PUC-RIO
INF1036 - Probabilidade Computacional

Material 3 - Técnicas de Contagem

Professora - Ana Carolina Letichevsky*

2022.1

*Material Adaptado de Professor Hélio Lopes

Teoria das Probabilidades

A Teoria das Probabilidades é uma área da matemática que desenvolve modelos que podem ser utilizados para estudar experimentos aleatórios. A probabilidade de um evento é a medida ou chance desse evento ocorrer.

Definição clássica de Probabilidade (também chamada definição de Laplace). Sejam:

- S o conjunto de possíveis resultados (saídas) de um experimento;
- ζ os elementos de S, que são chamados de ponto amostral ou simplesmente de amostra;
- E um subconjunto possível do espaço amostral S (evento).

Teoria das Probabilidades

Na definição de probabilidade de Laplace é preciso considerar que:

- Há um número finito (n) de elementos em S;
- A união de todos os eventos é S;
- Os pontos do espaço amostral são equiprováveis;
- Todo evento E é obtido a partir da união de m pontos do espaço amostral sendo $m \leq n$

$$P(E) = \frac{n(E)}{n(S)} = \frac{m}{n}$$

E, portanto, para todo evento E, $0 \leq P(E) \leq 1$.

Tipos de Espaço Amostral

- **Finito:** é aquele em que existe um número finito de possíveis saídas.

- **Discreto e infinito:** é aquele que possui um número infinito de possíveis saídas, porém esse conjunto de saídas é enumerável.

- **Contínuo:** é aquele em que as possíveis saídas constituem um conjunto contínuo.

Um conjunto é dito ser enumerável se ele possui uma correspondência biunívoca com o conjunto dos números inteiros.

Análise Combinatória

A Análise Combinatória é a parte da matemática que estuda estruturas e soluções discretas.

Uma das suas aplicações é resolver problemas de contagem (sem que seja necessário enumerar os elementos) de certos tipos de subconjuntos de um conjunto finito.

Breve resumo de Teoria dos Conjuntos

Sejam:

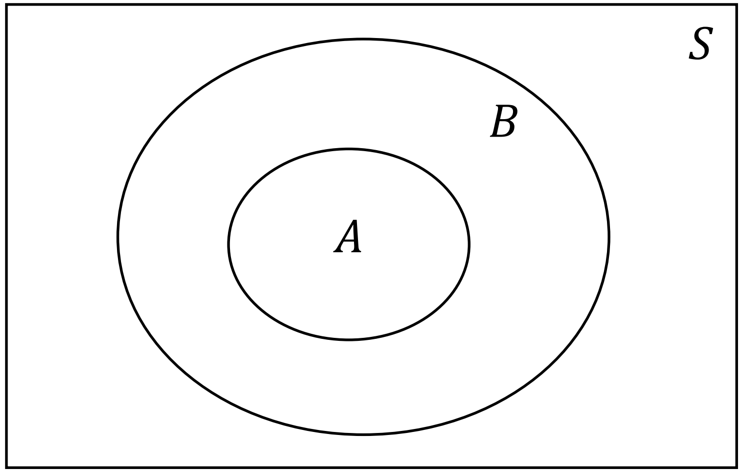
- A e B dois conjuntos;
- S o conjunto enumerável em uma determinada solução;
- $\emptyset$ o conjunto vazio.

Temos então que para todo conjunto:

- $A \subset S$
- $A \cup \emptyset = A$
- $A \cap \emptyset = \emptyset$

Breve resumo de Teoria dos Conjuntos

Se todo elemento de A é também elemento de B, dizemos que A é subconjunto de B e representamos por $A \subset B$.

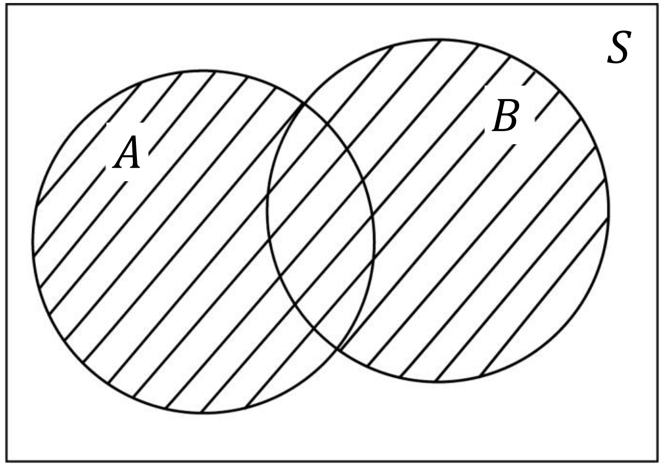


A é subconjunto de B

Breve resumo de Teoria dos Conjuntos

O conjunto de todos os elementos que pertencem a A ou a B ou a ambos é representado por $A \cup B$.

$$A \cup B = \{\zeta \in S \mid \zeta \in A \; ou \; \zeta \in B\}$$

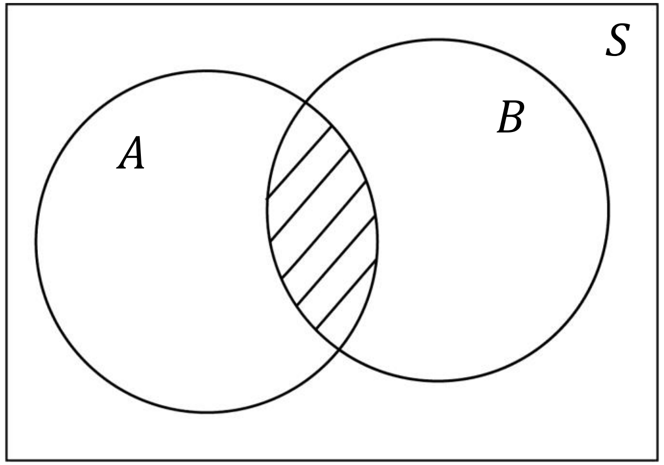


União de A com B

Breve resumo de Teoria dos Conjuntos

O conjunto de todos os elementos que pertencem, simultaneamente, a A e a B é representado por $A \cap B$.

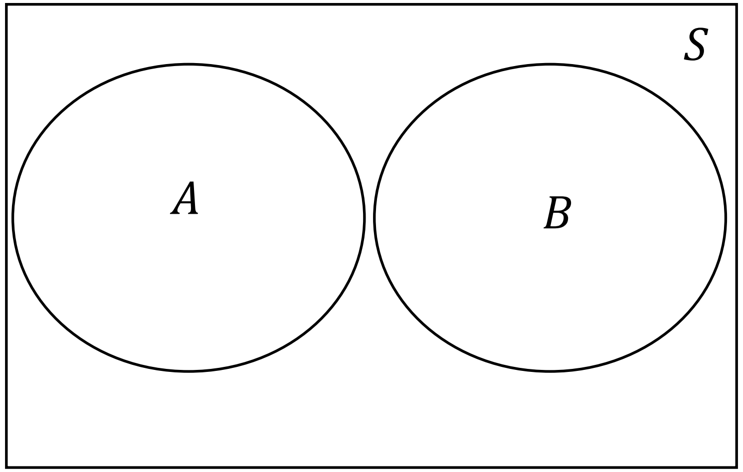
$$A \cap B = \{\zeta \in S \mid \zeta \in A \text{ e } \zeta \in B\}$$



Interseção de A com B

Breve resumo de Teoria dos Conjuntos

Os conjuntos A e B são disjuntos se $A \cap B = \emptyset$. Os conjuntos que não possuem elementos comuns também são chamados de mutuamente exclusivos.



A e B são disjuntos

Breve resumo de Teoria dos Conjuntos

Se temos n conjuntos A_1, A_2,…, A_n podemos estender a definição de união e interseção.

A união de n conjuntos é representada por:

$$\bigcup_{i=1}^{n} A_i = A_1 \cup A_2 \cup \cdots \cup A_n$$

A interseção de n conjuntos é representada por:

$$\bigcap_{i=1}^{n} A_i = A_1 \cap A_2 \cap \cdots \cap A_n$$

Dizemos que n conjuntos são disjuntos se eles forem disjuntos quando tomados 2 a 2.

Breve resumo de Teoria dos Conjuntos

Se temos infinitos conjuntos podemos estender a definição de união e interseção.

A união de infinitos conjuntos é representada por:

$$\bigcup_{i=1}^{\infty} A_i = A_1 \cup A_2 \cup \cdots$$

A interseção de infinitos conjuntos é representada por:

$$\bigcap_{i=1}^{\infty} A_i = A_1 \cap A_2 \cap \cdots$$

Breve resumo de Teoria dos Conjuntos

Se temos n conjuntos A_1, A_2,..., A_n, então:

$A_i \subset A_j$, se somente se $A_i \cup A_j = A_j$

$A_i \subset A_j$, se somente se $A_i \cap A_j = A_i$

$A_i \cup (A_j \cup A_k) = (A_i \cup A_j) \cup A_k$

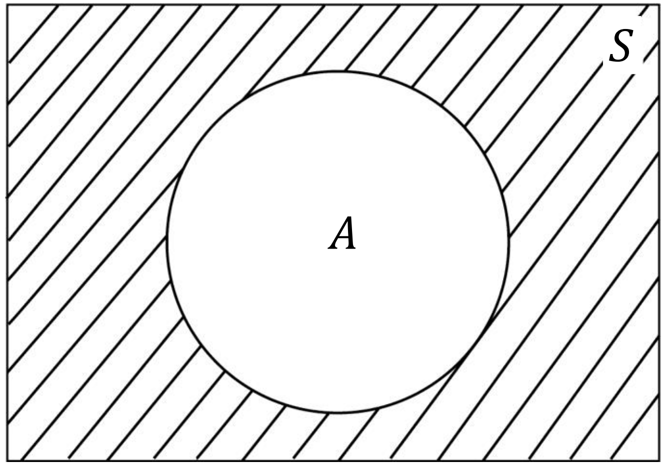
$A_i \cap (A_j \cap A_k) = (A_i \cap A_j) \cap A_k$

$A_i \cup (A_j \cap A_k) = (A_i \cup A_j) \cap (A_i \cup A_k)$

Breve resumo de Teoria dos Conjuntos

Chamamos de conjunto complementar de A o conjunto dos elementos de S que não pertencem a A e representamos por A^c ou por $\bar{A}$.

$$\bar{A} = \{\zeta \in S \mid \zeta \notin A\}$$



Complemento de A

Breve resumo de Teoria dos Conjuntos

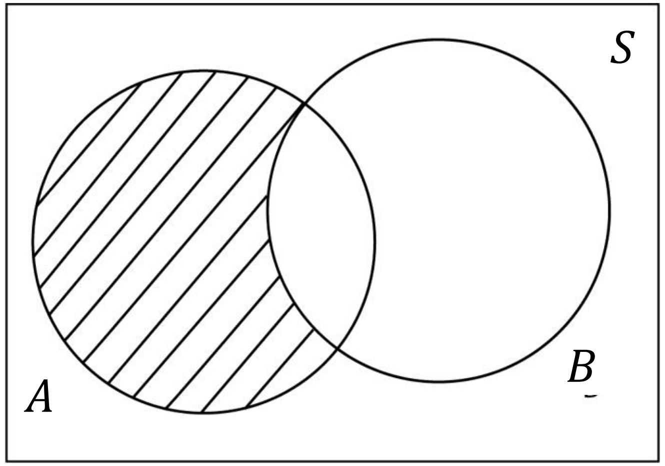
Algumas propriedades relacionadas ao conjunto complementar:

- $(A^c)^c = \bar{\bar{A}} = A$
- $A \cup \bar{A} = S$
- $A \cap \bar{A} = \emptyset$
- $\bar{\emptyset} = S$
- $\bar{S} = \emptyset$
- $S \cup A = S$
- $S \cap A = A$

Breve resumo de Teoria dos Conjuntos

Conjunto diferença de A e B é representado por $A - B$:

$$A - B = A \cap \bar{B} = \{\zeta \in S \mid \zeta \in A \; e \; \zeta \notin B\}$$



Conjunto diferença de A e B

Partição

Dizemos que uma coleção de conjuntos $A_1, A_2, \ldots, A_n$ é uma **partição de um conjunto** B se:

$$A_1 \cup A_2 \cup \cdots \cup A_n = B \quad \text{e se} \quad A_i \cap A_j = \emptyset \text{ quando } i \neq j$$

Ou seja, uma partição de um conjunto B é uma coleção de subconjuntos disjuntos não-vazios de B, cuja união é igual a B.

$$B = \{a, b, c, d, e, f, g\}$$

$\{\{a, b\}, \{c, d\}, \{e, f, g\}\}$ é uma partição de B?

Produto Cartesiano

O par de elementos a e b, onde a é chamado de *primeiro* elemento e b de *segundo* elemento, é definido como um **par ordenado**, e é denotado por (a, b).

Dois pares ordenados (a, b) e (c, d) são iguais se e somente se $a = c$ e $b = d$.

Para quaisquer dois conjuntos A e B, o **produto cartesiano** de A e B, escrito como $A \times B$, é o conjunto de todos os pares ordenados dos elementos, onde o primeiro elemento do par é um elemento do conjunto A e o segundo elemento do par é um elemento do conjunto B.

$$A \times B = \{(a, b) \mid a \in A \; e \; b \in B\}$$

Produto Cartesiano

Exemplo) Se $A = \{a, b, c\}$ e $B = \{p, q\}$, então:

$$A \times B = \{(a, p), (a, q), (b, p), (b, q), (c, p), (c, q)\}$$

e

$$B \times A = \{(p, a), (p, b), (p, c), (q, a), (q, b), (q, c)\}$$

```
A = ['a', 'b', 'c']
B = ['p', 'q’]

W = []
for i in range(len(A)):
   for j in range(len(B)):
      elemento = (A[i], B[j]) #elemento é um 'tuple'
      W.append(elemento)
print(W)
```

Produto Cartesiano em R

Exemplo) Se $A = \{a, b, c\}$ e $B = \{p, q\}$, então:

$$A \times B = \{(a,p),(a,q),(b,p),(b,q),(c,p),(c,q)\}$$

e

$$B \times A = \{(p,a),(p,b),(p,c),(q,a),(q,b),(q,c)\}$$

```
A = c('a', 'b', 'c')
B = c('p', 'q')

W = list() #W é uma lista
k = 1
for(i in 1: length(A)){
  for(j in 1: length(B)){
    W[[k]] = c(A[i], B[j])
    k = k + 1
  }
}
print(W)
```

Produto Cartesiano

Estendendo, se temos n conjuntos A_1, A_2,..., A_n , então,

o produto cartesiano:

$$A_{\boldsymbol{i}} \times A_{\boldsymbol{j}} \times \cdots \times A_{\boldsymbol{n}}$$

é definido como n-uplas $(a_i, a_j, \ldots, a_n)$, onde:

$$a_i \in A_i, a_j \in A_j, \ldots, a_n \in A_n$$

Produto Cartesiano

Algumas propriedades relacionadas ao produto cartesiano:

- $A \times \emptyset = \emptyset \times A = \emptyset$
- $n(A_1 \times A_2 \times \cdots \times A_n) = n(A_1) \times n(A_2) \times \cdots n(A_n)$
- $A \times (B \cup C) = (A \times B) \cup (A \times C)$
- $A \times (B \cap C) = (A \times B) \cap (A \times C)$

Propriedades das Operações de Conjuntos

As operações de união e interseção também satisfazem às seguintes propriedades:

- **Comutatividade**:

$$A \cup B = B \cup A$$

$$A \cap B = B \cap A$$

- **Associatividade**:

$$A \cup (B \cup C) = (A \cup B) \cup C$$

$$A \cap (B \cap C) = (A \cap B) \cap C$$

- **Distributividade** :

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

Estendidas

$$A \cap \bigcup_{i=1}^{n} B_i = \bigcup_{i=1}^{n} A \cap B_i$$

$$A \cup \bigcap_{i=1}^{n} B_i = \bigcap_{i=1}^{n} A \cup B_i$$

Propriedades das Operações de Conjuntos

- **Leis de De Morgan**:

$$\overline{A \cup B} = \bar{A} \cap \bar{B}$$

$$\overline{A \cap B} = \bar{A} \cup \bar{B}$$

Que podem ser estendidas nas seguintes formas:

$$\overline{\bigcup_{i=1}^{n} A_i} = \bigcap_{i=1}^{n} \overline{A_i}$$

$$\overline{\bigcap_{i=1}^{n} A_i} = \bigcup_{i=1}^{n} \overline{A_i}$$

Técnicas de Contagem

Princípio de Adição:

Se A e B são dois conjuntos finitos tais que $A \cap B = \emptyset$ (disjuntos) com p e q elementos, respectivamente, então $A \cup B$ possui $p + q$ elementos.

$$n(A \cup B) = n(A) + n(B)$$

Caso geral: sejam A_1, A_2,..., A_n, conjuntos dois a dois disjuntos

$$n\left(A_i \cup A_j \cup \cdots \cup A_n\right) = n(A_i) + n\left(A_j\right) + \cdots + n(A_n)$$

Princípio de Multiplicação:

Se uma decisão d_1 pode ser tomada de n maneiras e se, uma decisão d_2 pode ser tomada de m maneiras, então o número total de maneiras possíveis de se tomarmos as decisões d_1 e d_2 é $n \times m$.

