Matemática Discreta para Cursos de Computação

Jackson Gomes \ jgomes@ceulp.edu.br

Sumário

Lista de Tabelas

Lista de Figuras

Lista de Códigos-fontes

Prefácio

Este é um texto de apoio à disciplina Matemática Discreta para os cursos de computação do Centro Universitário Luterano de Palmas. Sempre que possível serão apresentadas referências a conceitos da computação e como eles se relacionam com os conceitos matemáticos apresentados. A principal referência do conteúdo utilizado aqui é (MENEZES, 2010).

Convenções

Os trechos de código apresentados no livro seguem o seguinte padrão:

- **comandos**: devem ser executados no prompt; começam com o símbolo `$`
- **códigos-fontes**: trechos de códigos-fontes de arquivos

A seguir, um exemplo de comando:

```
$ mkdir hello-world
```

O exemplo indica que o comando `mkdir`, com a opção `hello-world`, deve ser executado no prompt para criar uma pasta com o nome `hello-world`.

A seguir, um exemplo de código-fonte:

```
class Pessoa:
    pass
```

O exemplo apresenta o código-fonte da classe `Pessoa`. Em algumas situações, trechos de código podem ser omitidos ou serem apresentados de forma incompleta, usando os símbolos . . . e `#`, como no exemplo a seguir:

```
class Pessoa:
    def __init__(self, nome):
        self.nome = nome

    def salvar(self):
        # executa validação dos dados
```

```
        ...
        # salva
        return ModelManager.save(self)
```

Conhecimentos necessários e desejáveis

Este texto aborda conceitos matemáticos com aplicações em computação. Portanto, conhecimentos básicos dos cursos de computação são necessários, como noções de lógica, algoritmos e programação e estruturas de dados. Além disso, são desejáveis conhecimentos de bancos de dados e orientação a objetos e também podem ser recursos úteis:

- Git (GIT COMMUNITY, [s.d.])
- Visual Studio Code (MICROSOFT, [s.d.])
- TypeScript (MICROSOFT, [s.d.])
- Node.js (NODE.JS FOUNDATION, [s.d.])
- npm (NPM, INC., [s.d.])

Capítulo 1

Introdução

1.1 Matemática Discreta

Conforme (MENEZES, 2010) as Diretrizes Curriculares do MEC para os cursos de computação e informática definem que:

> A matemática, para a área de computação, deve ser vista como uma ferramenta a ser usada na definição formal de conceitos computacionais (linguagens, autômatos, métodos etc.). Os modelos formais permitem definir suas propriedades e dimensionar suas instâncias, dadas suas condições de contorno.

Além disso, afirmam:

> Considerando que a maioria dos conceitos computacionais pertencem ao domínio discreto, a **matemática discreta** (ou também chamada álgebra abstrata) é fortemente empregada.

Desta forma, a **Matemática Discreta** preocupa-se com o emprego de técnicas e abordagens da matemática para o entendimento de problemas a serem resolvidos com computação. Mas o que significa ser **discreto**? A matemática, por si, trata também do domínio **contínuo**. Assim, estes domínios são opostos: contínuo e discreto. Para entender isso melhor, observe a Figura 1.1:

Figura 1.1 representa a função $y = x^2$, com $0 \leq x \leq 5$ em dois gráficos, sendo que o da direita destaca pontos selecionados, que representam 6 amostras (0, 1, 2, 3, 4 e 5).

Aumentando-se o número de amostras em dois instantes, para 10, 100 e 1000 teríamos, como mostra a Figura 1.2:

O que se pode perceber pela Figura 1.2 é que quanto mais se aumenta o número de amostras, mais se aproxima de uma curva perfeita. Entretanto, há um certo limite de percepção da perfeição dessa curva, por assim dizer. Por exemplo, embora a quantidade de amostras do gráfico da esquerda seja menor, a diferença para o gráfico da direita, visualmente falando, é pouco perceptível.

