Construção e Análise de Algoritmos

aula 1y: Corretude de algoritmos II

1 Introdução

Hoje nós vamos continuar a nossa analogia entre máquinas e algoritmos.

A observação inicial é que, no momento em que as máquinas ficam muito grandes e complicadas, não é mais viável analisar o seu funcionamento examinando as peças isoladas.

De fato, também não é muito viável construir máquinas complicadas desse jeito.

Uma pequena história ilustra essa ideia.

> *Em uma cidade haviam dois relojoeiros famosos pelos seus bonitos relógios.*
>
> *Apesar de ambos realizarem o seu trabalho com muito cuidado, eles faziam as coisas de maneira bem diferente.*
>
> *O primeiro deles montava os seus relógios diretamente a partir das pecinhas miúdas.*
>
> *O trabalho exigia muita atenção e extrema precisão.*
>
> *Mas, quando tudo estava finalmente em seu lugar, o resultado era quase uma obra de arte: tic-tac, tic-tac, tudo funcionando de maneira suave e contínua, uma beleza de se ver.*
>
> *O segundo relojoeiro não trabalhava assim.*
>
> *Seus relógios eram montados a partir de mecanismos mais simples; esses mecanismos eram construídos a partir de componentes ainda mais simples; e cada componente era formado por meia dúzia de peças apenas.*
>
> *Esse relojoeiro passava os seus dias montando componentes, montando mecanismos, e montando os seus relógios.*
>
> *Em um único dia de trabalho ele podia finalizar a construção de ao menos três relógios, todos funcionando perfeitamente, e sem nenhum defeito de fabricação.*
>
> *Os dias do primeiro relojoeiro não eram assim.*
>
> *Se ele estava um pouco cansado, ou se havia confusão em casa, não era possível fazer muita coisa.*
>
> *Outras vezes ele passava o dia pensativo, imaginando peças que ainda não existiam, e imaginando maneiras de encaixar as peças que também ainda não existiam.*
>
> *Com sorte, ele conseguia terminar a construção de um único relógio por semana.*
>
> *É uma pena que não se fazem mais as coisas assim ...*

Voltando à nossa discussão, algoritmos complicados são como máquinas complicadas.

Em geral, eles são construídos a partir de componentes mais simples.

E, para analisar o seu funcionamento, nós concentramos a nossa atenção não apenas nas peças, mas também na funcionalidade dos componentes.

A seguir nós vamos ver alguns exemplos concretos desse tipo de análise.

2 Corretude do algoritmo da bolha

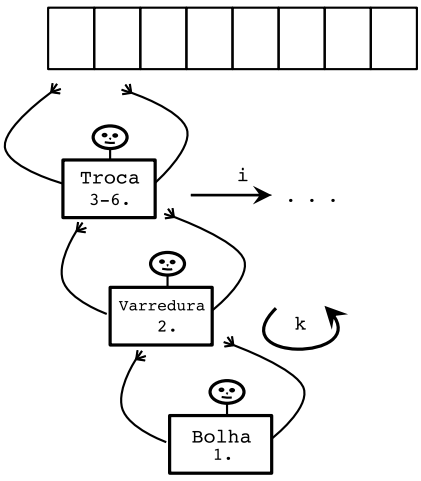
Abaixo nós temos o pseud-código do algoritmo de ordenação da bolha

```
     Procedimento  ord-Bolha ( V[1..n] )
      {
    1.    Para  k <-- 1  Até  n-1

    2.        Para  i <-- 1  Até  n-k

    3.            Se ( V[i] > V[i+1] )
                   {
  4-6.                aux  <--  V[i];    V[i]  <--  V[i+1];    V[i+1]  <--  aux
                   }
       }
```

E a figura abaixo ilustra a sua estrutura de componentes.



Isto é, o componente mais simples corresponde às linhas `3-6`, e sua função é trocar dois elementos adjacentes de posição caso o primeiro deles seja maior do que o segundo.

O segundo componente corresponde à linha `2`. Ele realiza uma varredura sobre a lista, executando o primeiro componente sobre todos os pares de elementos adjacentes. Sua função é mover o maior elemento da lista para o lado direito.

O terceiro componente corresponde à linha `1`, e sua função é ordenar a lista executando o segundo componente repetidamente.

A seguir, nós vamos mostrar que se todos os componentes realizam a sua função corretamente, o algoritmo como um todo funciona corretamente.

etapa 1: corretude do componente `Troca`:

Aqui não há muito o que fazer.

O componente `Troca` tem apenas 4 instruções (sem laço) e é imediato ver que cumpre a sua função.

etapa 2: corretude do componente `Varredura`:

Aqui, nós vamos fazer uma análise semelhante àquelas que fizemos na aula passada.

