ANÁLISE DO ALGORITMO LINEAR DE SELEÇÃO (Cormen, §9.3)

A seguinte é uma implementação do algoritmo de seleção de Blum et al. (1973), explicado em http://www.lia.ufc.br/~pmsf/2015-1/cana/arquivos/aula_selecao.pdf :

```
==============================================================================
Algoritmo: sel_linear
Entrada:   vetor v[1..n], índices "i", "p", "f".
Pré-cond.: 1 <= i <= p <= f <= n.
Saída:     nenhuma.
Pós-cond.: (1) v[i..f] é uma permutação dos elementos inicialmente em v[i..f].
           (2) Os elementos em v[1..i-1] e em v[f+1..n] não são alterados.
           (3) ∀j ∈ {i..p-1}, v[j] <= v[p].
           (4) ∀j ∈ {p+1..f}, v[p] <= v[j].
------------------------------------------------------------------------------
01. s := ⌈(f-i+1)/5⌉ /* número de segmentos */
02. SE s = 1
03. | insertion_sort (v[1..n], i, f)
04. | RETORNE.
05. a := i, c := i
06. REPITA
07. | b := mín {a+4, f} , m := ⌊(a+b)/2⌋
08. | insertion_sort (v[1..n], a, b)
09. | temp := v[c] , v[c] := v[m] , v[m] := temp
10. | a := a + 5 , ++c
11. ENQUANTO a <= f
12. um := i +s -1 /* última mediana */ , mm := ⌊(i + um)/2⌋
13. sel_linear (v[1..n], i, mm, um)
14. q := particionar (v[1..n], i, mm, f)
15. SE p = q
16. | RETORNE.
17. SE p < q
18. | sel_linear (v[1..n], i, p, q-1)
19. SENÃO
20. | sel_linear (v[1..n], q+1, p, f)
==============================================================================
```

TEMPO DE EXECUÇÃO DAS PARTES DO ALGORITMO

Nós mostraremos que, se os elementos em v[i..f] forem todos distintos, então o algoritmo acima executa em tempo linear com relação ao tamanho "t = f-i+1" do intervalo v[i..f]. Para tanto, observe primeiramente que:

(1) As linhas 1--5, 12, 15--17 e 19 executam em tempo constante, isto é, que pode ser limitado superiormente independentemente do valor de "t".

(2) Tanto o laço das linhas 6--11 quanto o particionamento da linha 14 executam em tempo linear com relação a "t".

(3) A chamada recursiva da linha 13 é feita sobre um intervalo de tamanho "s = ⌈t/5⌉".

(4) As chamadas recursivas das linhas 18 e 20 são feitas sobre intervalos de tamanho no máximo "t -3*⌊ ⌈t/5⌉/2 ⌋ +2".

Para verificar a última observação acima, observe primeiramente:

LEMA 1. Quando "$s \geq 2$", há pelo menos "$\lfloor s/2 \rfloor - 1$" segmentos, de exatamente 5 elementos, cujas medianas são menores que a mediana das medianas, e o mesmo vale trocando-se "menores" por "maiores".

PROVA:

Consideremos os casos possíveis com relação à paridade de "um - i":

1. Se "um = i + 2k", para algum natural "k", então

   $$\begin{aligned} mm &= \lfloor (i + um)/2 \rfloor \\ &= \lfloor (i + i + 2k)/2 \rfloor \\ &= \lfloor (2i + 2k)/2 \rfloor \\ &= \lfloor i + k \rfloor \\ &= i + k. \end{aligned}$$

   Nesse caso, como as medianas menores que a mediana das medianas são aquelas no intervalo v[i..mm-1], então o número dessas medianas é

   $$\begin{aligned} (mm-1) - i + 1 &= mm - i \\ &= i + k - i \\ &= k, \end{aligned}$$

   e todas essas medianas são medianas de segmentos de exatamente 5 elementos. Como o número de segmentos é "$s = um - i + 1 = 2k + 1$", então sabemos que o número de segmentos, de exatamente 5 elementos, cujas medianas são menores que a mediana das medianas é

   $$\begin{aligned} k &> k - 1 \\ &= \lfloor (2k+1)/2 \rfloor - 1 \\ &= \lfloor s/2 \rfloor - 1, \end{aligned}$$

   como desejado. Com relação, agora, aos segmentos cujas medianas são maiores que a mediana das medianas, observe que as medianas maiores que v[mm] são aquelas no intervalo v[mm+1..um]. A última dessas medianas pode, porém, ser de um segmento com menos que 5 elementos. Em todo caso, as medianas no intervalo v[mm+1..um-1] certamente são de segmentos de 5 elementos, e o número de medianas nesse intervalo é

   $$\begin{aligned} (um-1) - (mm+1) + 1 &= um - 1 - mm - 1 + 1 \\ &= um - 1 - mm \\ &= (i + 2k) - 1 - (i + k) \\ &= i + 2k - 1 - i - k \\ &= k - 1 \\ &= \lfloor (2k+1)/2 \rfloor - 1 \\ &= \lfloor s/2 \rfloor - 1, \end{aligned}$$

   como desejado.