Técnicas de Contagem

Exemplo 1) Quantos números naturais de 5 algarismos (na base 10) que sejam maiores que 6.000 e que não sejam divisíveis por 5, podem ser formados usando-se apenas os algarismos 4, 5, 6, 7 e 8?

Primeiro algarismo: 3 modos (pode ser 6, 7 ou 8)

Último algarismo: 4 modos (não pode ser 5)

Segundo algarismo: 5 modos

Terceiro algarismo: 5 modos

Quarto algarismo: 5 modos

$$3 \times 4 \times 5^3 = 1500$$

```
num = 3 * 4 * (5**3) # potenciação **
print(numeros)
```

```
num = 3 * 4 * (5**3) # potenciação ** ou ^
print(numeros)
```

Princípio de Inclusão-Exclusão

Se A e B são dois conjuntos, não obrigatoriamente disjuntos, $n(A \cup B) = n(A) + n(B) - n(A \cap B)$

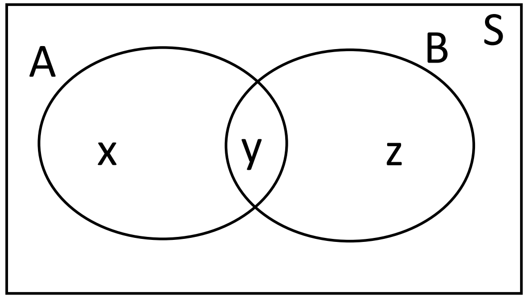
Seja:

y o número de elementos que pertencem a A e B.

x o número de elementos que pertencem a A e não pertencem a B.

z o número de elementos que pertencem a B e não pertencem a A.

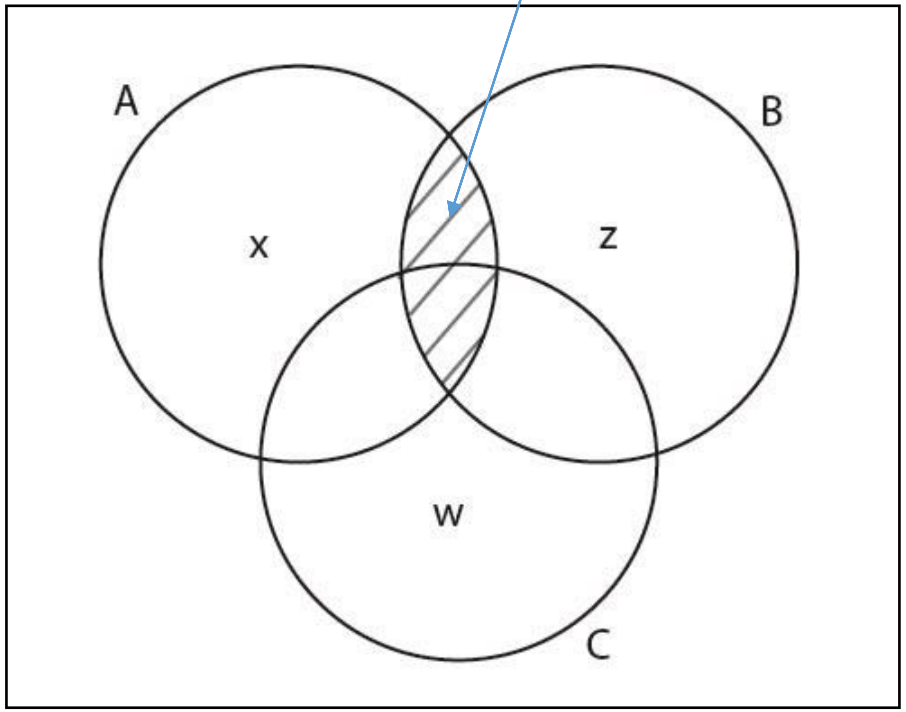
$$n(A \cup B) = n(A) + n(B) - n(A \cap B) = (x + y) + (y + z) - y = x + y + z$$



Princípio de Inclusão-Exclusão

Para 3 conjuntos temos

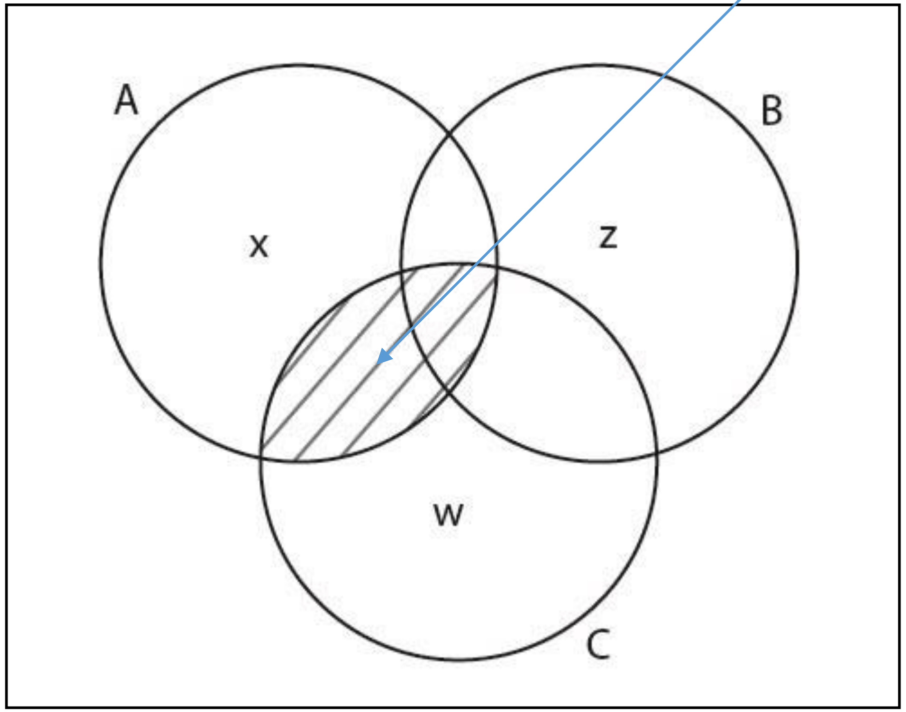
$$n(A \cup B \cup C) = n(A) + n(B) + n(C) - n(A \cap B) - n(A \cap C) - n(B \cap C) + n(A \cap B \cap C)$$



Princípio de Inclusão-Exclusão

Para 3 conjuntos temos

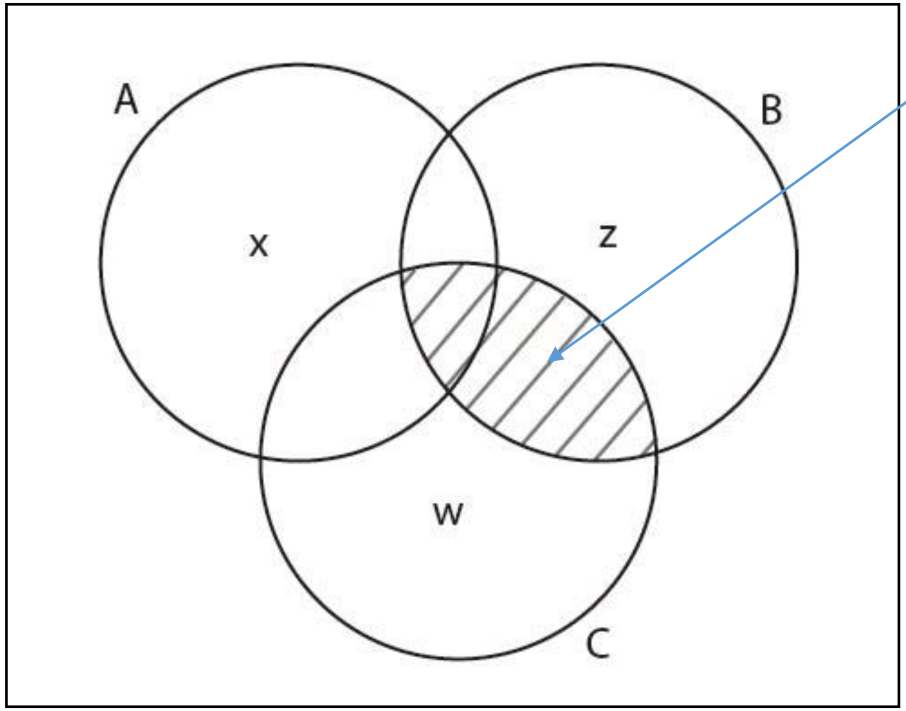
$$n(A \cup B \cup C) = n(A) + n(B) + n(C) - n(A \cap B) - n(A \cap C) - n(B \cap C) + n(A \cap B \cap C)$$



Princípio de Inclusão-Exclusão

Para 3 conjuntos temos

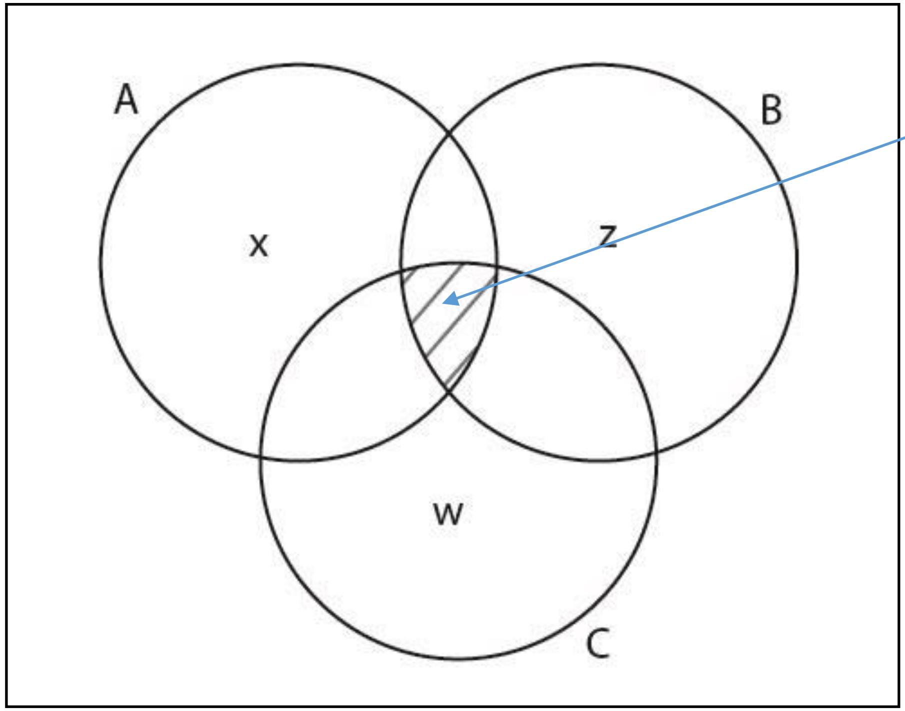
$$n(A \cup B \cup C) = n(A) + n(B) + n(C) - n(A \cap B) - n(A \cap C) - n(B \cap C) + n(A \cap B \cap C)$$



Princípio de Inclusão-Exclusão

Para 3 conjuntos temos

$$n(A \cup B \cup C) = n(A) + n(B) + n(C) - n(A \cap B) - n(A \cap C) - n(B \cap C) + n(A \cap B \cap C)$$



Princípio de Inclusão-Exclusão

Exemplo 1) Sejam $A = \{a, b, c, d, e\}$, $B = \{a, b, f, g, h\}$, $C = \{b, c, g, h, i, j\}$

$n(A) = 5$

$n(B) = 5$

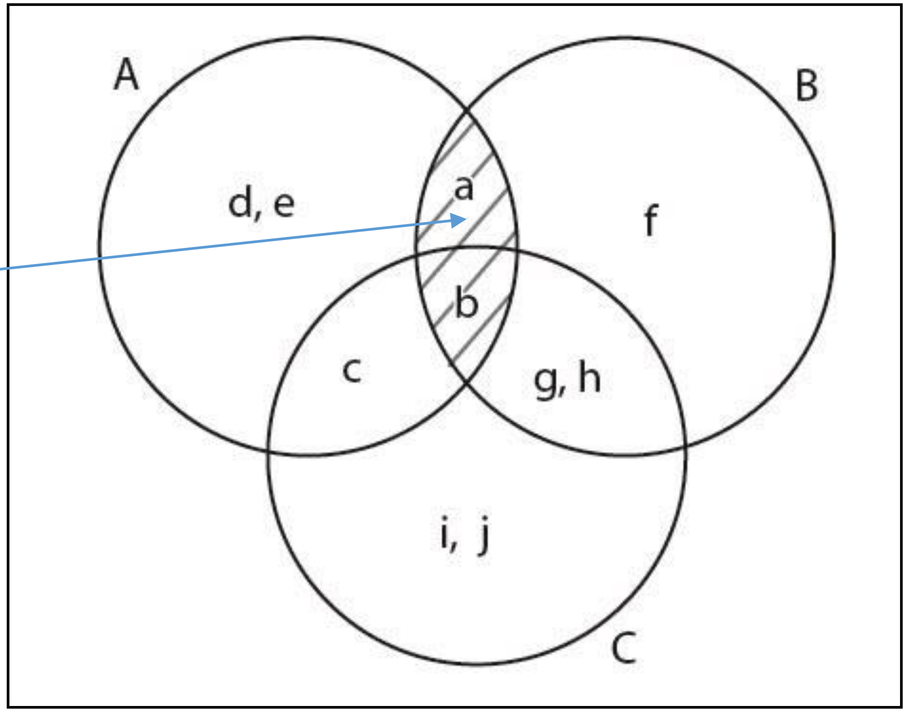
$n(C) = 6$

$n(A \cap B) = 2$

$n(A \cap C) = 2$

$n(B \cap C) = 3$

$n(A \cap B \cap C) = 1$



Princípio de Inclusão-Exclusão

Exemplo 1) Sejam $A = \{a, b, c, d, e\}$, $B = \{a, b, f, g, h\}$, $C = \{b, c, g, h, i, j\}$

$n(A) = 5$

$n(B) = 5$

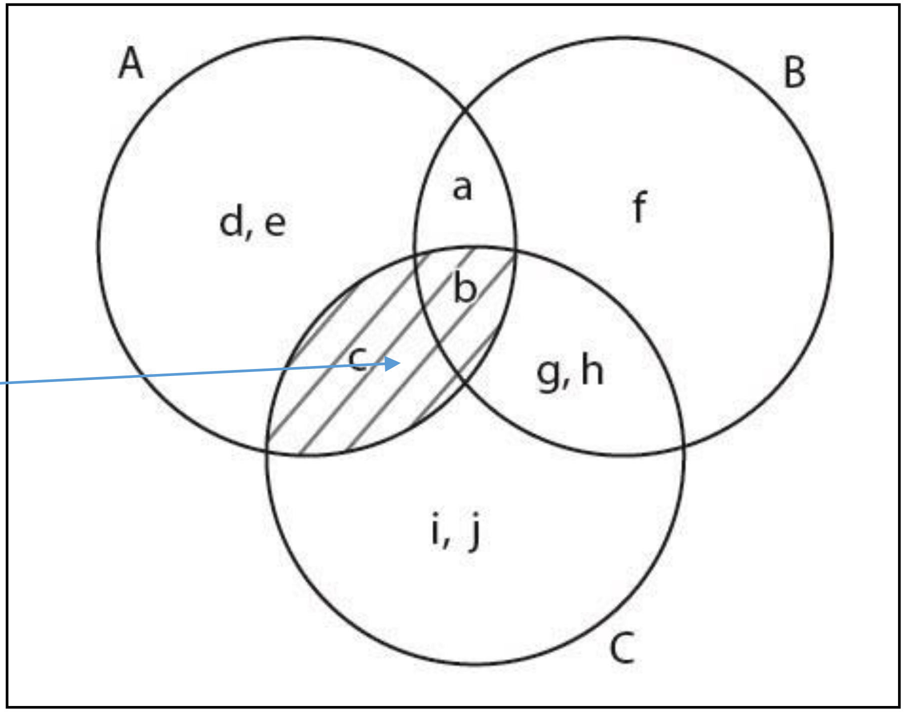
$n(C) = 6$

$n(A \cap B) = 2$

$n(A \cap C) = 2$

$n(B \cap C) = 3$

$n(A \cap B \cap C) = 1$



Princípio de Inclusão-Exclusão

Exemplo 1) Sejam $A = \{a, b, c, d, e\}$, $B = \{a, b, f, g, h\}$, $C = \{b, c, g, h, i, j\}$

