UNIVERSIDADE DO ESTADO DO AMAZONAS
ESCOLA SUPERIOR DE TECNOLOGIA
NÚCLEO DE COMPUTAÇÃO - NUCOMP

Algoritmos e Estruturas de Dados II

Geração de objetos combinatórios

Prof. Flávio José Mendes Coelho
fcoelho@uea.edu.br

Objetivos

Entender...

1. O algoritmo de *Steinhaus-Jonhson-Trotter* para gerar permutações
2. O algoritmo de geração de subconjuntos

Gerando permutações

Problema: dado um conjunto $A = \{1, 2, ..., n\}$, gerar todas as permutações dos elementos de A.

$P_3 = \{(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)\}$

$|P_n| = n!$

Gerando permutações

Ideia: Deseja-se gerar todas as n! permutações de {1, 2, ..., n}. Suponha que todas as (n–1)! já estejam geradas. Para gerar as n! permutações basta introduzir n em cada uma das n–1 posições de cada uma das (n–1)! permutações existentes.

Há, assim, n.(n–1)! = n! permutações.

Gerando permutações

Questão: e se o conjunto a ser permutado for um conjunto de outros objeto que não números (palavras, letras, símbolos gráficos, etc) ou for composto por números não consecutivos $A = \{a_1, a_2, ..., a_n\}$?

Permutar os índices de A corresponde a permutar A.

Gerando permutações

$A = \{1, 2, 3\}; P_3?$

Início: 1

Insere $\overleftarrow{2}$ em 1: 12

Insere $\overleftarrow{3}$ em 12: 123 132 312

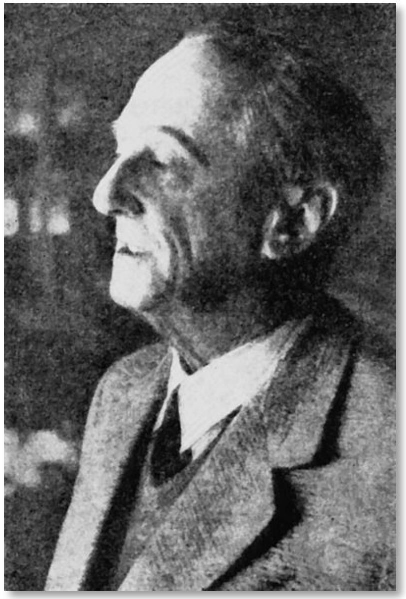
Insere $\overleftarrow{2}$ em 1: 21

Insere $\overrightarrow{3}$ em 21: 321 231 213

Gerando permutações

O algoritmo de *Steinhaus-Jonhson-Trotter* produz permutações sem precisar gerá-las a partir de valores menores do que *n*.

Descoberto independentemente pelos matemáticos Hugo Steinhaus (polonês), Selmer M. Jonhson (americano) e Hale F. Trotter (?)



Hugo Steinhaus
(1887-1972)

Algoritmo *Jonhson-Trotter*

$$\overrightarrow{3}\,\overleftarrow{2}\,\overrightarrow{4}\,\overleftarrow{1}$$

É um ***móvel***: seta aponta para adjacente menor

outro *móvel*

Um elemento *móvel* deve se mover pela permutação no sentido em que sua seta aponta.

Algoritmo *Jonhson-Trotter*

Entrada

Inteiro positivo *n.*

Saída

Lista de todas as permutações de $\{1, ..., n\}$.

```
JHONSON-TROTTER( N )
inicie com a permutação ←1 ←2 ... ←N na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

JHONSON-TROTTER(N)

```
inicie com a permutação 1⃖ 2⃖ ... N⃖ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{1}\ \overleftarrow{2}\ \overleftarrow{3}$$

$P=\{123\}$

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overleftarrow{1}\ \overleftarrow{2}\ \overleftarrow{3}$$

$P=\{123\}$

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P

enquanto a última permutação tiver um móvel

    ache seu maior móvel K

    troque K com seu adjacente apontado por K

    inverta as setas dos maiores do que K

    adicione a nova permutação gerada à lista P

retorne P

$$\overleftarrow{1}\ \overleftarrow{2}\ \overleftarrow{3}$$

P={123}

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overleftarrow{1}\ \overleftarrow{3}\ \overleftarrow{2}$$

$P=\{123\}$

JHONSON-TROTTER(N)

```
inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{1}\ \overleftarrow{3}\ \overleftarrow{2}$$

$P = \{123\}$

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overleftarrow{1}\ \overleftarrow{3}\ \overleftarrow{2}$$

P={123, 132}

