{x \in S \mid X(x)=0\}} 0.\Pr(x) + \sum_{\{x \in S \mid X(x)=1\}} 1.\Pr(x) \\ &= \sum_{\{x \in S \mid X(x)=1\}} \Pr(x) \\ &= \Pr[X = 1]. \end{aligned}$$

De volta ao máximo

Problema: Encontrar o elemento máximo de um vetor $A[1..n]$ de números inteiros distintos.

```
MAX (A, n)
1  max ← 0
2  para i ← 1 até n faça
3      se A[i] > max
4          então max ← A[i]
5  devolva max
```

Quantas vezes a linha 4 é executada no caso médio?

Suponha que
$A[1..n]$ é permutação aleatória uniforme de $1, \ldots, n$.

Cada permutação tem probabilidade $1/n!$.

Exemplos

$A[1..2]$	linha 4
1,2	2
2,1	1
$\mathrm{E}[X]$	3/2

$A[1..3]$	linha 4
1,2,3	3
1,3,2	2
2,1,3	2
2,3,1	2
3,1,2	1
3,2,1	1
$\mathrm{E}[X]$	11/6

Mais um exemplo

$A[1\,..\,4]$	linha 4	$A[1\,..\,4]$	linha 14
1,2,3,4	4	3,1,2,4	2
1,2,4,3	3	3,1,4,2	2
1,3,2,4	3	3,2,1,4	2
1,3,4,2	3	3,2,4,1	2
1,4,2,3	2	3,4,1,2	2
1,4,3,2	2	3,4,2,1	2
2,1,3,4	3	4,1,2,3	1
2,1,4,3	2	4,1,3,2	1
2,3,1,4	3	4,2,1,3	1
2,3,4,1	3	4,2,3,1	1
2,4,1,3	2	4,3,1,2	1
2,4,3,1	2	4,3,2,1	1
		$\mathrm{E}[X]$	50/24

Variáveis aleatórias

$$X = \text{número total de execuções da linha 4}$$

$$X_i = \begin{cases} 1 & \text{se “}max \leftarrow A[i]\text{” é executado} \\ 0 & \text{caso contrário} \end{cases}$$

$$\begin{aligned} X &= \text{número total de execuções da linha 4} \\ &= X_1 + \cdots + X_n \end{aligned}$$

Esperanças:

$$\begin{aligned} \mathrm{E}[X_i] &= \text{probabilidade de que } A[i] \text{ seja máximo em } A[1\,..\,i] \\ &= 1/i \end{aligned}$$

Esperança

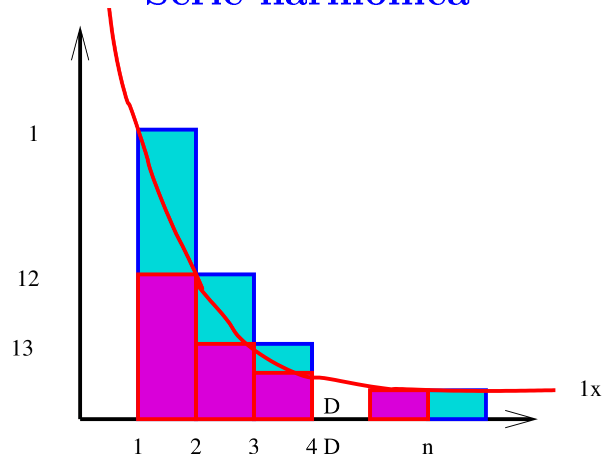
$$\begin{aligned} \mathrm{E}[X] &= \mathrm{E}[X_1 + \cdots + X_n] \\ &= \mathrm{E}[X_1] + \cdots + \mathrm{E}[X_n] \\ &= 1/1 + \cdots + 1/n \\ &< 1 + \ln n \\ &= \Theta(\lg n) \end{aligned}$$

$$2.92 < \frac{1}{1} + \cdots + \frac{1}{10} < 2.93 < 3.30 < 1 + \ln 10$$

$$5.18 < \frac{1}{1} + \cdots + \frac{1}{100} < 5.19 < 6.60 < 1 + \ln 100$$

$$9.78 < \frac{1}{1} + \cdots + \frac{1}{10000} < 9.79 < 10.21 < 1 + \ln 10000$$

Série harmônica



$$\ln n = \int_1^n \frac{dx}{x} < H_n = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots + \frac{1}{n}$$

$$< 1 + \int_1^n \frac{dx}{x} = 1 + \ln n.$$

Constante de Euler

Ou de Euler-Mascheroni.

$$\gamma = \lim_{n\to\infty} (H_n - \ln n)$$
$$= 0.5772156649\ldots$$

Experimentos

Para cada valor de $n = 252, 512, 1024, \ldots$
foram geradas 10, 100 ou 200 amostras de
sequências de inteiros através do trecho de código

```
for (i = 0; i < n; i++){
  v[i]=(int)((double)INT_MAX*rand()/(RAND_MAX+1.0));
}
```

onde `rand()` é a função geradora de números (pseudo-)aleatórios da biblioteca do `C`.