Isto é, a ideia é concentrar a atenção na variável `i`:

- a variável `i` indica o par de elementos que está sendo analisado nesse momento (pelo componente `Troca`)

  ela tem a propriedade de que `V[i]` é maior elemento da lista até a posição `i`

Agora que nós descrevemos a função da variável `i`, é fácil ver que no início tudo está em ordem:

```
i <-- 1
```

A seguir, nós examinamos o que acontece quando nós damos voltas no laço.

A ideia é utilizar o fato de que o componente `Troca` funciona corretamente (etapa 1).

Isto é, nós não vamos examinar as linhas `3-6` diretamente, mas apenas levar em conta a sua funcionalidade.

Vejamos.

Existem basicamente dois casos possíveis:

- `V[i]` é maior do que `V[i+1]` ou não

No primeiro caso, o componente `Troca` irá trocar esses dois elementos de posição.

Isto é, o elemento que estava na posição `i` irá para a posição `i+1`.

Esse elemento era maior do que todos os elementos até a posição `i-1` (pela propriedade da variável `i`).

E agora ele também é maior do que o elemento que ficará na posição i.

Logo, quando a variável `i` for incrementada no final do laço, ela continuará obedecendo a sua propriedade.

No segundo caso, o componente `Troca` não irá trocar os dois elementos de posição.

Nós sabemos que o elemento da posição `i+1` é maior do que o elemento da posição `i`.

E nós já sabíamos que o elemento da posição `i` é maior do que todos os elementos até a posição `i-1` (pela propriedade da variável `i`).

Logo, quando a variável `i` for incrementada no final do laço, ela continuará obedecendo a

sua propriedade.

Ou seja, em ambos os casos, tudo permanece no lugar quando o laço dá as suas voltas.

Finalmente, a última volta do laço começa com `i` tendo o valor `n-k`.

E ao final dessa volta nós temos a garantia de que o elemento da posição `n-k+1` é maior do que todos os elementos até a posição `n-k`.

(Essa é a função do componente `Varredura`.)

etapa 3: corretude do componente `Bolha`:

Intuitivamente, a coisa é simples.

- o componente `Varredura` trabalha sobre a faixa `V[1..n-k+1]` da lista
- e a sua função é mover o maior elemento dessa faixa para a última posição
- logo, quando ele é executado com `k=1`, ele coloca o maior elemento da lista na última posição
- quando ele é executado com `k=2`, ele coloca o segundo maior elemento da lista na penúltima posição
- e assim por diante ...

A seguir, nós vamos argumentar de maneira ligeiramente mais precisa.

A peça chave dessa vez é a variável `k`:

- a variável `k` indica a faixa de trabalho do componente `Varredura`

  ela tem a propriedade de que, após a execução do componente `Varredura`, os `k` maiores elementos já estão em sua posição correta

Mais uma vez, a ideia é utilizar o fato de que o componente `Varredura` funciona corretamente (etapa 2).

Vejamos.

No início, nós temos que

```
k <-- 1
```

e, nessa primeira volta do laço, o procedimento `Varredura` irá trabalhar na faixa

$$\texttt{V[i..n} - 1 + 1\texttt{]} \quad = \quad \texttt{V[1..n]}$$

e mover o maior elemento da lista para a última posição.

Ou seja, até aqui tudo bem: após a execução do procedimento `Varredura`, a variável `k` obedece a sua propriedade (i.e., $\texttt{k} = 1$ e o maior elemento está no lugar correto).

A seguir, nós precisamos mostrar que as voltas do laço não tiram as coisas do lugar.

Para fazer isso, nós consideramos uma volta qualquer do laço, digamos quando `k` vale `m`.

Nós queremos verificar se essa volta tira as coisas do lugar.

Então, nós assumimos que no início dessa volta tudo ainda está no lugar.

Isto é, nós assumimos que no final da volta anterior, a variável `k` ainda obedecia a sua propriedade: os $\mathtt{m}-1$ maiores elementos já estavam no lugar correto.

Tendo assumido isso, agora basta observar que nessa volta o componente `Varredura` irá trabalhar na faixa

$$\mathtt{V}[\mathtt{i}..\mathtt{n}-\mathtt{m}+1] \quad = \quad \mathtt{V}[1..\mathtt{n}-(\mathtt{m}-1)]$$

e irá mover o maior elemento dessa faixa para a posição `V[n-(m-1)]`.

Ora, como os $\mathtt{m}-1$ maiores elementos já estão em suas posições corretas, o maior elemento dessa faixa é precisamente o `m`-ésimo maior elemento da lista.

E o seu lugar correto é precisamente a posição `V[n-(m-1)]`.

(porque?)

Portanto, ao final dessa volta, a variável `k` (que tem o valor `m`), obedece a sua propriedade: os `m` maiores elementos da lista estão no lugar correto.

Ou seja, as voltas do laço não tiram as coisas do lugar.

Finalmente, como isso é o caso, basta observar que após a última volta do laço

```
k <-- n-1
```

todos o $\mathtt{n}-1$ maiores elementos estão em seu lugar correto.

Mas, isso significa que a lista está completamente ordenada.

Exercícios