Considere outro exemplo: um computador possui uma capacidade de armazenamento virtualmente infinita. “Virtualmente” porque embora se aceite um limite, ele não é conhecido, já que a quan-

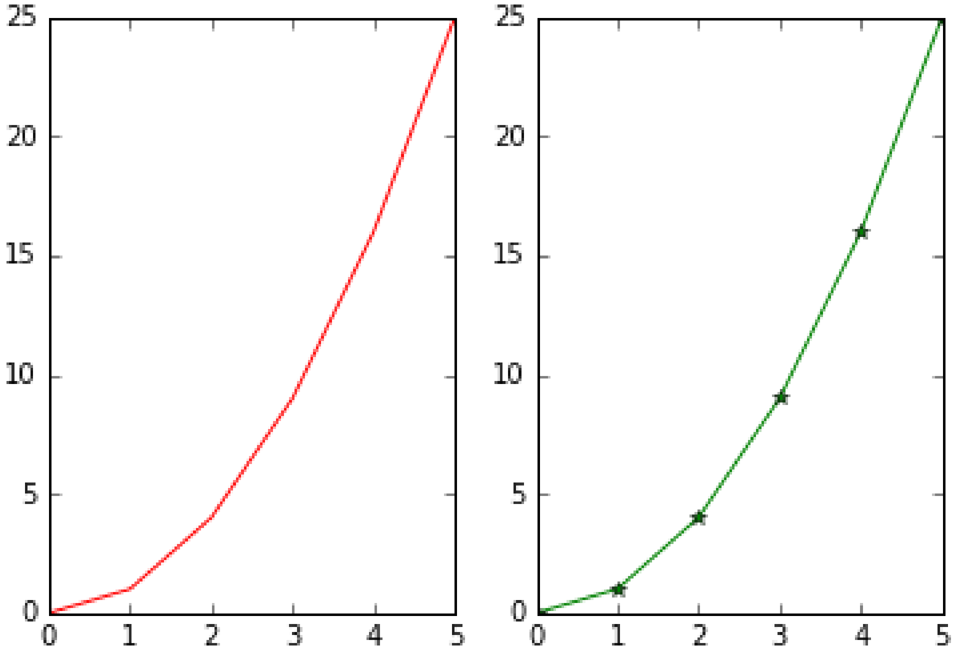

Figura 1.1: Gráfico da $y = x^2$, com $0 \leq x \leq 5$

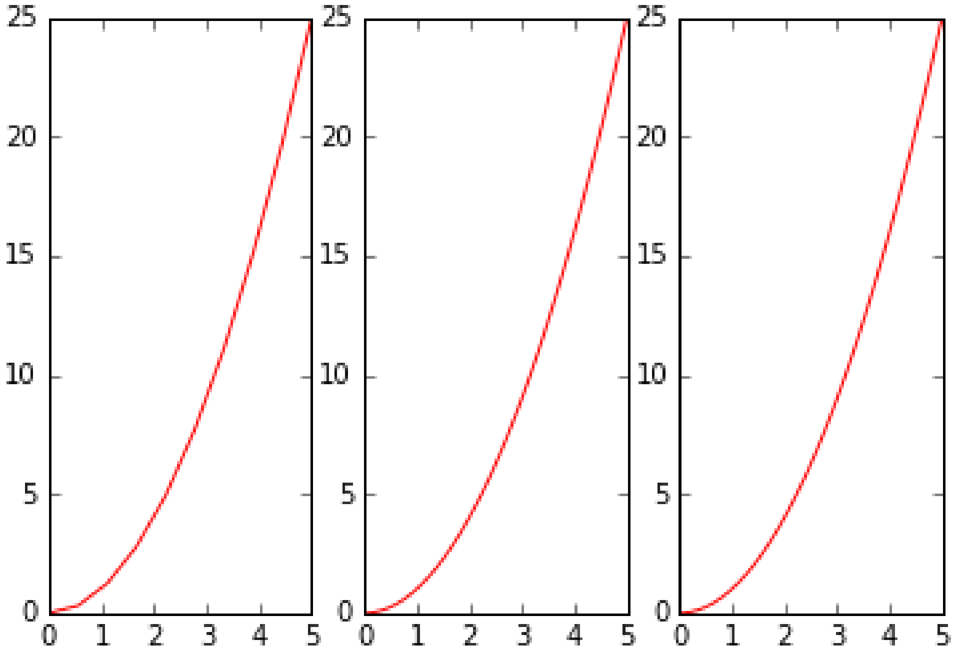

Figura 1.2: Gráfico da $y = x^2$ com mais amostras

tidade de unidades de armazemamento pode ser bastante grande, mas é **contável**. Assim, no contexto da computação, embora algo possa ser considerado finito ou infinito, ele é *contável* ou *discreto* no sentido de que pode ser enumerado ou sequenciado, de forma que não existe um elemento entre quaisquer dois elementos consecutivos da enumeração.