$n(A) = 5$

$n(B) = 5$

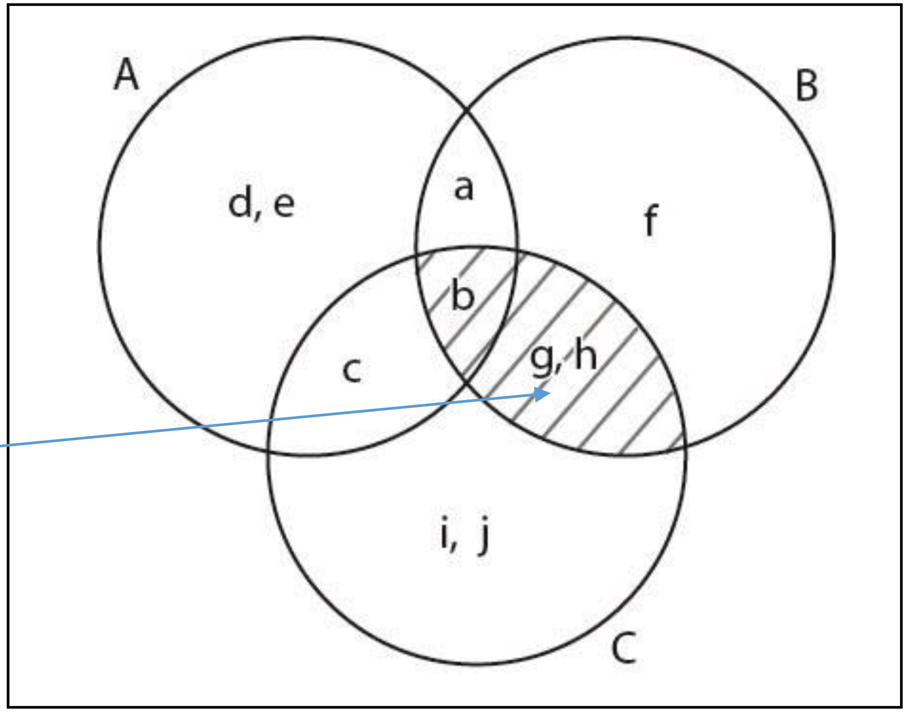
$n(C) = 6$

$n(A \cap B) = 2$

$n(A \cap C) = 2$

$n(B \cap C) = 3$

$n(A \cap B \cap C) = 1$



Princípio de Inclusão-Exclusão

Exemplo 1) Sejam $A = \{a, b, c, d, e\}$, $B = \{a, b, f, g, h\}$, $C = \{b, c, g, h, i, j\}$

$n(A) = 5$

$n(B) = 5$

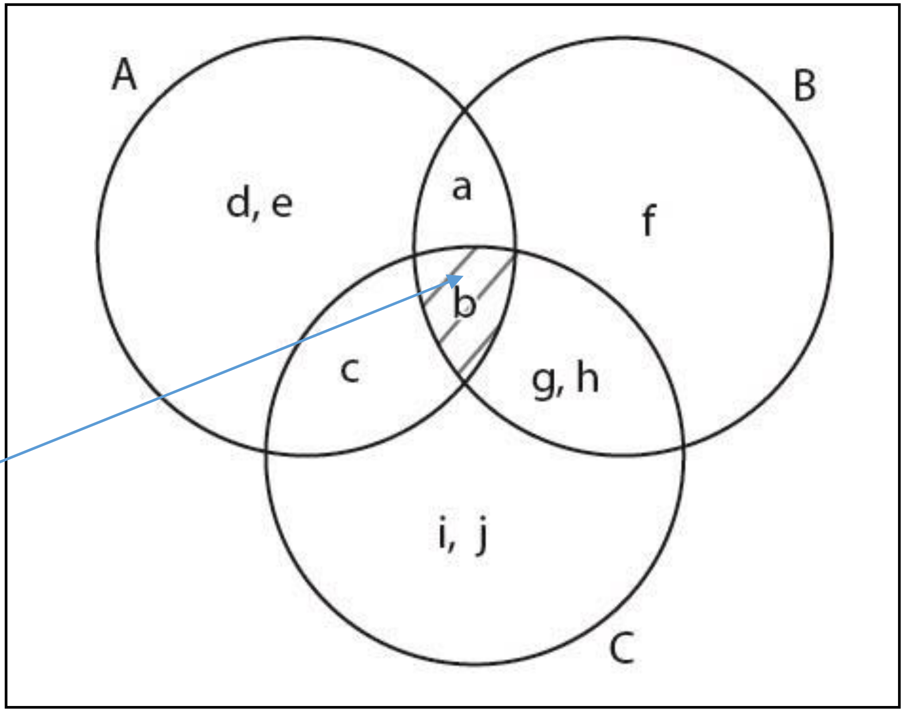
$n(C) = 6$

$n(A \cap B) = 2$

$n(A \cap C) = 2$

$n(B \cap C) = 3$

$n(A \cap B \cap C) = 1$



$n(A \cup B \cup C) = 5 + 5 + 6 - 2 - 2 - 3 + 1 = 10$

$A \cup B \cup C =$ { a, b, c, d, e , f , g, h, i, j}

Princípio de Inclusão-Exclusão

Se temos um conjunto S e n subconjuntos de S: $A_1, A_2, \dots A_n$ o principio de **Inclusão-Exclusão** pode ser generalizado. O número de elementos da união de n conjuntos quaisquer é dada por:

$$
\begin{aligned}
n\left(\bigcup_{i=1}^{n} A_i\right) = {} & nA_1 + \ nA_2 + \dots + nA_n \\
& -n(A_1 \cap A_2) - n(A_1 \cap A_3) - n(A_1 \cap A_4) - \dots - n(A_1 \cap A_n) \\
& -n(A_2 \cap A_3) - n(A_2 \cap A_4) - \dots - n(A_2 \cap A_n) \\
& \dots \\
& -n(A_{n-1} \cap A_n) \\
& +n(A_1 \cap A_2 \cap A_3) + n(A_1 \cap A_2 \cap A_4) + n(A_1 \cap A_2 \cap A_5) + \dots + n(A_1 \cap A_2 \cap A_n) \\
& +n(A_2 \cap A_3 \cap A_4) + n(A_2 \cap A_3 \cap A_5) + \dots + n(A_2 \cap A_3 \cap A_n) \\
& \dots \\
& +n(A_{n-2} \cap A_{n-1} \cap A_n) \\
& \dots \\
& +(-1)^{n-1} n(A_1 \cap A_2 \cap A_3 \cap \dots \cap A_n)
\end{aligned}
$$

Análise Combinatória:

- Permutações Simples
- Permutações com Repetição
- Permutações Circulares
- Permutações Caóticas
- Arranjos
- Arranjos com Repetição
- Combinações Simples
- Combinações Completas

Permutações Simples

Considere n objetos distintos $o_1, o_2, \dots, o_n$.

De quantos modos é possível ordená-los?

$$n(n-1)(n-2)\dots 1 = n!$$

O número de modos de ordenar n objetos distintos é representado por P_n.

$$P_n = n!$$

Cada ordenação possível dos n objetos é chamada de **permutação simples de n objetos**.

```
def fatorial(n):
    fat = 1
    i = 2
    while (i <= n):
        fat = fat * i
        i = i + 1
    return (fat)
```

```
def fatorial_recursivo(n):
    if (n == 0):
        return (1)
    else:
        return (n * fatorial_recursivo(n - 1))
```

Permutações Simples em R

Considere n objetos distintos $o_1, o_2, \dots, o_n$.

De quantos modos é possível ordená-los?

$$n(n-1)(n-2)\dots 1 = n!$$

O número de modos de ordenar n objetos distintos é representado por P_n.

$$P_n = n!$$

Cada ordenação possível dos n objetos é chamada de **permutação simples de n objetos**.

```
fatorial <- function(n){
  fat = 1
  i = 2
  while (i <= n){
    fat = fat * i
    i = i + 1
  }
  return (fat)
}
```

```
fatorial_recursivo <- function(n){
  if (n == 0)
    return (1)
  else
    return (n * fatorial_recursivo(n - 1))
}
```

Permutações Simples

Exemplo 1) Quantos são os anagramas da palavra PERMUTA?

$$P_7 = 7!$$

```
anagramas = fatorial(7)
print(anagramas)
```


```
anagramas = fatorial(7)
print(anagramas)
```


Exemplo 2) Quantos são os anagramas da palavra PERMUTA que começam e terminam por vogal?

Primeira letra: 3 modos

Última letra: 2 modos

As 5 letras restantes: 5! modos

3 x 2 x 5! = 720

```
anagramas = 3 * 2 * fatorial(5)
print(anagramas)
```


```
anagramas = 3 * 2 * fatorial(5)
print(anagramas)
```

Permutações com Repetição

Exemplo 1) Quantos são os anagramas da palavra PROBABILIDADE?

Se todas as letras fossem distintas formaríamos $P_{13} = 13!$ anagramas.

Como temos 5 letras distintas e as letras B, A, I e D aparecem 2 vezes cada, precisamos eliminar os anagramas repetidos:

$$P_{13}^{2,\,2,\,2,\,2} = \frac{13!}{2!\,2!\,2!\,2!} = 389.188.800$$

```
anagramas = fatorial(13) / (fatorial(2)**4)
print(anagramas)
```

Outra forma de resolver seria:

$$C_{13}^{2}\ .\ C_{11}^{2}\ .\ C_{9}^{2}\ .\ C_{7}^{2}\ .\ 5! = 389.188.800$$

```
def combinacao(n, p):
    return (fatorial(n) / (fatorial(p) * fatorial(n - p)))

anagramas = combinacao(13, 2)*combinacao(11, 2)*combinacao(9, 2)*combinacao(7, 2)*fatorial(5)
print(anagramas)
```

Permutações com Repetição em R

Exemplo 1) Quantos são os anagramas da palavra PROBABILIDADE?

Se todas as letras fossem distintas formaríamos $P_{13} = 13!$ anagramas.

Como temos 5 letras distintas e as letras B, A, I e D aparecem 2 vezes cada, precisamos eliminar os anagramas repetidos:

$$P_{13}^{2,\,2,\,2,\,2} = \frac{13!}{2!\,2!\,2!\,2!} = 389.188.800$$

```
anagramas = fatorial(13) / (fatorial(2)**4)
print(anagramas)
```

Outra forma de resolver seria:

$$C_{13}^{2} \,.\, C_{11}^{2} \,.\, C_{9}^{2} \,.\, C_{7}^{2} \,.\, 5! = 389.188.800$$

```
combinacao <- function(n, p){
  return (fatorial(n) / (fatorial(p) * fatorial(n - p)))
}
anagramas = combinacao(13, 2)*combinacao(11, 2)*combinacao(9, 2)*combinacao(7, 2)*fatorial(5)
print(anagramas)
```

Permutações com Repetição

No caso geral teríamos:

$$P_n^{n_1,n_2,\ldots,n_s} = C_n^{n_1} C_{n-n_1}^{n_2} \ldots C_{n-n_1-n_2-\cdots-n_s}^{n_s} = \frac{n!}{n_1!n_2!\ldots n_s!}$$

Também podemos utilizar a terminologia $i + j + k + \cdots + z = n$.

$$P_n^{i,j,k,\ldots z} = C_n^i C_{n-i}^j \ldots C_{n-i-j-k\ldots-z}^z = \frac{n!}{i!j!k!\ldots z!}$$

Permutações Circulares

De quantos modos podemos colocar n objetos distintos em n lugares equiespaçados em torno de um círculo? Vamos considerar equivalentes as disposições que possam coincidir por rotação.

O número de modos é presentados por $(PC)_n$.

$$(PC)_n = \frac{n!}{n} = (n-1)!$$

Uma permutação circular de n objetos distintos é qualquer distribuição desses n objetos em torno de um círculo.

Na permutação simples os lugares que os objetos ocupam importam.

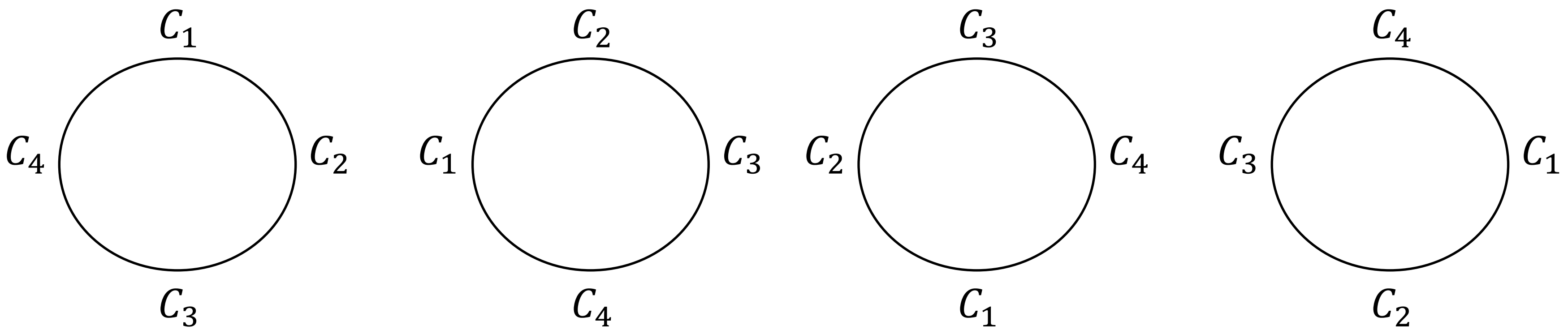
Na permutação circular apenas a posição relativa dos objetos entre si importa.

Permutações Circulares

Duas distribuições **circulares são iguais** se a posição relativa dos objetos é a mesma, ou seja, nenhuma pode ser obtida de outra por meio de uma rotação em torno do centro do círculo.

$$\begin{array}{|l|l|l|l} C_1 & C_2 & C_3 & C_4 \\ C_2 & C_3 & C_4 & C_1 \\ C_3 & C_4 & C_1 & C_2 \\ C_4 & C_1 & C_2 & C_3 \end{array}$$

4 permutações simples



A posição relativa é a mesma nos 4 círculos

Permutações Circulares

Exemplo 1) De quantos modos podemos formar uma roda de ciranda com 8 crianças:

a) De modo que 2 determinadas crianças não fiquem juntas?

Podemos formar $(PC)_6 = 5!$ com as 6 crianças que não apresentam restrição de lugar.

Há 6 modos de colocar a criança C na roda.

Após colocar a criança C há 5 modos de colocar a criança C' na roda.

$6 \times 5 \times (PC)_6 = 6 \times 5 \times 5! = 3600$

b) De modo que 2 determinadas crianças sempre fiquem juntas.

$2 \times (PC)_7 = 2 \times 6! = 1440$

Permutações Caóticas

Uma permutação de n elementos é caótica quando nenhum dos elementos está no seu lugar primitivo.

Ou seja, uma permutação dos objetos $o_1, o_2, \dots o_n$, inicialmente nessa ordem, é qualquer permutação desses objetos que não deixam nenhum deles na sua posição inicial.