```
JHONSON-TROTTER( N )
    inicie com a permutação ←1 ←2 ... ←N na lista P
    enquanto a última permutação tiver um móvel
        ache seu maior móvel K
        troque K com seu adjacente apontado por K
        inverta as setas dos maiores do que K
        adicione a nova permutação gerada à lista P
    retorne P
```

$$\overleftarrow{1}\ \overleftarrow{3}\ \overleftarrow{2}$$

$P = \{123,\ 132\}$

JHONSON-TROTTER(N)

```
inicie com a permutação 1← 2← ... N← na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{1}\ \overleftarrow{3}\ \overleftarrow{2}$$

P={123, 132}

Jhonson-Trotter(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P

enquanto a última permutação tiver um móvel

    ache seu maior móvel K

    troque K com seu adjacente apontado por K

    inverta as setas dos maiores do que K

    adicione a nova permutação gerada à lista P

retorne P

$$\overleftarrow{1}\ \overleftarrow{3}\ \overleftarrow{2}$$

P={123, 132}

```
JHONSON-TROTTER( N )
    inicie com a permutação ←1 ←2 ... ←N na lista P
    enquanto a última permutação tiver um móvel
        ache seu maior móvel K
        troque K com seu adjacente apontado por K
        inverta as setas dos maiores do que K
        adicione a nova permutação gerada à lista P
    retorne P
```

$$\overleftarrow{3}\ \overleftarrow{1}\ \overleftarrow{2}$$

$P=\{123, 132\}$

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \dots \overleftarrow{N}$ na lista P

enquanto a última permutação tiver um móvel

    ache seu maior móvel K

    troque K com seu adjacente apontado por K

    inverta as setas dos maiores do que K

    adicione a nova permutação gerada à lista P

retorne P

$$\overleftarrow{3}\ \overleftarrow{1}\ \overleftarrow{2}$$

P={123, 132}

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overleftarrow{3}\ \overleftarrow{1}\ \overleftarrow{2}$$

$P=\{123, 132, 312\}$

JHONSON-TROTTER(N)

```
inicie com a permutação ←1 ←2 ... ←N na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{3}\ \overleftarrow{1}\ \overleftarrow{2}$$

P={123, 132, 312}

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overleftarrow{3}\ \overleftarrow{1}\ \overleftarrow{2}$$

P={123, 132, 312}

JHONSON-TROTTER(N)

```
inicie com a permutação ←1 ←2 ... ←N na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{3}\ \overleftarrow{1}\ \overleftarrow{2}$$

$P=\{123, 132, 312\}$

JHONSON-TROTTER(N)

```
inicie com a permutação ←1 ←2 ... ←N na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{3}\ \overleftarrow{2}\ \overleftarrow{1}$$

P={123, 132, 312}

JHONSON-TROTTER(N)

```
inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista $P$
**enquanto** a última permutação tiver um móvel
    ache seu maior móvel $K$
    troque $K$ com seu adjacente apontado por $K$
    inverta as setas dos maiores do que $K$
    adicione a nova permutação gerada à lista $P$
**retorne** $P$
```

$$\overrightarrow{3}\ \overleftarrow{2}\ \overleftarrow{1}$$

P={123, 132, 312}

JHONSON-TROTTER(N)

```
inicie com a permutação ←1 ←2 ... ←N na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overrightarrow{3}\ \overleftarrow{2}\ \overleftarrow{1}$$

P={123, 132, 312, 321}

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \dots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overrightarrow{3}\ \overleftarrow{2}\ \overleftarrow{1}$$

P={123, 132, 312, 321}

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overrightarrow{3}\ \overleftarrow{2}\ \overleftarrow{1}$$

P={123, 132, 312, 321}

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overrightarrow{3}\ \overleftarrow{2}\ \overleftarrow{1}$$

P={123, 132, 312, 321}

JHONSON-TROTTER(N)

```
inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \dots \overleftarrow{N}$ na lista $P$
enquanto a última permutação tiver um móvel
    ache seu maior móvel $K$
    troque $K$ com seu adjacente apontado por $K$
    inverta as setas dos maiores do que $K$
    adicione a nova permutação gerada à lista $P$
retorne $P$
```

$$\overleftarrow{2}\ \overrightarrow{3}\ \overleftarrow{1}$$

$P = \{123, 132, 312, 321\}$

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overleftarrow{2}\ \overrightarrow{3}\ \overleftarrow{1}$$

P={123, 132, 312, 321}

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overleftarrow{2}\ \overrightarrow{3}\ \overleftarrow{1}$$

P={123, 132, 312, 321, 231}

JHONSON-TROTTER(N)

```
inicie com a permutação ←1 ←2 ... ←N na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{2}\ \overrightarrow{3}\ \overleftarrow{1}$$

P={123, 132, 312, 321, 231}

JHONSON-TROTTER(N)