2. Se "um = i + 2k + 1", para algum natural "k", então o resultado pode ser demonstrado de forma análoga ao caso anterior (veja o exercício abaixo).

Nós concluímos então que o resultado vale nos dois casos possíveis, CQD.

EXERCÍCIO 1. Demonstre o caso 2 do lema acima.

Com base no lema acima, podemos concluir:

LEMA 2. Pelo menos "$3*\lfloor s/2 \rfloor - 2$" elementos de v[i..f] ficam de fora dos intervalos das chamadas recursivas das linhas 18 e 20.

==

PROVA:

Se a chamada recursiva da linha 18 é realizada, então pelo menos "$3(\lfloor s/2 \rfloor - 1) + 1$" elementos de v[i..f] não estão no intervalo v[i..q-1] dessa chamada, pois:

1. Pelo lema 1, há pelo menos "$\lfloor s/2 \rfloor - 1$" segmentos de 5 elementos cujas medianas são maiores que a mediana das medianas, e, em cada tal segmento, há 3 elementos maiores que a mediana das medianas: a mediana do segmento e os dois elementos do segmento que são maiores que esta última.

2. A mediana das medianas também não é incluída no intervalo da chamada recursiva.

Raciocínio análogo se aplica à chamada da linha 20, o que conclui a prova.

==

Pelo lema acima, portanto, as chamadas recursivas das linhas 18 e 20 são feitas para intervalos de no máximo

$$\begin{aligned} t -(3*\lfloor s/2 \rfloor - 2) &= t -3*\lfloor s/2 \rfloor +2 \\ &= t -3*\lfloor \lceil t/5 \rceil /2 \rfloor +2 \end{aligned}$$

elementos, conforme argumentado anteriormente.

TEMPO DE EXECUÇÃO DO ALGORITMO

Com base nas observações da seção anterior, podemos concluir que o tempo de execução do algoritmo no pior caso é limitado superiormente por "g(t)", onde

$$g(t) = \begin{cases} a, & \text{se } t \le 5 \\ b + c*t + g(\lceil t/5 \rceil) + g(t -3*\lfloor \lceil t/5 \rceil /2 \rfloor +2), & \text{se } t > 5, \end{cases}$$

para certas constantes reais "$a,b,c > 0$" suficientemente grandes.

TEOREMA 1. $g(t) = O(t)$.

==

PROVA:

Temos que mostrar que existem "$m,d > 0$" tais que, para todo $t >= m$,

$$0 <= g(t) <= d*t.$$

De fato, sejam

$$m = 1, \text{ e}$$

$$d = \text{máx}\{ g(1), g(2), g(3), ... , g(99) , b + 20c \} > 0.$$

Como é evidente que a função g(t) sempre assume valores positivos, resta mostrar apenas que "$g(t) <= d*t$" para todo "$t >= m$". Nós o faremos por indução em "t":

* BASE: seja t ∈ [1..99]. Temos:

```
g(t) <= d     // pela definição de "d"
     <= d*t,  // pois t >= 1
```

como desejado.

* PASSO: consideremos um natural "t >= 100" e suponhamos que "g(k) <= d*k"
  ----- para todo k ∈ [1..t-1] (hipótese indutiva); nós mostraremos então que
  g(t) <= d*t. De fato, temos:

```
g(t)  = b + c*t + g( ⌈t/5⌉ ) + g( t -3*⌊ ⌈t/5⌉/2 ⌋ +2 )
     <= b + c*t + d*( ⌈t/5⌉ ) + d*( t -3*⌊ ⌈t/5⌉/2 ⌋ +2 )  // Pela H.I.
      = b + c*t + 2d + d*( ⌈t/5⌉ + t -3*⌊ ⌈t/5⌉/2 ⌋ )
      < b + c*t + 2d + d*( ⌈t/5⌉ + t -3*( ⌈t/5⌉/2 - 1) )  // -⌊x⌋ < -(x-1)
      = b + c*t + 2d + d*( ⌈t/5⌉ + t -3*⌈t/5⌉/2 +3 )
      = b + c*t + 5d + d*( t - ⌈t/5⌉/2 )
     <= b + c*t + 5d + d*( t - (t/5)/2 )  // -⌈x⌉ <= -x
      = b + c*t + 5d + d*( t - t/10 )
      = b + c*t + 5d + d*t*9/10
     <= d*t*9/10 + d*t*1/10  // pois "b + c*t + 5d <= d*t/10" (ver abaixo)
      = d*t,
```

como desejado. Em particular, a última inequação acima é verdadeira porque

```
b + c*t + 5d <= d*t/10  <==>  b + 5d <= t*(d/10 - c)
                        <==   b + 5d <= 100*(d/10 - c)  // Pois t >= 100
                        <==>  b + 5d <= 10d - 100c
                        <==>  b + 100c <= 5d
                        <==>  (b + 100c)/5 <= d
                        <==   b + 20c <= d,
```

o que é verdade pela escolha de "d".

Nós provamos então que "g(t) <= d*t" para todo "t >= m", o que conclui a prova.
==