Seja A_i o conjunto das permutações de $(1, 2, \dots n)$ em que o número i ocupa o i-ésimo lugar,

$i \in \{1, 2, \dots, n\}$

Queremos calcular o número de elementos do conjunto S que pertencem a exatamente zero dos conjuntos $A_1, A_2, \dots, A_n$.

$$S_0 = n(S) = n!;$$

$$S_1 = \sum_{i=1}^{n} (n(A_i)) = \sum_{i=1}^{n} (n - 1)! = n\,(n-1)! = n!\,;$$

$\dots$

Permutações Caóticas

$$S_0 = n(S) = n!;$$

$$S_1 = \sum_{i=1}^{n}(n(A_i)) = \sum_{i=1}^{n}(n-1)! = n\,(n-1)! = n!\,;$$

$$S_2 = \sum_{1 \le i < j \le n} n\left(A_i \cap A_j\right) = \sum_{1 \le i < j \le n}(n-2)! = C_n^2(n-2)! = \frac{n!}{2!};$$

$$S_3 = \sum_{1 \le i < j < k \le n} n\left(A_i \cap A_j \cap A_k\right) = \sum_{1 \le i < j < k \le n}(n-3)! = C_n^3(n-3)! = \frac{n!}{3!};$$

…

$$S_n = C_n^n(n-n)! = \frac{n!}{n!};$$

Permutações Caóticas

O número de elementos de S que pertencem a exatamente zero dos $A_1, A_2, \ldots, A_n$ é:

$$a_p = \sum_{k=0}^{n-p} (-1)^k C_{p+k}^{k} S_{p+k};$$

p = 0

$$a_0 = \sum_{k=0}^{n-0} (-1)^k C_{0+k}^{k} S_{0+k}$$

Ou seja:

$$\sum_{k=0}^{n} (-1)^k S_k;$$

$$= S_0 - S_1 + S_2 - S_3 + \ldots + (-1)^n S_n$$

$$= n! - n! + \frac{n!}{2!} - \frac{n!}{3!} + \ldots + (-1)^n \frac{n!}{n!}$$

$$= n! \left[\frac{1}{0!} - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \ldots + \frac{(-1)^n}{n!}\right]$$

Logo, o número de permutações caóticas de $(1, 2, \ldots, n)$ é:

$$D_n = n! \left[\frac{1}{0!} - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \ldots + \frac{(-1)^n}{n!}\right]$$

$$= n! \sum_{i=0}^{n} \frac{(-1)^i}{i!}$$

Permutações Caóticas em R

O número de elementos de S que pertencem a exatamente zero dos $A_1, A_2, \ldots, A_n$ é:

$$D_n = n!\left[\frac{1}{0!} - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \ldots + \frac{(-1)^n}{n!}\right] = n!\sum_{i=0}^{n}\frac{(-1)^i}{i!}$$

```r
permutacao_caotica <- function(n){
  retorno = 0
  for(i in 0:n){
    retorno = retorno + (((-1)^i)/fatorial(i))
  }
  retorno = fatorial(n) * retorno
  return(retorno)
}
```

```python
def permutacao_caotica(n):
    retorno = 0
    for i in range(n + 1):
        retorno = retorno + (((-1)**i)/fatorial(i))
    retorno = fatorial(n) * retorno
    return (retorno)
```

Permutações Caóticas

Exemplo 1) Quantas são as permutações das letras a, b, c e d?

$$P_4 = 4! = 24$$

Exemplo 2) Quantas são as permutações das letras a, b, c e d, nesta ordem, que:

a) deixem

i. a letra a fixa?

$a\ b\ c\ d$, logo, 3!

ii. a letra b fixa?

$a\ b\ c\ d$, logo, 3!

iii. a letra c fixa?

$a\ b\ c\ d$, logo, 3!

iv. a letra d fixa?

$a\ b\ c\ d$, logo, 3!

a b c d
a b d c
a c b d
a c d b
a d c b
a d b c

b) deixem as letras c e d fixas?

$a\ b\ c\ d$

2!

c) deixem as letra a ou a letra b fixa?

3! + 3! – 2!

Permutações Caóticas

Exemplo 1) Quantas são as permutações das letras a, b, c e d?

$$P_4 = 4! = 24$$

Exemplo 2) Quantas são as permutações das letras a, b, c e d, nesta ordem, que:

a) deixem

i. a letra a fixa?

$a\ b\ c\ d$, logo, 3!

ii. a letra b fixa?

$a\ b\ c\ d$, logo, 3!

iii. a letra c fixa?

$a\ b\ c\ d$, logo, 3!

iv. a letra d fixa?

$a\ b\ c\ d$, logo, 3!

a b c d
a b d c
a c b d
a c d b
a d c b
a d b c

a b c d
a b d c
c b a d
c b d a
d b a c
d b c a

a b c d
a b d c

b) deixem as letras c e d fixas?

$a\ b\ c\ d$

2!

c) deixem as letra a ou a letra b fixa?

3! + 3! – 2!

Permutações Caóticas

Exemplo 1) Quantas são as permutações das letras a, b, c e d?

$$P_4 = 4! = 24$$

Exemplo 2) Quantas são as permutações das letras a, b, c e d, nesta ordem, que:

a) deixem

i. a letra a fixa?

$a\ b\ c\ d$, logo, 3!

ii. a letra b fixa?

$a\ b\ c\ d$, logo, 3!

iii. a letra c fixa?

$a\ b\ c\ d$, logo, 3!

iv. a letra d fixa?

$a\ b\ c\ d$, logo, 3!

a b c d
a b d c
a c b d
a c d b
a d c b
a d b c

a b c d
a b d c
c b a d
c b d a
d b a c
d b c a

a b c d
a b d c

b) deixem as letras c e d fixas?

$a\ b\ c\ d$

2!

c) deixem as letra a ou a letra b fixa?

3! + 3! – 2!

Permutações Caóticas

d) deixem pelo menos um dos elementos fixos?

$$n(A_1 \cup A_2 \cup A_3 \cup A_4) = n(A_1)+n(A_2)+n(A_3)+n(A_4)$$
$$- n(A_1 \cap A_2) - n(A_1 \cap A_3) - n(A_1 \cap A_4) - n(A_2 \cap A_3) - n(A_2 \cap A_4) - n(A_3 \cap A_4)$$
$$+ n(A_1 \cap A_2 \cap A_3) + n(A_1 \cap A_2 \cap A_4) + n(A_1 \cap A_3 \cap A_4) + n(A_2 \cap A_3 \cap A_4)$$
$$- n(A_1 \cap A_2 \cap A_3 \cap A_4)$$

$$n(A_1 \cup A_2 \cup A_3 \cup A_4) = (C_1^4 3!) - (C_2^4 2!) + (C_3^4 1!) - (C_4^4 0!) = 15$$

e) não deixem nenhum algarismo fixo?

$$P_4 - 15 = 9$$

Permutações Caóticas

Exemplo 2) Seis casais participam de uma gincana. Em uma das etapas o grupo deve ser dividido em duplas sendo que não é permitido a formação de duplas de cônjuges. De quantas maneiras as duplas podem ser organizadas?

Se considerarmos os casais:

c_1, c_2, c_3, c_4, c_5 e c_6, então não são permitidas formações como:

$$\begin{pmatrix} c_{1a} & c_{2a} & c_{3a} & c_{4a} & c_{5a} & c_{6a} \\ c_{1b} & c_{2b} & c_{3b} & c_{4b} & c_{5b} & c_{6b} \end{pmatrix}, \begin{pmatrix} c_{1a} & c_{2a} & c_{3a} & c_{4a} & c_{5a} & c_{6a} \\ c_{1b} & c_{3b} & c_{2b} & c_{4b} & c_{5b} & c_{6b} \end{pmatrix} \text{ e } \begin{pmatrix} c_{1a} & c_{2a} & c_{3a} & c_{4a} & c_{5a} & c_{6a} \\ c_{2b} & c_{3b} & c_{1b} & c_{4b} & c_{6b} & c_{5b} \end{pmatrix}$$

$$D_6 = 6!\left(\frac{1}{0!} - \frac{1}{1!} + \frac{1}{2!} - \frac{1}{3!} + \frac{1}{4!} - \frac{1}{5!} + \frac{1}{6!}\right) = 265$$

```
duplas = permutacao_caotica(6)
print(duplas)
```

```
duplas = permutacao_caotica(6)
print(duplas)
```

Arranjos

Considere n objetos distintos $o_{\mathbf{1}}, o_{\mathbf{2}}, \ldots, o_{\boldsymbol{n}}$.

De quantos modos podemos ordenar p objetos, com $1 \leq p \leq n$?

$$n\,(n-1)\,(n-2)\;\ldots\;(n-p+1)$$

O número de modos de ordenar p objetos distintos é representado por $A_{n,p}$.

$$A_{n,p} = \frac{n!}{(n-p)!}$$

```python
def arranjo(n, p):
    return (fatorial(n) / fatorial(n - p))
```

```r
arranjo <- function(n, p){
  return (fatorial(n) / fatorial(n - p))
}
```

Arranjos

Exemplo 1) Uma pessoa possui 8 livros distintos do mesmo tamanho. Em uma prateleira é possível guardar apenas 5 deles. De quantos modos 5 dos 8 livros podem ser escolhidos e colocados em uma pilha na prateleira?

$$A_{8,5} = \frac{8!}{(8-5)!} = 6720$$

```
pilha = arranjo(8, 5)
print(pilha)
```


```
pilha = arranjo(8, 5)
print(pilha)
```

Arranjos com Repetição

Considere n objetos distintos $o_1, o_2, \ldots, o_n$.

De quantos modos podemos ordenar p objetos, com $1 \leq p \leq n$?

$$(AP)_{n,p} = n\ x\ n\ x \ldots x\ n = n^p$$

```
def arranjo_com_repeticao(n, p):
    return (n**p)
```

```
Arranjo_com_repeticao <- function(n, p){
  return (n^p)
}
```

Arranjos com Repetição

Exemplo 1) A senha de um sistema possui 5 dígitos sendo formada exclusivamente por números. Qual é a quantidade de senhas possíveis?

$$(AP)_{10,5} = 10^5 = 10000$$

```
possibilidades = arranjo_com_repeticao(10, 5)
print(possibilidades)
```


```
possibilidades = arranjo_com_repeticao(10, 5)
print(possibilidades)
```

Combinações Simples

De quantos modos podemos escolher p objetos distintos entre n objetos distintos?

O que equivale a perguntar:

Quantos são os subconjuntos com p elementos do conjunto $\{o_1, o_2, \ldots, o_n\}$?

$$C_n^p = \frac{n!}{p!(n-p)!}$$

Cada subconjunto de p elementos é uma combinação simples de classe p dos n objetos.

```
def combinacao(n, p):
    return (fatorial(n) / (fatorial(p) * fatorial(n - p)))
```

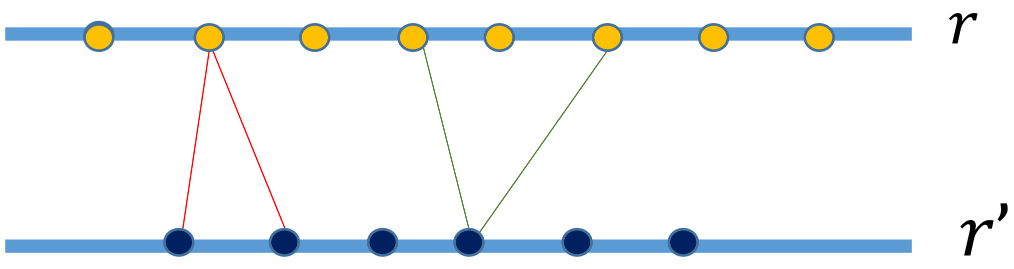
```
combinacao <- function(n, p) {
  return (fatorial(n) / (fatorial(p) * fatorial(n - p)))
}
```

Combinações Simples

Exemplo 1) Sejam r e r' duas retas paralelas. Marcam-se 8 pontos sobre a reta r e 6 pontos sobre a reta r'. Quantos triângulos existem com vértices em 3 desses 14 pontos?

Triângulos com um vértice em r e dois em r':

$$8\,C_6^2 = 8\left(\frac{6 \times 5}{2!}\right) = 120$$



Triângulos com um vértice em r' e dois em r:

$$6\,C_8^2 = 6\left(\frac{8 \times 7}{2!}\right) = 168$$

Resposta: $120 + 168 = 288$

```
triangulo = 8 * combinacao(6, 2) + 6 * combinacao(8, 2)
print(triangulo)
```


```
triangulo = 8 * combinacao(6, 2) + 6 * combinacao(8, 2)
print(triangulo)
```

Combinações Simples

Exemplo 2) De quantos modos é possível arrumar uma fila com 9 estudantes sendo:

- 3 de engenharia,
- 3 de administração, e
- 3 de direito,

de modo que não fiquem dois estudantes do mesmo curso juntos?

Combinações Simples

Exemplo 2)

Inicialmente vamos calcular quantas filas podemos formar ficando um aluno de engenharia e um de administração fixos no início da fila.

$$\frac{4!}{2!2!} = \frac{24}{4} = 6 \text{ possibilidades}$$

*EA*EEAA

*EA*EAEA

*EA*AEAE

*EA*AEEA

*EA*AAEE

*EA*EAAE

Combinações Simples

Exemplo 2)

Considere abaixo que:

. indica um local na fila no qual é possível incluir um aluno de direito, e

_ indica um local na fila no qual é necessário incluir um aluno de direito.

EA . E_E . A_A .

EA . E . A . E . A.

*EA*_A . E . A . E.

*EA*_A . E_E . A .

*EA*_A_A . E_E .

EA . E . A_A . E.

Combinações Simples

Exemplo 2)

EA . E_E . A_A .	C_3^1 = 3
EA . E . A . E . A.	C_5^3 = 10
*EA*_A . E . A . E.	C_4^2 = 6
*EA*_A . E_E . A .	C_3^1 = 3
*EA*_A_A . E_E .	C_2^0 = 1
EA . E . A_A . E.	C_4^2 = 6

O aluno de direito pode ser encaixado em cada uma dessas filas de 29 modos.

Logo temos 174 (6 x 29) possíveis filas ficando um aluno de engenharia e um de administração fixos no início da fila.

Combinações Simples

Exemplo 2)

Podemos replicar o raciocínio para as filas começando com AE, DA, DE, AD e ED.

O número de filas será 174 sem contar as permutações entre os diferentes alunos do mesmo curso que será 3!

Logo teremos:

174 $(3!)^3$ = 37.584 modos possíveis.

Combinações Complementares

Para cada combinação de p elementos a partir de um conjunto de n elementos é possível formar com os $n - p$ elemento restantes uma combinação complementar tal que:

$$C_n^p = C_n^{n-p}$$

```python
def combinacao_complementar(n, p):
    nc = n
    pc = n - p
    return (fatorial(nc) / (fatorial(pc) * fatorial(nc - pc)))
```

```r
combinacao_complementar <- function(n, p) {
  nc = n
  pc = n - p
  return (fatorial(nc) / (fatorial(pc) * fatorial(nc - pc)))
}
```

Combinações Completas (com Repetição)

De quantos modos podemos escolher p objetos, distintos ou não, entre n objetos distintos?

Outra forma de perguntar seria:

Qual o número de soluções para a equação $x_1 + x_2 + \ldots + x_n = p$ em inteiros não negativos?

$$CR_n^p = C_{n+p-1}^p$$

```
def combinacao_completa(n, p):
    nc = n + p - 1
    pc = p
    return (fatorial(nc) / (fatorial(pc) * fatorial(nc - pc)))
```

```
combinacao_completa <- function(n, p) {
  nc = n + p - 1
  pc = p
  return (fatorial(nc) / (fatorial(pc) * fatorial(nc - pc)))
}
```

Combinações Completas

Exemplo 1) De quantos modos é possível comprar 2 computadores em uma loja que oferece 3 modelos (todos com 2 unidades ou mais disponíveis)

x_1		x_2		x_3
CC	|	0	|	0
0	|	CC	|	0
0	|	0	|	CC
C	|	C	|	0
C	|	0	|	C
0	|	C	|	C

O total de incógnitas é 2 já que $\boldsymbol{x_1} + \boldsymbol{x_2} + \boldsymbol{x_3} = 2$

O total de traços é 2.

O modo de arrumar em fila seria $P_4^{2,\,2} = \frac{4!}{2!2!} = C_4^2$

No caso geral temos p computadores e n – 1 traços, logo:

$$CR_n^p = P_{p+n-1}^{p,\,n\,-1} = \frac{(n+p-1)!}{p!(n-1)!} = C_{n+p-1}^p$$

```
computador = combinacao_completa(3, 2)
print(computador)
```

```
computador = combinacao_completa(3, 2)
print(computador)
```

Generalização do Princípio de Inclusão-Exclusão

Agora que já vimos combinação é mais fácil formalizar a Generalização do Princípio de Inclusão-Exclusão.

Sejam S um conjunto, $A_1, A_2, \ldots, A_n$ subconjuntos de S e

$$S_0 = n(S);$$

$$S_1 = \sum_{i=1}^{n} (n(A_i));$$

$$S_2 = \sum_{1 \leq i < j \leq n} n\left(A_i \cap A_j\right);$$

$$S_3 = \sum_{1 \leq i < j < k \leq n} n\left(A_i \cap A_j \cap A_k\right);$$

$$\vdots$$

(Há C_n^1 parcelas S_1, C_n^2 parcelas em S_2 etc...)