```
inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista $P$
enquanto a última permutação tiver um móvel
    ache seu maior móvel $K$
    troque $K$ com seu adjacente apontado por $K$
    inverta as setas dos maiores do que $K$
    adicione a nova permutação gerada à lista $P$
retorne $P$
```

$$\overleftarrow{2}\ \overrightarrow{3}\ \overleftarrow{1}$$

P={123, 132, 312, 321, 231}

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overleftarrow{2}\ \overrightarrow{3}\ \overleftarrow{1}$$

P={123, 132, 312, 321, 231}

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P

$$\overleftarrow{2}\ \overleftarrow{1}\ \overrightarrow{3}$$

P={123, 132, 312, 321, 231}

JHONSON-TROTTER(N)

```
inicie com a permutação ←1 ←2 ... ←N na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{2}\ \overleftarrow{1}\ \overrightarrow{3}$$

P={123, 132, 312, 321, 231}

JHONSON-TROTTER(N)

```
inicie com a permutação ←1 ←2 ... ←N na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{2}\ \overleftarrow{1}\ \overrightarrow{3}$$

$$P=\{123, 132, 312, 321, 231, 213\}$$

JHONSON-TROTTER(N)

```
inicie com a permutação ←1 ←2 ... ←N na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{2}\;\overleftarrow{1}\;\overrightarrow{3}$$

P={123, 132, 312, 321, 231, 213}

JHONSON-TROTTER(N)

```
inicie com a permutação ←1 ←2 ... ←N na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{2}\,\overleftarrow{1}\,\overrightarrow{3}$$

P={123, 132, 312, 321, 231, 213}

JHONSON-TROTTER(N)

```
inicie com a permutação ←1 ←2 ... ←N na lista P
enquanto a última permutação tiver um móvel
    ache seu maior móvel K
    troque K com seu adjacente apontado por K
    inverta as setas dos maiores do que K
    adicione a nova permutação gerada à lista P
retorne P
```

$$\overleftarrow{2}\ \overleftarrow{1}\ \overrightarrow{3}$$

P={123, 132, 312, 321, 231, 213}

JHONSON-TROTTER(N)

inicie com a permutação $\overleftarrow{1}\ \overleftarrow{2} \ldots \overleftarrow{N}$ na lista P

enquanto a última permutação tiver um móvel

    ache seu maior móvel K

    troque K com seu adjacente apontado por K

    inverta as setas dos maiores do que K

    adicione a nova permutação gerada à lista P

retorne P

$$P=\{123, 132, 312, 321, 231, 213\}$$

Algoritmo *Jonhson-Trotter*

Um dos métodos mais eficientes para gerar permutações.

Tempo: $\Theta(n!)$ proporcional ao número de permutações.

Gerando subconjuntos

Problema: dado um conjunto $C = \{a_1, a_2, ..., a_n\}$, gerar todos os subconjuntos de C (conj. potência $\mathscr{P}(C)$).

$\mathscr{P}(\{1, 2, 3\})$: $\{\emptyset, \{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$

$|\mathscr{P}(C)| = 2^n$

Gerando subconjuntos

Ideia: Deseja-se gerar todos os subconjuntos de $A = \{a_1, a_2, ..., a_n\}$, isto é, $\mathscr{P}(A)$. Dentre todos os subconjuntos de A, há aqueles que contém a_n e os que não contém a_n. O grupo dos que não contém a_n , corresponde aos subconjuntos de $\{a_1, ..., a_{n-1}\}$. O grupo que contém a_n corresponde aos subconjuntos de $\{a_1,..., a_{n-1}\}$ com a_n incluído em cada um deles.

Gerando subconjuntos

n	subconjuntos
0	$\varnothing$
1	$\varnothing, \{a_1\}$
2	$\varnothing, \{a_1\}, \{a_2\}, \{a_1, a_2\}$
3	$\varnothing, \{a_1\}, \{a_2\}, \{a_1, a_2\},$
...	$\{a_3\}, \{a_1, a_3\}, \{a_2, a_3\}, \{a_1, a_2, a_3\}$

Gerando subconjuntos

Números inteiros binários podem gerar os subconjuntos de um conjunto.

1 – 000: $\varnothing$

2 – 001: $\{\quad a_3\}$

3 – 010: $\{\quad a_2 \quad\}$

4 – 011: $\{\quad a_2, a_3\}$

5 – 100: $\{a_1 \quad\}$

6 – 101: $\{a_1 \quad a_3\}$

7 – 110: $\{a_1, a_2 \quad\}$

8 – 111: $\{a_1, a_2, a_3\}$

Referências

- A. Levitin. *Introduction to the Design and Analysis of Algorithms*. 3rd edition. Addison-Wesley, 2007.