PAA - Aula 08 bônus

Bases de Probabilidade, Problema da Seleção

Espaço amostral: U é o conjunto de todos os resultados possíveis.

- Cada resultado r em U tem uma probabilidade $Pr(r) \geq 0$
  - e $\sum_{r \in U} Pr(r) = 1$

Eventos: um evento é um subconjunto $S \subseteq U$ e $Pr(S) = \sum_{r \in S} Pr(r)$

Variáveis aleatórias: uma variável aleatória X é uma função $X: U \to \mathbb{R}$

Cenário 1: resultado da rolagem de 1 dado.

- $U = \{1,2,3,4,5,6\}$ e $Pr(r) = 1/6 \ \forall r \in U$
- Evento → Resultado ser par, i.e., S = $\{2, 4, 6\}$. Pr(S) = $\sum_{r \in S} Pr(r) = |S| \cdot 1/6 = 3/6 = 1/2$
- Variável → X é 1 se o resultado é par.

Cenário 2: resultado da rolagem de 2 dados.

- $U = \{(1,1),(1,2),(1,3),\dots,(6,5),(6,6)\}$ e $Pr(r) = 1/36 \ \forall r \in U$
- Evento → Soma dos resultados ser 7, i.e., S = $\{(1,6),(2,5),(3,4),(4,3),(5,2),(6,1)\}$
  - Pr(S) = $\sum_{r \in S} Pr(r) = |S| \, 1/36 = 6/36 = 1/6$
- Variável → Y é a soma dos resultados dos dados.

Esperança: a esperança de uma variável aleatória X : U → R é

- a média de seus valores ponderada pelas probabilidades dos mesmos,
    - i.e., E[X] = $\sum_{r \in U} X(r) P_r(r)$

Linearidade da esperança: sendo X e Y variáveis aleatórias definidas sobre U,

- a linearidade da esperança diz que $E[X+Y] = E[X] + E[Y]$
- Demonstração: $E[X+Y] = \sum_{r \in U} (X(r)+Y(r)) P_r(r) = \sum_{r \in U} X(r) P_r(r) + \sum_{r \in U} Y(r) P_r(r) = E[X] + E[Y]$
- Note que, a propriedade vale mesmo que as variáveis sejam dependentes.
- Quiz1: generalizar para a soma de n variáveis X1, X2, X3, … , Xn.

Cenário 1: resultado da rolagem de 1 dado. U = {1, 2, 3, 4, 5, 6} e

- Pr(r) = 1/6 para r em U. X é 1 se o resultado é par.
- E[X] = $\sum_{r=1}^{6} X(r) P_r(r) = \sum_{r=1}^{6} X(r) \frac{1}{6} = \frac{1}{6}(\underset{0}{X(1)} + \underset{1}{X(2)} + \underset{0}{X(3)} + \underset{1}{X(4)} + \underset{0}{X(5)} + \underset{1}{X(6)}) = 3/6 = 1/2$

Cenário 2: resultado da rolagem de 2 dados. U = {(1,2), (1,3), … , (6,5), (6,6)} e

- Pr(r) = 1/36 para r em U. Y é a soma dos resultados dos dois dados.
- Seja Y1 e Y2 o resultado de cada dado. Note que Y = $Y1 + Y2$
- Temos que E[Y1] = E[Y2] = $\sum_{r=1}^{6} Y2(r) P_r(r) = \sum_{r=1}^{6} r \cdot 1/6 = 1/6(1+2+3+4+5+6) = \frac{21}{6} = 3,5$
- Assim, E[Y] = $E[Y1+Y2] = E[Y1] + E[Y2] = 3,5 + 3,5 = 7$

↳ usando a linearidade da esperança

Problema da seleção

No problema da seleção, dado um vetor com n elementos em qualquer ordem, queremos determinar o k-ésimo menor elemento.

- Este problema generaliza o problema de encontrar a mediana de um vetor.

Podemos resolver este problema reduzindo-o ao problema da ordenação, ou seja, ordenamos o vetor e devolvemos o elemento que terminou na posição k.

- Como conhecemos algoritmos de ordenação que levam tempo O(n log n), esta abordagem também tem essa complexidade de tempo.

Mas, será que conseguimos fazer melhor que isso?

- Afinal, ao menos intuitivamente, ordenar parece bem trabalhoso do que encontrar o elemento que ocupa uma certa posição na ordem.

Como motivação podemos considerar um caso particular deste problema,

- que é encontrar o mínimo ou o máximo. Sabemos fazer isso em tempo linear.

Será que conseguimos usar divisão e conquista

- para resolver o problema da seleção em tempo linear?

Nos inspirando no algoritmo quickSort, vamos usar o algoritmo particiona,

- que resolve o problema da separação em tempo linear,
- e a ideia de escolher um pivô aleatoriamente.

```
selecao(vetor v, int k):
        se tam(v) == k: devolva v[k]
        p = escolhePivo(v)
        (vl, vr) = particiona(v, p)
        m = tam(vl) + 1
        se k = m: devolva v[m]
        se k < m: devolva selecao(vl, k)
        senão /* k > m */ devolva selecao(vr, k-m)
```

Note que, no algoritmo anterior, supomos que k está no intervalo [1, tam(v)].

Prova de corretude: Basta aplicar indução matemática no tamanho do subproblema. Fica como exercício.

Análise de eficiência do algoritmo recursivo para seleção

Qual a eficiência de pior caso deste algoritmo? Ou seja, qual o pior tipo de pivô que podemos escolher?

- R.: O menor elemento (ou o maior). Neste caso, a cada chamada recursiva eliminamos apenas um elemento.
- Qual a eficiência neste caso? R: Theta(n^2).
- Isso porque o particiona percorre todo o vetor em cada chamada. Assim, no total temos n + n-1 + n-2 + ... + 3 + 2 + 1 = n(n-1)/2 iterações do particiona.