No exemplo do computador, embora a quantidade de unidades de armazenamento não seja conhecida, ela é contável e enumerável e não se pode afirmar que exista, por um exemplo, um disco rídigo desconhecido entre os array de discos composto por D1 e D2. Outro exemplo: na matemática, o conjunto dos números naturais é contável (ou enumerável), equanto o conjunto dos números reais não é contável.

Assim, a matemática discreta possui como ênfase os estudos matemáticos baseados em conjuntos contáveis, sejam eles finitos ou infinitos. De forma oposta, a *matemática do continuum* possui ênfase nos conjuntos não contáveis. Um exemplo disso são o cálculo diferencial e integral.

1.2 Teoria dos Conjuntos

Os **conjuntos** são a base da forma de representação de enumerações de elementos em matemática discreta. Por definição um conjunto é:

> uma estrutura que agrupa objetos e constitui uma base para construir estruturas mais complexas.

Segue uma definição mais formal:

> Um *conjunto* é uma coleção de zero ou mais objetos distintos, chamados *elementos* do conjunto, os quais não possuem qualquer ordem associada.

O fato de não haver uma *ordem associada* não significa que os elementos não possam estar ordenados, num dado contexto, conforme algum critério. Apenas indica que, no geral, isso não é obrigatório.

Há duas formas (notações) de representar conjuntos: notação por extensão e notação por compreensão.

Notação por extensão é quando todos os elementos do conjunto estão enumerados, representados entre chaves e separados por vírgula. Exemplo:

Vogais $= \{a, e, i, o, u\}$.

Entende-se que se um conjunto pode ser representado por extensão, então ele é *finito*. Caso contrário, é *infinito*.

Notação por compreensão representa conjuntos usando propriedades. Os exemplos a seguir usam uma pequena diferença de notação, mas representam a mesma coisa:

- Pares $= \{n \mid n$ é um número par$\}$
- Pares $= \{n : n$ é um número par$\}$

Este conjunto é interpretado como: o conjunto de todos os elementos n tal que n é um número par. A forma geral de representar um conjunto por propriedades é:

$X = \{x : p(x)\}$

Isso quer dizer que x é um elemento de X se a propriedade $p(x)$ for verdadeira.

A notação por propriedades é uma boa forma de representar conjuntos *infinitos*.

Há ainda uma outra forma aceitável de representar conjuntos usando uma representação semelhante à de por extensão. Exemplos:

- Digitos $= \{0, 1, 2, ..., 9\}$
- Pares $= \{0, 2, 4, 6, ...\}$

Embora haja elementos ausentes, substituídos por reticências (...) é completamente aceitável e entendível o que se quer informar com a descrição do conjunto.

O número de elementos de um conjunto A é representado por $|A|$ (isso também é chamado "cardinalidade"). Portanto, se $A = \{1, 2, 3\}$, $|A| = 3$ e $|\emptyset| = 0$.

A seguir, revemos conceitos de algumas relações entre e com conjuntos ou elementos.

1.2.1 Pertinência

Se um elemento a pertence ao conjunto A isso é representado como: $a \in A$. Caso contrário, se a não pertence a A, então representa-se como: $a \notin A$.

Exemplos: Pertence, não pertence

Quanto ao conjunto Vogais $= \{a, e, i, o, u\}$:
a) $a \in$ Vogais
b) $h \notin$ Vogais

Quanto ao conjunto $B = \{x : x$ é brasileiro$\}$:
a) Pele $\in B$
b) Bill Gates $\notin B$

1.2.2 Conjuntos importantes

O **conjunto vazio** é um conjunto sem elementos, representado como $\{\}$ ou $\emptyset$. Exemplos:

- o conjunto de todos os brasileiros com mais de 300 anos;
- o conjunto dos números que são, simultaneamente, ímpares e pares.

O **conjunto unitário** é um conjunto constituído por um único elemento. Exemplos:

- o conjunto constituído pelo jogador de futebol Pelé;
- o conjunto de todos os números que são, simultaneamente, pares e primos, ou seja: $P = \{2\}$;
- um conjunto unitário cujo elemento é irrelevante: $1 = \{*\}$.

O **conjunto universo**, normalmente denotado por U, contém todos os conjuntos considerados em um dado contexto.

Outros conjuntos importantes:

- N: o conjunto dos números naturais (inteiros positivos e o zero)
- Z: o conjunto dos números inteiros (inteiros negativos, positivos e o zero)
- Q: o conjunto dos números racionais (os que podem ser representados na forma de fração)
- I: o conjunto dos números irracionais
- R: o conjunto dos números reais

1.2.3 Alfabetos, palavras e linguagens

Em computação, e mais especificamente em linguagens de programação, um conceito importante é o que define o conjunto de elementos ou termos-chave da linguagem.