Generalização do Princípio de Inclusão-Exclusão

Sejam S um conjunto, $A_1, A_2, \dots, A_n$ subconjuntos de S e

$$S_0 = n(S);$$

$$S_1 = \sum_{i=1}^{n} (n(A_i));$$

$$S_2 = \sum_{1 \le i < j \le n} n\left(A_i \cap A_j\right);$$

$$S_3 = \sum_{1 \le i < j < k \le n} n\left(A_i \cap A_j \cap A_k\right);$$

$$\vdots$$

Então:

a) O número de elementos de S que pertencem a **exatamente** $p(p \le n)$ dos conjuntos A_1, $A_2, \dots, A_n$ é

$$a_p = \sum_{k=0}^{n-p} (-1)^k C_{p+k}^{k} S_{p+k};$$

Generalização do Princípio de Inclusão-Exclusão

Sejam S um conjunto, $A_1, A_2, \dots, A_n$ subconjuntos de S e

$$S_0 = n(S);$$

$$S_1 = \sum_{i=1}^{n} (n(A_i));$$

$$S_2 = \sum_{1 \le i < j \le n} n(A_i \cap A_j);$$

$$S_3 = \sum_{1 \le i < j < k \le n} n(A_i \cap A_j \cap A_k);$$

$$\vdots$$

Então:

b) O número de elementos de S que pertencem **a pelo menos** $p(p \le n)$ dos conjuntos A_1, A_2, ..., A_n é

$$b_p = \sum_{k=0}^{n-p} (-1)^k C_{p+k-1}^{k} S_{p+k};$$

Generalização do Princípio de Inclusão-Exclusão

Sejam S um conjunto, $A_1, A_2, \ldots, A_n$ subconjuntos de S e

$$S_0 = n(S);$$

$$S_1 = \sum_{i=1}^{n}(n(A_i));$$

$$S_2 = \sum_{1 \leq i < j \leq n} n(A_i \cap A_j);$$

$$S_3 = \sum_{1 \leq i < j < k \leq n} n(A_i \cap A_j \cap A_k);$$

$$\vdots$$

Então:

c) O número de elementos do conjunto $A_1 \cup A_2 \cup \ldots A_n$ é

$$S_1 - S_2 + \ldots + (-1)^{n-1} S_n.$$

Exemplo 1) Quantos são os inteiros entre 1 e 10.000 (inclusive) que:

a) são divisíveis por exatamente dois dos números $2, 3, 5$ e 7.

Define-se

$$S = \{x \in \mathbb{N} \mid 1 \leq x \leq 10.000\}$$

$$A_1 = \{x \in S \mid 2 \; divide \; x\}$$

$$A_2 = \{x \in S \mid 3 \; divide \; x\}$$

$$A_3 = \{x \in S \mid 5 \; divide \; x\}$$

$$A_4 = \{x \in S \mid 7 \; divide \; x\}$$

Obs: Denotaremos a parte inteira de x por $\lfloor x \rfloor$

Generalização do Princípio de Inclusão-Exclusão

S_0 = n(S) = 10.000

S_1 = n(A_1) + n(A_2) + n(A_3) + n(A_4) =

$= \left\lfloor \frac{10.000}{2} \right\rfloor + \left\lfloor \frac{10.000}{3} \right\rfloor + \left\lfloor \frac{10.000}{5} \right\rfloor + \left\lfloor \frac{10.000}{7} \right\rfloor$ = 5000 + 3333 + 2000 + 1428 = 11.761

S_2 = n($A_1 \cap A_2$) + n($A_1 \cap A_3$) + n($A_1 \cap A_4$) + n($A_2 \cap A_3$) + n($A_2 \cap A_4$) + n($A_3 \cap A_4$) =

$\left\lfloor \frac{10.000}{6} \right\rfloor + \left\lfloor \frac{10.000}{10} \right\rfloor + \left\lfloor \frac{10.000}{14} \right\rfloor + \left\lfloor \frac{10.000}{15} \right\rfloor + \left\lfloor \frac{10.000}{21} \right\rfloor + \left\lfloor \frac{10.000}{35} \right\rfloor$ = 1666 + 1000 + 714 + 666 + 476 + 285 = 4807

S_3 = n($A_1 \cap A_2 \cap A_3$) + n($A_1 \cap A_2 \cap A_4$) + n($A_1 \cap A_3 \cap A_4$) + n($A_2 \cap A_3 \cap A_4$) =

$\left\lfloor \frac{10.000}{30} \right\rfloor + \left\lfloor \frac{10.000}{42} \right\rfloor + \left\lfloor \frac{10.000}{70} \right\rfloor + \left\lfloor \frac{10.000}{105} \right\rfloor$ = 333 + 238 + 142 + 95 = 808

S_4 = n($A_1 \cap A_2 \cap A_3 \cap A_4$) = $\left\lfloor \frac{10.000}{210} \right\rfloor$ = 47

a) são divisíveis por exatamente dois dos números 2, 3, 5 e 7.

$$a_2 = \sum_{k=0}^{4-2} (-1)^k \, C_{2+k}^{k} \, S_{2+k} = (-1)^0 C_2^0 S_2 + (-1)^1 C_3^1 S_3 + (-1)^2 C_4^2 S_4$$

Como:

S_2 = 4807

S_3 = 808

S_4 = 47

Então:

$a_2 =$ 4807 – (3 x 808) + (6 x 47) = 2665

Generalização do Princípio de Inclusão-Exclusão

Exemplo 1) Quantos são os inteiros entre 1 e 10.000 (inclusive) que:

b) são divisíveis por no mínimo dois dos números 2, 3, 5 e 7?

Em outras palavras queremos calcular o número de elementos que pertencem a pelo menos dois dos conjuntos $A_1, A_2, A_3\ e\ A_4$.

$$b_2 = \sum_{k=0}^{4-2} (-1)^k\, C_{2+k-1}^{k}\, S_{2+k} \;= (-1)^0\, C_1^0 S_2 + (-1)^1 C_2^1 S_3 + (-1)^2 C_3^2 S_4$$

$=$ 4807 – (2 x 808) + (3 x 2665) = 3332

Conjunto Potência

O conjunto potência de um conjunto A, denotado por $\mathcal{P}(A)$ é o conjunto cujos membros são todos os possíveis subconjuntos de A.

Exemplos: Se $A = \{x, y\}$, então $\mathcal{P}(A) = \{\emptyset, \{x\}, \{y\}, \{x, y\}\}$.

Se $B = \{x, y, z\}$, então $\mathcal{P}(B) =$ {∅, {x}, {y}, {x, y}, {z}, {x, z}, {y, z}, {x, y, z} }

O número de subconjuntos é 2^n, onde n é número de elementos do conjunto.

Conjunto Potência

O conjunto potência de um conjunto A, denotado por $\mathcal{P}(A)$ é o conjunto cujos membros são todos os possíveis subconjuntos de A.

Exemplos: Se $A = \{x, y\}$, então $\mathcal{P}(A) = \{\emptyset, \{x\}, \{y\}, \{x, y\}\}$.

Se $B = \{x, y, z\}$, então $\mathcal{P}(B) = \{\emptyset$, {x}, {y}, {x, y}, {z}, {x, z}, {y, z}, {x, y, z} }

Na linha $return\, r\ +\ [s\ +\ [c[-1]]\ for\ s\ in\ r]\ ,r$ armazena a lista obtida como resultado da chamada recursiva.

```
A = ['x', 'y']

def potencia_recursiva(c):
    if (len(c) == 0): # Caso base deve retornar um conjunto que contenha o conjunto vazio
        return [[]] # [[]]é uma lista que contém a lista vazia
    r = potencia_recursiva(c[:-1]) # Chamada recursiva removendo o último elemento
    return r + [s + [c[-1]] for s in r] # s representa todos os subconjuntos de r

P = potencia_recursiva(A)
print(P)
```

Conjunto Potência

O conjunto potência de um conjunto A, denotado por $\mathcal{P}(A)$ é o conjunto cujos membros são todos os possíveis subconjuntos de A.

Exemplos: Se $A = \{x, y\}$, então $\mathcal{P}(A) = \{\emptyset, \{x\}, \{y\}, \{x, y\}\}$.

Se $B = \{x, y, z\}$, então $\mathcal{P}(B) = \{\emptyset$, {x}, {y}, {x, y}, {z}, {x, z}, {y, z}, {x, y, z} }

A expressão [… $for\ s\ in\ r$] é uma compressão de lista que permite que cada subconjunto s contido no conjunto r seja processado.

```
A = ['x', 'y']

def potencia_recursiva(c):
   if (len(c) == 0): # Caso base deve retornar um conjunto que contenha o conjunto vazio
      return [[]] # [[]]é uma lista que contém a lista vazia
   r = potencia_recursiva(c[:-1]) # Chamada recursiva removendo o último elemento
   return r + [s + [c[-1]] for s in r] # s representa todos os subconjuntos de r

P = potencia_recursiva(A)
print(P)
```

Conjunto Potência

O conjunto potência de um conjunto A, denotado por $\mathcal{P}(A)$ é o conjunto cujos membros são todos os possíveis subconjuntos de A.

Exemplos: Se $A = \{x, y\}$, então $\mathcal{P}(A) = \{\emptyset, \{x\}, \{y\}, \{x, y\}\}$.

Se $B = \{x, y, z\}$, então $\mathcal{P}(B) =$ {∅, {x}, {y}, {x, y}, {z}, {x, z}, {y, z}, {x, y, z} }

A expressão s + [c [−1]] na compressão de lista adiciona o último elemento do conjunto c (o elemento do índice −1) a cada subconjunto s.

```
A = ['x', 'y']

def potencia_recursiva(c):
   if (len(c) == 0): # Caso base deve retornar um conjunto que contenha o conjunto vazio
      return [[]] # [[]]é uma lista que contém a lista vazia
   r = potencia_recursiva(c[:-1]) # Chamada recursiva removendo o último elemento
   return r + [s + [c[-1]] for s in r] # s representa todos os subconjuntos de r

P = potencia_recursiva(A)
print(P)
```

Conjunto Potência

O conjunto potência de um conjunto A, denotado por $\mathcal{P}(A)$ é o conjunto cujos membros são todos os possíveis subconjuntos de A.

Exemplos: Se $A = \{x, y\}$, então $\mathcal{P}(A) = \{\emptyset, \{x\}, \{y\}, \{x, y\}\}$.

Se $B = \{x, y, z\}$, então $\mathcal{P}(B) =$ {∅, {x}, {y}, {x, y}, {z}, {x, z}, {y, z}, {x, y, z} }

A expressão completa r + [...] concatena as duas metades do resultado descrito nas observações 2 e 3.

```
A = ['x', 'y']

def potencia_recursiva(c):
   if (len(c) == 0): # Caso base deve retornar um conjunto que contenha o conjunto vazio
      return [[]] # [[]]é uma lista que contém a lista vazia
   r = potencia_recursiva(c[:-1]) # Chamada recursiva removendo o último elemento
   return r + [s + [c[-1]] for s in r] # s representa todos os subconjuntos de r

P = potencia_recursiva(A)
print(P)
```

Conjunto Potência

Observações da chamada recursiva:

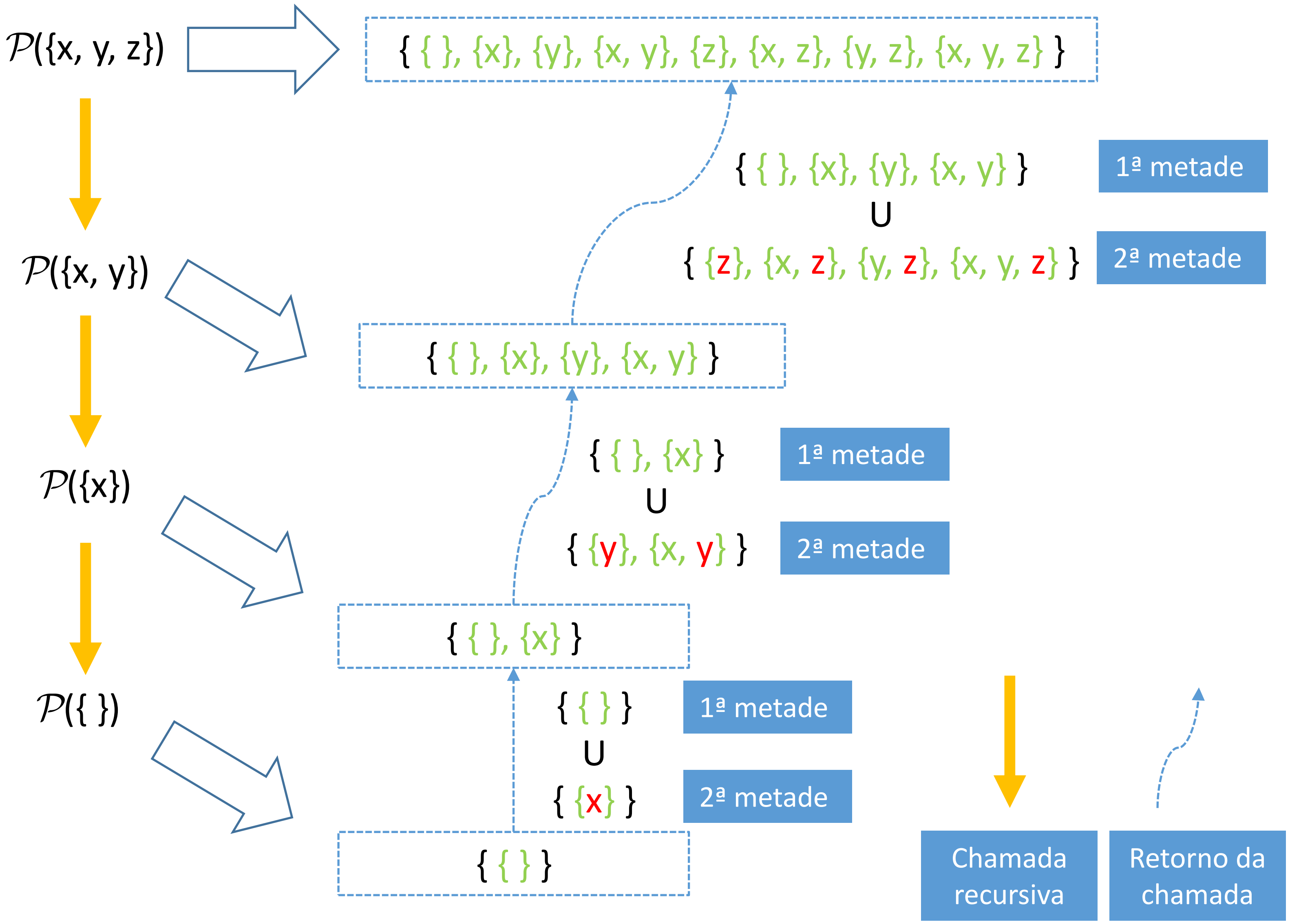
1. O resultado de $\mathcal{P}(\{x, y, z\})$ possui o dobro de elementos de $\mathcal{P}(\{x, y\})$.

2. A primeira metade do resultado de $\mathcal{P}(\{x, y, z\})$ é composta pelos mesmos elementos do resultado de $\mathcal{P}(\{x, y\})$.

3. A segunda metade do resultado de $\mathcal{P}(\{x, y, z\})$ é composta pelos mesmos elementos do resultado de $\mathcal{P}(\{x, y\})$ mas tendo a adição ao final de cada subconjunto do elemento z.

4. As observações acima também se aplicam aos demais conjuntos potência.

5. Analisando recursivamente, a cada rodada devemos fornecer como parâmetro da função recursiva um conjunto que contenha os elementos atuais menos o último elemento e o retorno da função deve ser um conjunto formado pelas duas metades indicadas nas observações 2 e 3.



Conjunto Potência

O conjunto potência de um conjunto A, denotado por $\mathcal{P}(A)$ é o conjunto cujos membros são todos os possíveis subconjuntos de A.

Exemplos: Se $A = \{x, y\}$, então $\mathcal{P}(A) = \{\emptyset, \{x\}, \{y\}, \{x, y\}\}$.

Se $B = \{x, y, z\}$, então $\mathcal{P}(B) = \{\emptyset$, {x}, {y}, {x, y}, {z}, {x, z}, {y, z}, {x, y, z} }

```
A = ['x', 'y']

def potencia_recursiva(c):
   if (len(c) == 0): # Caso base deve retornar um conjunto que contenha o conjunto vazio
      return [[]] # [[]]é uma lista que contém a lista vazia
   r = potencia_recursiva(c[:-1]) # Chamada recursiva removendo o último elemento
   return r + [s + [c[-1]] for s in r] # s representa todos os subconjuntos de r

P = potencia_recursiva(A)
print(P)
```

Conjunto Potência em R

O conjunto potência de um conjunto A, denotado por $\mathcal{P}(A)$ é o conjunto cujos membros são todos os possíveis subconjuntos de A.

Exemplo: Se $A = \{x, y\}$, então $\mathcal{P}(A) = \{\emptyset, \{x\}, \{y\}, \{x, y\}\}$.

```
potencia_recursiva <- function (posicao, vetor, vetoruso) {
  subconjunto <- NULL
  if ((posicao - 1) == length(vetor)) { # Montamos o subconjunto
    for (i in 1:length(vetor)) # Escrevemos o subconjunto
      if (vetoruso[i])
        subconjunto <- c(subconjunto, vetor[i])
    print(subconjunto)
  } else { # Se não terminamos, continuar a gerar
    vetoruso[posicao] = T # Subconjuntos que incluem o elemento corrente
    potencia_recursiva(posicao + 1, vetor, vetoruso) # Chamada recursiva
    vetoruso[posicao] = F # Subconjuntos que não incluem o elemento corrente
    potencia_recursiva(posicao + 1, vetor, vetoruso) # Chamada recursiva
  }
}
A <- c("x","y")
uso <- rep(F, length(A))
potencia_recursiva (1, A, uso)
```

Conjunto Potência em R

Suponha $A = \{y, z\}$ e seja a inclusão no conjunto representada por um vetor de booleanos:

Exemplos: [T, T] representa $\{y, z\}$; [T, F] representa $\{y\}$

Então todos os subconjuntos são:

- Vetores nos quais a 1ª posição é T ∪ todos os subconjuntos seguintes

+

- Vetores nos quais a 1ª posição é F ∪ todos os subconjuntos seguintes.