Mas este caso é bastante raro se escolhermos o pivô aleatoriamente.

Qual a eficiência no melhor caso deste algoritmo? I.e., qual o melhor tipo de pivô?

- R.: Se desconsideramos a sorte de escolhermos justamente o k-ésimo elemento como pivô, o melhor tipo de pivô é o que divide o vetor ao meio.
- Neste caso especial teríamos a seguinte recorrência: $T(n) = T(n/2) + O(n)$
    - A que função de tempo essa recorrência corresponde?

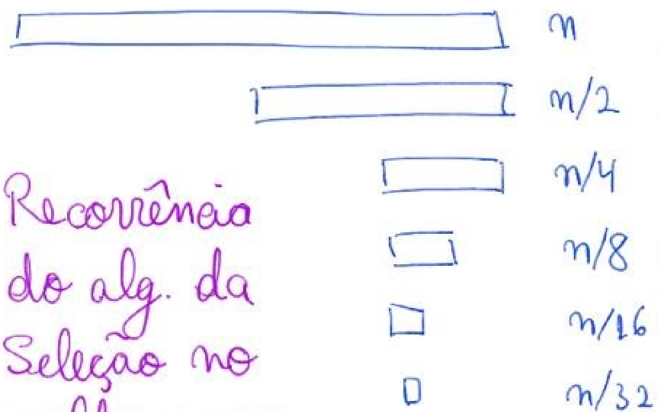
Lembrando do Teorema mestre: $T(n) = a\,T(n/b) + c\,n^d$

1. $a = b^d \rightarrow T(n) = O(n^d \log n)$
2. $a < b^d \rightarrow T(n) = O(n^d)$
3. $a > b^d \rightarrow T(n) = O(n^{\log_b a})$

Assim, para $T(n) = T(n/2) + O(n)$ temos $a = 1$, $b = 2$ e $d = 1$.

- Como $a = 1 < 2^1 = b^d$ caímos no caso 2 do Teorema Mestre.
- Portanto, $T(n) = O(n^1) = O(n)$, i.e., tempo linear no tamanho da entrada.

$T(n) = T(n/2) + O(n)$



Recorrência do alg. da Seleção no melhor caso

Qual a eficiência esperada para este algoritmo?

- Para esta análise vamos usar o conceito de fase.
- Dizemos que o algoritmo selecao está na fase j se tam,
    - o tamanho do vetor atual, está em n*(3/4)^(j+1) < tam <= n*(3/4)^j.
    - Note que, no início o algoritmo está na fase 0.
- Seja X_j uma variável aleatória que indica
    - o número de chamadas recursivas realizadas pelo algoritmo na fase j.
- O trabalho não recursivo que o algoritmo selecao faz é <= c * tam,
    - já que a subrotina particiona leva tempo linear.
- Assim, podemos contabilizar o trabalho total do algoritmo selecao como

  T(n) <= \sum_{j \in fases} X_j * c * tamanho atual do vetor
  <= \sum_{j \in fases} X_j * c * n*(3/4)^j
  = cn \sum_{j \in fases} X_j * (3/4)^j

Mas, quanto deve durar a fase j?

- Definimos um bom pivô como aquele que gera uma partição 25%-75%,
    - ou seja, que quebra o vetor corrente em dois de forma que
        - nenhum é menor que 25% ou maior que 75%.
- Note que, metade dos elementos do vetor corrente são bons pivôs, ou seja,
    - a probabilidade de escolher um bom pivô aleatoriamente é 50% = ½.
- Note também que a fase sempre termina quando escolhemos um bom pivô.
    - Quiz2: Por que?

Agora, considere a variável geométrica Y com probabilidade de sucesso igual a 1/2.

- Variável geométrica é aquela que contabiliza o número de lances de uma "moeda" com probabilidade de sucesso p até obter o primeiro sucesso.
- Podemos calcular a esperança de Y como segue

  $E[Y] = 1 + 1/2*E[Y] \rightarrow E[Y] - 1/2*E[Y] = 1 \rightarrow 1/2E[Y] = 1 \rightarrow E[Y] = 2$

- Por curiosidade, no geral a esperança de uma variável geométrica
  - com probabilidade de sucesso p é 1/p.

Como uma fase sempre termina quando um pivô bom é escolhido

- e a probabilidade de um pivô bom ser escolhido é 1/2, temos que
  - a esperança de Y é um limitante superior para a esperança de X_j,
    - i.e., para todo j temos E[X_j] <= E[Y] = 2.

- Por que limitante superior? R: Porque uma fase também pode terminar com uma divisão ruim. Quiz3: Como?

Voltando à nossa equação principal do tempo de execução

E[T(n)] <= E[c*n \sum_{j \in fases} X_j (3/4)^j]
(usando linearidade da esperança)
<= c*n \sum_{j=0^+\inf} E[X_j] (3/4)^j
<= c*n \sum_{j=0^+\inf} E[Y] (3/4)^j
<= c*n \sum_{j=0^+\inf} 2 (3/4)^j
<= 2*c*n \sum_{j=0^+\inf} (3/4)^j
<= 2*c*n 1/(1-(3/4))
= 2*c*n 4/(4-3)
= 8*c*n = O(n)

sendo que a última desigualdade deriva da fórmula para

- a soma dos termos de uma PG de razão menor que 1.

$$qS_\infty = \sum_{j=0}^{\infty} q^{j+1}$$

Soma dos termos de uma Progressão

$$S_\infty = \sum_{j=0}^{\infty} q^j$$

$$\Rightarrow (1-q)S_\infty = 1 \Rightarrow S_\infty = 1/(1-q)$$

Geométrica de razão menor que 1