A coluna $\mathrm{E}[\hat{X}]$ nas tabelas a seguir mostra o número médio de vezes que a linha 4 do algoritmo MAX foi executada para cada valor de n e cada amostra de sequências.

Experimentos (10)

n	$E[\hat{X}]$	$1 + \ln n$
256	7.20	6.55
512	6.90	7.24
1024	7.30	7.93
2048	7.10	8.62
4096	10.20	9.32
8192	9.00	10.01
16384	10.80	10.70
32768	11.00	11.40
65536	12.50	12.09
131072	12.60	12.78
262144	13.20	13.48
524288	13.20	14.17
1048576	12.80	14.86
2097152	13.90	15.56
4194304	14.90	16.25
8388608	17.90	16.94

Experimentos (100)

n	$E[\hat{X}]$	$1 + \ln n$
256	5.92	6.55
512	6.98	7.24
1024	7.55	7.93
2048	8.39	8.62
4096	8.97	9.32
8192	9.26	10.01
16384	10.44	10.70
32768	11.32	11.40
65536	11.66	12.09
131072	12.38	12.78
262144	13.17	13.48
524288	13.56	14.17
1048576	14.54	14.86
2097152	15.10	15.56
4194304	15.61	16.25
8388608	16.56	16.94

Experimentos (200)

n	$E[\hat{X}]$	$1 + \ln n$
256	6.12	6.55
512	6.86	7.24
1024	7.38	7.93
2048	7.96	8.62
4096	8.87	9.32
8192	9.41	10.01
16384	10.28	10.70
32768	10.92	11.40
65536	11.31	12.09
131072	12.37	12.78
262144	12.92	13.48
524288	13.98	14.17
1048576	14.19	14.86
2097152	15.62	15.56
4194304	15.74	16.25
8388608	17.06	16.94

De volta ao Quicksort

Rearranja $A[p \mathinner{.\,.} r]$ em ordem crescente.

```
QUICKSORT (A, p, r)
se p < r
    então q ← PARTICIONE (A, p, r)
          QUICKSORT (A, p, q − 1)
          QUICKSORT (A, q + 1, r)
```

O consumo de tempo do QUICKSORT é $\mathrm{O}(n^2)$ e $\Omega(n \lg n)$.

Por que ele é melhor na prática que outros algoritmos que têm consumo de tempo $\Theta(n \lg n)$?

Particione

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

```
PARTICIONE (A, p, r)
1  x ← A[r]        ▷ x é o “pivô”
2  i ← p−1
3  para j ← p até r faça
4      se A[j] ≤ x
5          então i ← i + 1
6                A[i] ↔ A[j]
7  devolva i
```


Análise de caso médio

Considere que $A[1 \mathinner{.\,.} n]$ é permutação escolhida uniformemente dentre todas as permutações de 1 a n.

Seja $X(A)$ o número de vezes que a linha 4 do PARTICIONE é executada para uma chamada de QUICKSORT$(A, 1, n)$.

Observe que X é uma variável aleatória.

Uma maneira de estimarmos o consumo de tempo médio do QUICKSORT é calcularmos $\mathrm{E}[X]$.

Ideia: Escrever X como soma de variáveis aleatórias binárias, cuja esperança é mais fácil de calcular.

Quem serão essas variáveis aleatórias binárias?

Exemplo

1	3	6	2	5	7	4
1	3	2	4	5	7	6
1	2	3	4	5	6	7

	1	2	3	4	5	6	7
1		1	0	1	0	0	0
2	1		1	1	0	0	0
3	0	1		1	0	0	0
4	1	1	1		1	1	1
5	0	0	0	1		1	0
6	0	0	0	1	1		1
7	0	0	0	1	0	1	

Consumo de tempo esperado

Para a e b em $\{1, \ldots, n\}$, seja

X_{ab} = número de comparações entre a e b na linha 4 de PARTICIONE.

Queremos calcular

$$X = \text{total de execuções da linha 4 do PARTICIONE} = \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} X_{ab}$$

Consumo de tempo esperado

Supondo $a < b$,

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

Qual a probabilidade de X_{ab} valer 1?

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

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

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

Consumo de tempo esperado

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

Conclusões

O consumo de tempo esperado do QUICKSORT, quando sua entrada é uma permutação de $1, \ldots, n$ escolhida uniformemente, é $\mathrm{O}(n \log n)$.

Aula que vem:

Quicksort aleatorizado e sua análise.