[T, T] = $\{y, z\}$

[T, F] = $\{y\}$

$\{y\}$ ∪ subconjuntos de $\{z\}$ (que são $\{z\}$ e $\{\}$) $\rightarrow \{y, z\}\ e\ \{y\}$

[F, T] = $\{z\}$

[F, F] = $\{\}$

$\{\}$ ∪ subconjuntos de $\{z\}$ (que são $\{z\}$ e $\{\}$) $\rightarrow \{z\}\ e\ \{\}$

Conjunto Potência em R

Seja $A = \{x, y, z\}$, então todos os subconjuntos são:

- Vetores nos quais a 1ª posição é T ∪ todos os subconjuntos seguintes

+

- Vetores nos quais a 1ª posição é F ∪ todos os subconjuntos seguintes.

[T,T,T] = $\{x, y, z\}$

[T,T,F] = $\{x, y\}$

[T,F,T] = $\{x, z\}$

[T,F,F] = $\{x\}$

$\{x\}$ ∪ subconjuntos de $\{y, z\}$ (que são $\{y, z\}$, $\{y\}$, $\{z\}$ e $\{\}$)
$\rightarrow \{x, y, z\}, \{x, y\}, \{x, z\}\ e\ \{x\}$

Observe a recursividade para o exemplo anterior com $A = \{y, z\}$.

[F,T,T] = $\{y, z\}$

[F,T,F] = $\{y\}$

[F,F,T] = $\{z\}$

[F,F,F] = $\{\}$

$\{\}$ ∪ subconjuntos de $\{y, z\}$ (que são $\{y, z\}$, $\{y\}$, $\{z\}$ e $\{\}$)
$\rightarrow \{y, z\}, \{y\}, \{z\}\ e\ \{\}$

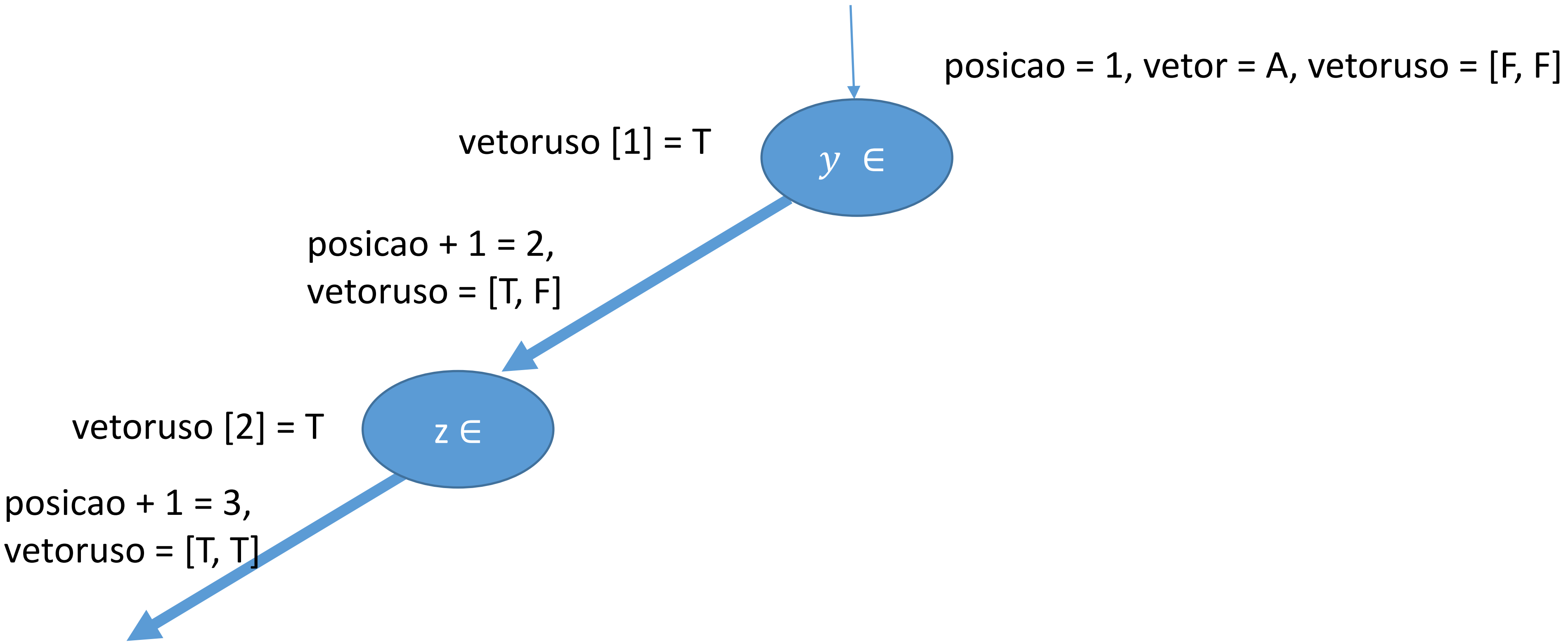
Conjunto Potência em R

Aproveitando o que foi exposto e visualizando o problema através de uma estrutura de árvore, temos:

- deve-se construir um vetor de *boolean* (T ou F) com o mesmo tamanho do vetor A;
- para $A = \{y, z\}$, teríamos um vetor inicial com $\{F, F\}$; e para $A = \{x, y, z\}$, teríamos um vetor inicial com $\{F, F, F\}$;
- o vetor de *boolean* ($vetoruso$) irá ajudar na construção de todos os subconjuntos;
- $vetoruso[i]$ indica se o i-ésimo elemento está ou não presente no subconjunto;
- a variável chamada $posicao$ irá indicar qual elemento de A será colocado ou removido do subconjunto;
- **potencia_recursiva** é a função que irá gerar os subconjuntos com a seguinte assinatura $potencia\_recursiva\ (posicao, vetor, vetoruso)$;
- **potencia_recursiva** é inicialmente chamada com $posicao = 1$, $vetor = A$ e $vetoruso$ com seu valor inicial.

Conjunto Potência em R

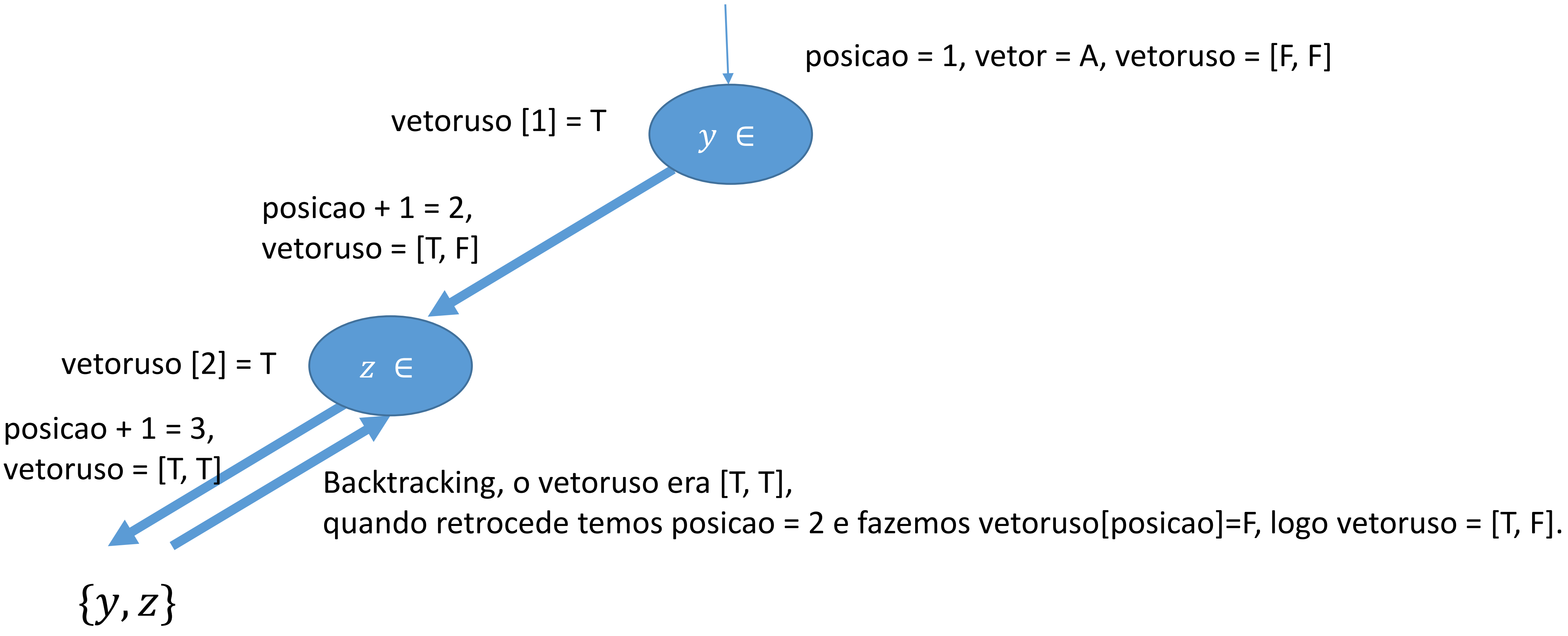
Construindo a árvore para $A = \{y, z\}$ e gerando os subconjuntos $\mathcal{P}$:



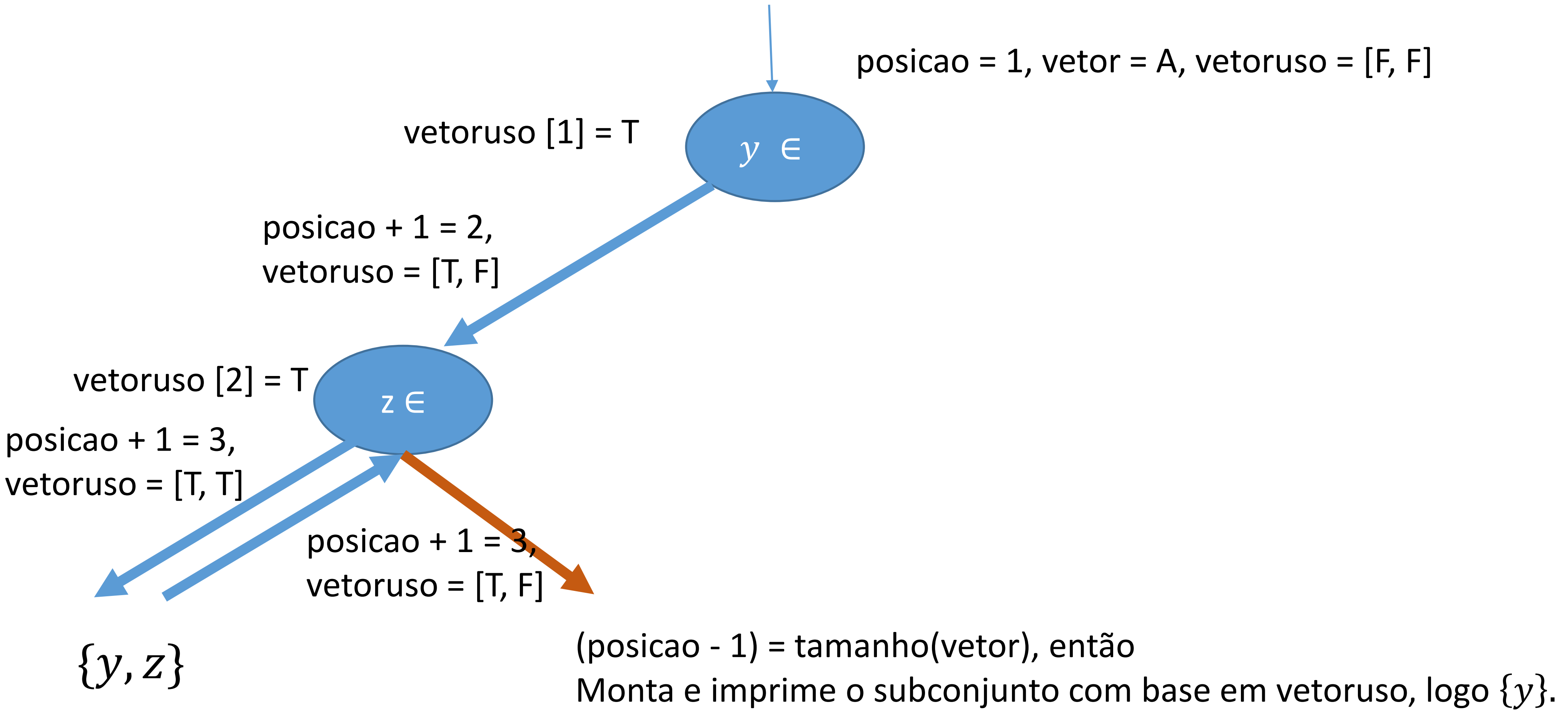
(posicao – 1) = tamanho(vetor), então
Monta e imprime o subconjunto com base em vetoruso, logo $\{y, z\}$.

Conjunto Potência em R

Construindo a árvore para $A = \{y, z\}$ e gerando os subconjuntos $\mathcal{P}$:

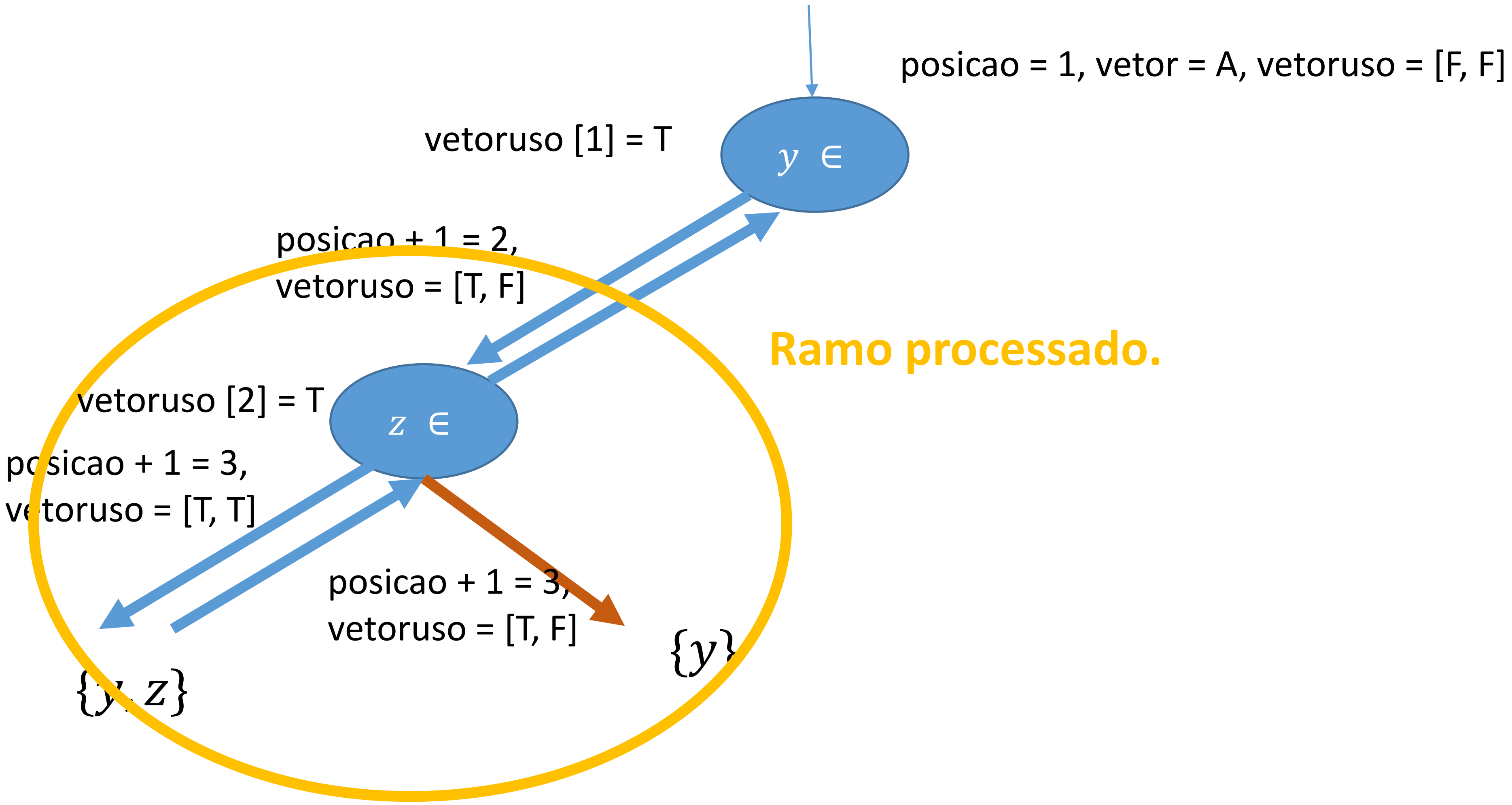


Construindo a árvore para $A = \{y, z\}$ e gerando os subconjuntos $\mathcal{P}$:



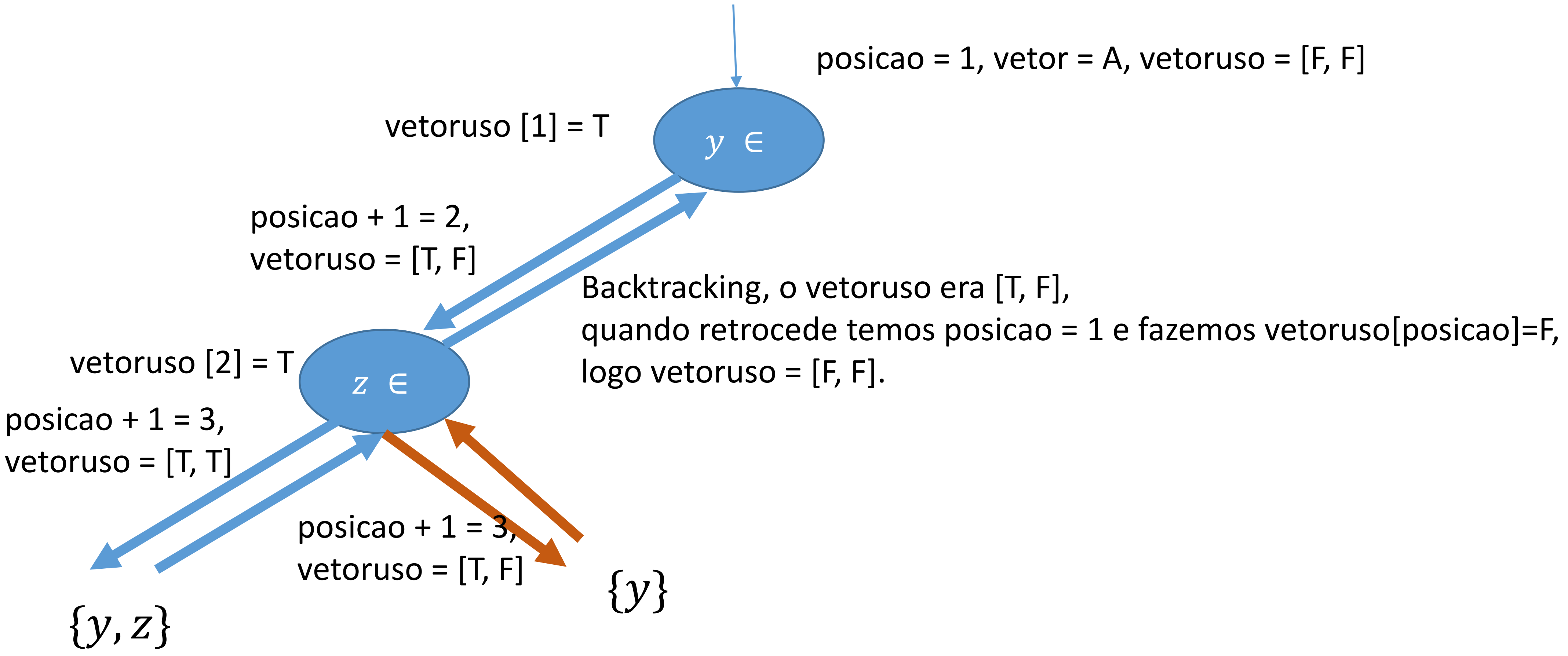
Conjunto Potência em R

Construindo a árvore para $A = \{y, z\}$ e gerando os subconjuntos $\mathcal{P}$:



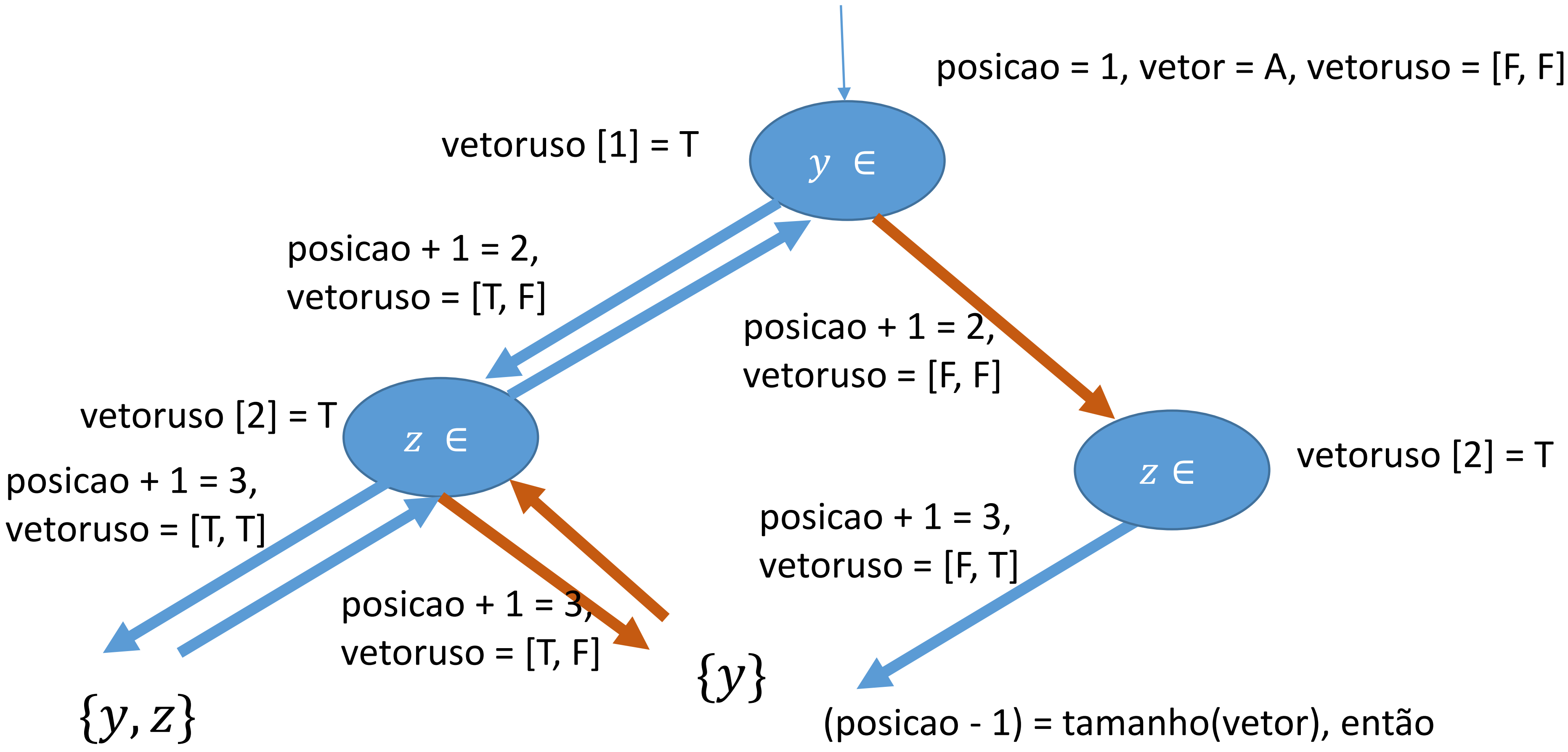
Conjunto Potência em R

Construindo a árvore para $A = \{y, z\}$ e gerando os subconjuntos $\mathcal{P}$:



Conjunto Potência em R

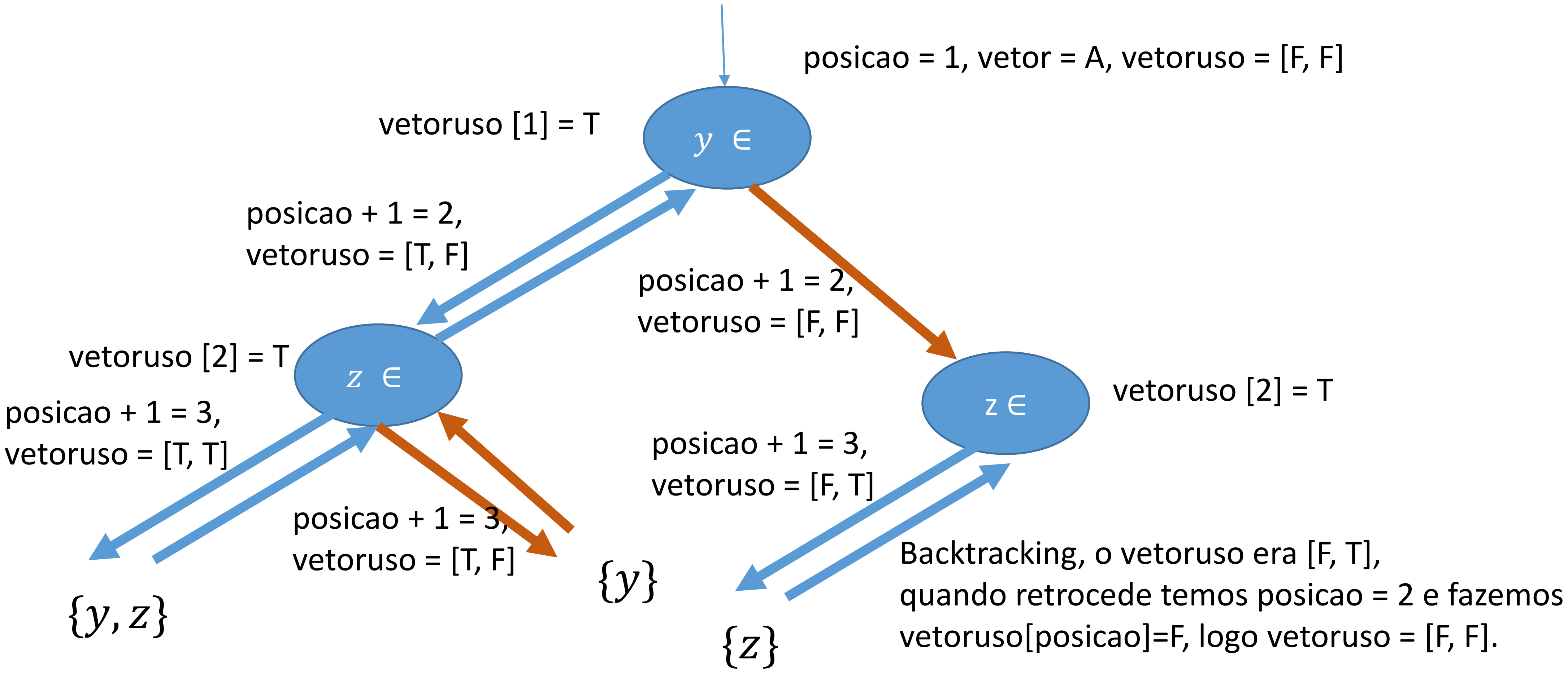
Construindo a árvore para $A = \{y, z\}$ e gerando os subconjuntos $\mathcal{P}$:



(posicao - 1) = tamanho(vetor), então
Monta e imprime o subconjunto com base em vetoruso, logo $\{z\}$.

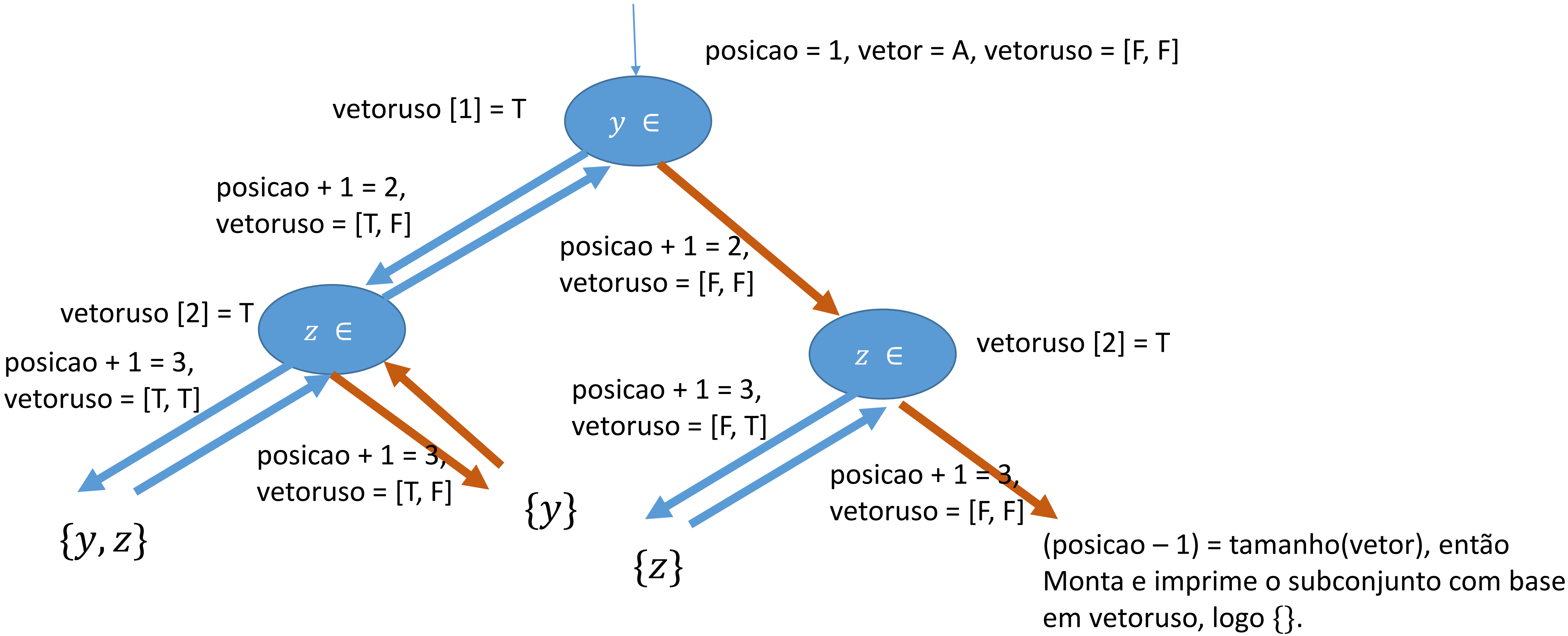
Conjunto Potência em R

Construindo a árvore para $A = \{y, z\}$ e gerando os subconjuntos $\mathcal{P}$:



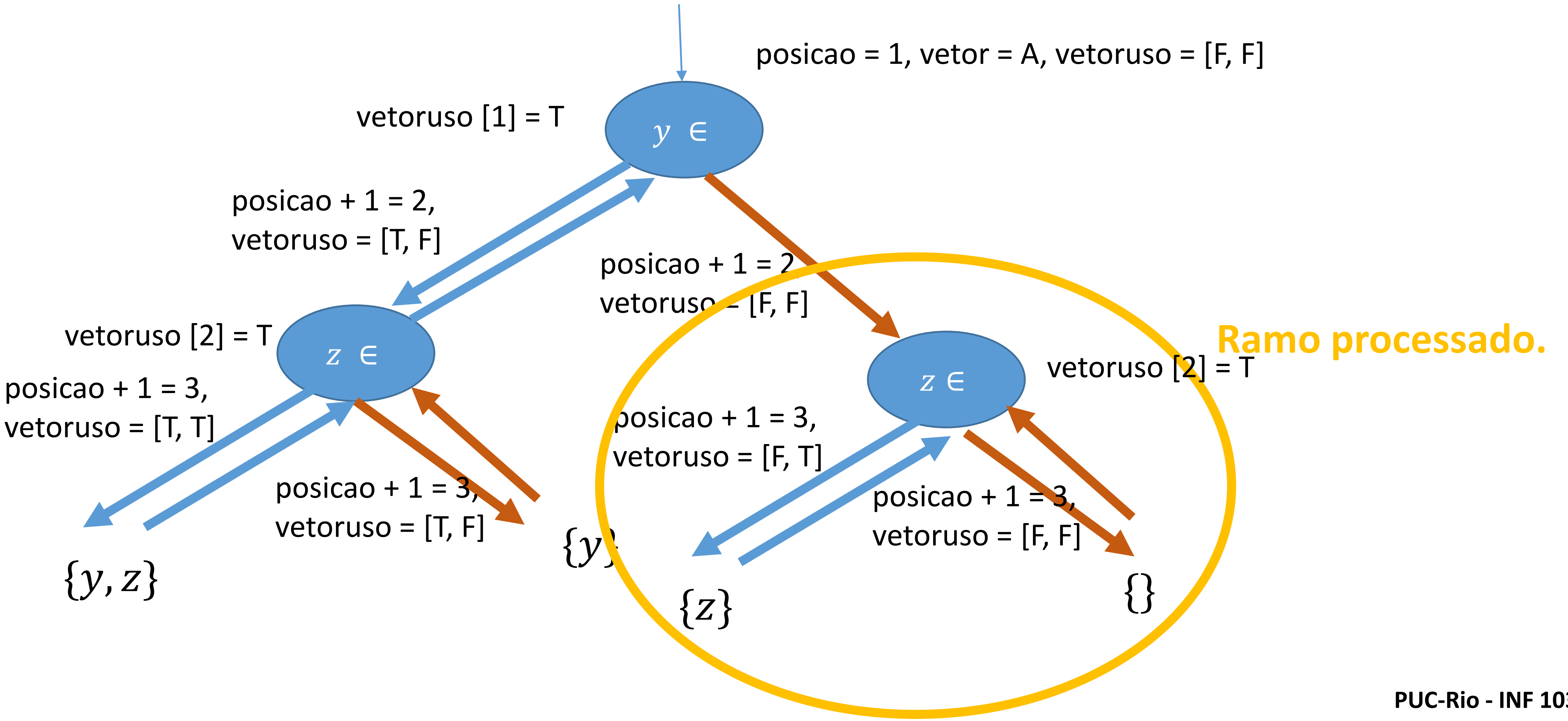
Conjunto Potência em R

Construindo a árvore para $A = \{y, z\}$ e gerando os subconjuntos $\mathcal{P}$:



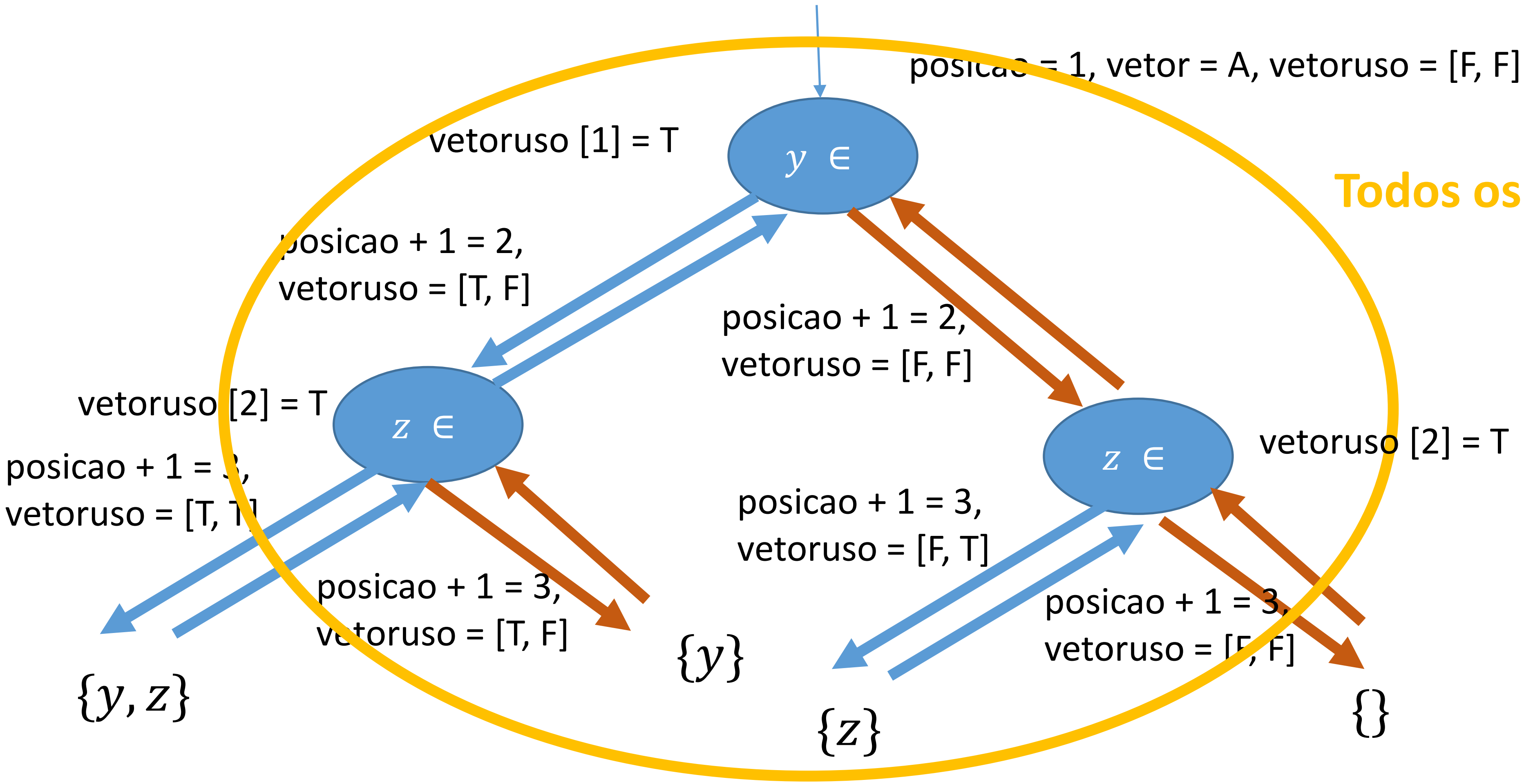
Conjunto Potência em R

Construindo a árvore para $A = \{y, z\}$ e gerando os subconjuntos $\mathcal{P}$:



Conjunto Potência em R

Construindo a árvore para $A = \{y, z\}$ e gerando os subconjuntos $\mathcal{P}$:



Todos os subconjuntos gerados.

Sigma Álgebra

Seja S um espaço amostral, e considere que $\mathcal{F}$ seja uma coleção de subconjuntos de S com as seguintes propriedades:

1. $S \in \mathcal{F}$.
2. Se $A \in \mathcal{F}$, então $\bar{A} \in \mathcal{F}$.
3. Se $A_i \in \mathcal{F}, i = 1,2, \ldots,$ então:

$$\bigcup_{i=1}^{\infty} A_i \in \mathcal{F}.$$

Uma coleção que satisfaz essas duas propriedades é chamada de uma **σ-álgebra**. Os elementos que constituem essa coleção são chamados de **eventos aleatórios**.

Sigma Álgebra

Exemplos:

1. Como o conjunto potência de S, $\mathcal{F} = \mathcal{P}(S)$, é o conjunto de todos os subconjuntos possíveis de S, então $\mathcal{P}(S)$ é uma σ-álgebra. Em particular, essa é a maior σ-álgebra de S.

2. Se S é um conjunto qualquer, então $\mathcal{F} = \{\emptyset, S\}$ é uma σ-álgebra de S. De fato, essa é a menor σ-álgebra de S.

3. Se A é um subconjunto qualquer de S, então$\mathcal{F} = \{\emptyset, A, \bar{A}, S\}$ é uma σ-álgebra de S.

Exercício: Suponha que $S = \{a, b, c\}$. Liste 4 diferentes σ-álgebras de S.

$$\mathcal{F} = \{\emptyset, \{a, b, c\}\}$$
$$\mathcal{F} = \{\emptyset, \{a\}, \{b, c\}, \{a, b, c\}\}$$
$$\mathcal{F} = \{\emptyset, \{b\}, \{a, c\}, \{a, b, c\}\}$$
$$\mathcal{F} = \mathcal{P}(S) = \{\emptyset, \{a\}, \{a, b\}, \{a, c\}, \{b\}, \{b, c\}, \{c\}, \{a, b, c\}\}$$

Sigma Álgebra

Propriedades:

- Suponha que A pertença a uma σ-álgebra $\mathcal{F}$ de S. Então, pela definição, $\bar{A} \in \mathcal{F}$ e $(A \cup \bar{A}) \in \mathcal{F}$. Portanto $S \in \mathcal{F}$.
- Também, pela definição de σ-álgebra, se $S \in \mathcal{F}$, então $\bar{S} = \emptyset \in \mathcal{F}$.
- Usando a Lei de De Morgan é simples provar a seguinte proposição:

Se $A_i \in \mathcal{F}, i = 1,2, \ldots,$ então

$$\bigcap_{i=1}^{\infty} A_i \in \mathcal{F}.$$

Álgebra de Borel

Um exemplo de uma σ-álgebra de muito interesse para nós é a **álgebra de Borel**. Ela será denotada por $\mathcal{B}$.

Aqui consideraremos a álgebra de Borel da reta real, mas existem também as álgebras de Borel do intervalo [0, 1], do plano $\mathbb{R}^2$, etc..

Álgebra de Borel

Para facilitar o entendimento da álgebra de Borel, vejamos o seguinte exemplo:

Suponha que $S = \{a, b, c, d\}$. É fácil checar que a coleção $\{\emptyset, \{a\}, \{a, b\}\}$ não é uma σ-álgebra de S.

Considere agora, que possamos adicionar a essa coleção os conjuntos que justamente a fazem se tornar uma σ-álgebra de S.

Teríamos, então, que adicionar os seguintes conjuntos:

1) $\overline{\{a\}} = \{b, c, d\}$, $\overline{\{a, b\}} = \{c, d\}$ => $\{\emptyset, \{a\}, \{a, b\}, \{c, d\}, \{b, c, d\}\}$

2) $\{a\} \cup \{c, d\} = \{a, c, d\}$, $\{a\} \cup \{b, c, d\} = \{a, b, c, d\}$ => $\{\emptyset, \{a\}, \{a, b\}, \{a, c, d\}, \{c, d\}, \{b, c, d\}, \{a, b, c, d\}\}$

3) $\overline{\{a, c, d\}} = \{b\}$

Com isso, a coleção

$\{\emptyset, \{a\}, \{a, b\}, \{a, c, d\}, \{b, c, d\}, \{c, d\}, \{b\}, \{a, b, c, d\}\}$ é uma σ-álgebra de S.

Álgebra de Borel

A σ-álgebra construída, conforme o exemplo anterior, a partir de uma dada coleção de subconjuntos C de S é chamada de σ**–álgebra gerada por** C. E ela corresponde à menor σ-álgebra de S que contém C.

Álgebra de Borel

Para construir a álgebra de Borel $\mathcal{B}$ da reta real faremos o seguinte algoritmo:

1. Inclua em $\mathcal{B}$ todos os intervalos do tipo $(-\infty, a]$, onde a é um número real qualquer.

2. Para $\mathcal{B}$ ser uma σ-álgebra, ela deve satisfazer à primeira condição. Isso implica que todos os intervalos na forma (a, ∞) devem estar em $\mathcal{B}$.

3. Suponha que a e b sejam dois números reais, com $a < b$.

Como $(-\infty, b] \in \mathcal{B}$ e $(a, \infty) \in \mathcal{B}$, então

$(-\infty, b] \cap (a, \infty) = (a, b]$ pertence a $\mathcal{B}$.

Função de Euler

Considere a função de Euler representada por $\varphi(x)$, definida para um número natural x como sendo igual à quantidade de números menores ou iguais a x co-primos em relação a ele:

$$\varphi(x) = \{y \in \mathbb{N} \mid y \leq x \wedge mdc(y, x) = 1\}$$

O valor de $\varphi(x)$ pode ser calculado a partir da decomposição de x em fatores primos.

Se a decomposição de x em fatores primos é

$$x = p_1^{j_1}, p_2^{j_2} \dots p_r^{j_r} \ (p_1, p_2, \dots, p_r \text{ primos distintos}),$$

Então:

$$\varphi(x) = x \left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right) \dots \left(1 - \frac{1}{p_r}\right)$$

Função de Euler

$$\varphi(x) = x\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\ldots\left(1 - \frac{1}{p_r}\right)$$

```
def funcao_Euler(n):
    retorno = 0
    fator = 2
    decomposicao = []
    produto = 1
    nfatorado = n
    while (nfatorado != 1):
        multiplicidade = 0
        while ((nfatorado % fator) == 0):
            nfatorado = nfatorado / fator
            multiplicidade = multiplicidade + 1
        if (multiplicidade != 0):
             decomposicao.append((fator, multiplicidade))
        fator = fator + 1
    for elemento in decomposicao:
         produto = produto * ( 1 - (1/elemento[0]))
    retorno = n * produto
    return (retorno)
```

Função de Euler

Exemplo 1) As decomposições de 120, 360 e 729 em fatores primos são:

$$120 = 2^3 \times 3 \times 5;$$

$$360 = 2^3 \times 3^2 \times 5 \text{ e}$$

$$729 = 3^6, \text{ temos}$$

$$\varphi(120) = 120\left(1-\frac{1}{2}\right)\left(1-\frac{1}{3}\right)\left(1-\frac{1}{5}\right) = 32;$$

$$\varphi(360) = 360\left(1-\frac{1}{2}\right)\left(1-\frac{1}{3}\right)\left(1-\frac{1}{5}\right) = 96 \text{ e}$$

$$\varphi(729) = 729\left(1-\frac{1}{3}\right) = 486$$

```
print(funcao_Euler(120))

print(funcao_Euler(360))

print(funcao_Euler(729))
```

Ou seja,

no conjunto $\{1, 2, \ldots, 120\}$ há 32 números co-primos em relação a 120,

no conjunto $\{1, 2, \ldots, 360\}$ há 96 números co-primos em relação a 360 e

no conjunto $\{1, 2, \ldots, 729\}$ há 486 números co-primos em relação a 729.

Função de Euler

Define-se:

S = Conjunto dos números positivos menores ou iguais a x;

A_i = Conjunto dos elementos de S que são múltiplos de p_i (1 ≤ i ≤ r)

Queremos determinar o número de elementos de S que são primos com x.

$\phi(x)$ é pois o número de elementos de S que pertencem a exatamente zero dos conjuntos A_1, A_2, ..., A_r.

Temos:

$$A_i = \left\{p_i, 2p_i, \ldots, \frac{x}{p_i}p_i\right\},\ n(A_i) = \frac{n}{p_i};$$

$$A_i \cap A_j = \left\{p_ip_j, 2p_ip_j, \ldots, \frac{x}{p_ip_j}p_ip_j\right\}, n(A_i \cap A_j) = \frac{x}{p_ip_j}\ (i \neq j);$$

e assim sucessivamente.

Logo:

$$S_0 = n(S) = x;$$

$$S_1 = \sum n(A_i) = \sum \frac{x}{p_i};$$

$$S_2 = \sum_{i<j} n(A_i \cap A_j) = \sum_{i<j} \frac{x}{p_ip_j};$$

$$\vdots$$

Função de Euler

Assim,

φ(x) = a_0

$$= \sum_{k=0}^{r} (-1)^k C_{0+k}^{k} S_{0+k};$$

$$= S_0 - S_1 + S_2 - \ldots + (-1)^r S_r$$

$$= x - \left(\frac{x}{p_1} + \frac{x}{p_2} + \ldots + \frac{x}{p_r}\right) +$$

$$\left(\frac{x}{p_1 p_2} + \frac{x}{p_1 p_3} + \ldots + \frac{x}{p_{r-1} p_r}\right) - \ldots$$

$$+(-1)^r \frac{x}{p_1 p_2 \ldots p_r}$$

$$= x\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\ldots\left(1 - \frac{1}{p_r}\right).$$

Função de Euler

$$\varphi(x) = x\left(1 - \frac{1}{p_1}\right)\left(1 - \frac{1}{p_2}\right)\ldots\left(1 - \frac{1}{p_r}\right)$$

```
funcao_Euler <- function (n) {
  retorno <- 0
  fator <- 2
  decomposicao <- NULL
  produto <- 1
  nfatorado <- n
  while (nfatorado != 1){
    multiplicidade = 0
    while ((nfatorado %% fator) == 0) {
      nfatorado = nfatorado / fator
      multiplicidade = multiplicidade + 1
    }
    if (multiplicidade != 0)
      decomposicao <- rbind (decomposicao, c(fator, multiplicidade))
    fator = fator + 1
  }
  for (i in seq(1:nrow(decomposicao)))
    produto = produto * ( 1 - (1/decomposicao[i, 1]))
  retorno = n * produto
  return (retorno)
}
```

Referências Bibliográficas

1. Augusto C. Morgado, et al Análise Combinatória e Probabilidade. Rio de Janeiro: SBM, 2006.

2. Jane M. Horgan, Probability with R, Willey, 2009.

3. Hwei Hsu, Probability, Random Variables, and Random Processes, Schaum's outlines, 1996.

4. Ramakant Khazanic, Basic Probability Theory and Applications, Goodyear Pub., 1976.