Um **alfabeto** um conjunto finito cujos elementos são denominados *símbolos* ou *caracteres*.

Uma **palavra** (cadeia de caracteres ou sentença) sobre um alfabeto é uma sequência finita de símbolos justapostos.

Uma **linguagem** [**formal**] é um conjunto de palavras sobre um alfabeto.

Exemplos: alfabeto, palavra

a) Os conjuntos $\emptyset$ e $\{a, b, c\}$ são alfabetos
b) O conjunto N não é um alfabeto
c) ϵ é uma palavra vazia
d) Σ é geralmente usada para representar um alfabeto
e) Σ^* é o conjunto de todas as palavras possíveis sobre o alfabeto Σ
f) ϵ é uma palavra do alfabeto $\emptyset$
g) $\{a, b\}^* = \{\epsilon, a, b, aa, ab, ba, bb, aaa, ...\}$

1.2.4 Continência, subconjunto e igualdade de conjuntos

A *continência* permite introduzir os conceitos de *subconjunto* e *igualdade de conjunto*.

Se todos os elementos de um conjunto A também são elementos de um conjunto B, então A está *contido* em B, o que é representado por: $A \subseteq B$. Isso também é lido como A é *subconjunto* de B.

Se $A \subseteq B$, mas há $b \in B$ tal que $b \notin A$, então pode-se dizer que A está *contido propriamente* em B, ou que A é *subconjunto próprio* de B. Isso é denotado por: $A \subset B$.

A negação de *subconjunto* e *subconjunto próprio* é, respectivamente:

- $A \nsubseteq B$ e
- $A \not\subset B$

Exemplos: continência, subconjunto
a) $\{a, b\} \subseteq \{b, a\}$ b) $\{a, b\} \subset \{a, b, c\}$, e $\{a, b\} \subseteq \{a, b, c\}$

A *igualdade de conjuntos* é um conceito baseado em pertinência: se os elementos de A também são elementos de B e vice-versa, então $A = B$. Formalmente, uma condição para $A = B$ é que $A \subseteq B$ e $B \subseteq A$.

Exemplo
$\{1, 2, 3\} = \{3, 3, 3, 2, 2, 1\}$

É importante notar que pertinência ($\in$) é usada entre elementos e conjuntos, enquanto continência ($\subset$ e $\subseteq$) é usada entre conjuntos.

Por definição, um conjunto qualquer é subconjunto de si mesmo, e $\emptyset$ é subconjunto de qualquer conjunto.

Exemplo
Seja $A = \{1, 2\}$ então os subconjuntos de A são: $\emptyset$, $\{1\}$, $\{2\}$ e $\{1, 2\}$.

1.3 Conjuntos, Tuplas e Listas

Uma **Tupla** (ou ênupla) é uma sequência ordenada de n elementos (ou componentes). As principais diferenças para **conjunto** são:

- uma ênupla pode conter um elemento mais de uma vez; e
- os elementos são, obrigatoriamente, ordenados, ou seja, cada elemento está em uma posição diferente.

A notação utilizada é $X = (c_1, c_2, ..., c_n)$ onde:

- X é uma ênupla com n componentes
- c_1 é o primeiro componente da ênupla
- c_n é o último componente da ênupla
- c_i é o i-ésimo componente da ênupla

Exemplos:

- $X = (1, 1, 2, 3)$
- $X = (1_1, 1_2, 2_3, 3_4)$
- $Y = (3_1, 2_2, 1_3)$

As mesmas relações de pertinência entre conjuntos e elementos podem ser aplicadas entre ênuplas e seus componentes.

Uma **Lista** é uma estrutura de dados que implementa o conceito matemático de **Tupla**, então podemos afirmar para Vogais $= (a, e, i, o, u)$:

- Vogais é uma lista com 5 elementos
- a é a primeira vogal
- u é a última vogal

1.4 Exercícios

Questão 1. Para cada conjunto abaixo: a) descreva de forma alternativa (usando outra forma de notação); e b) diga se é finito ou infinito.

a) todos os números inteiros maiores que 10
b) $\{1, 3, 5, 7, 9, 11, ...\}$
c) todos os países do mundo (Terra)
d) a linguagem de programação Python

Questão 2. Para $A = \{1\}$, $B = \{1, 2\}$ e $C = \{\{1\}, 1\}$, marque as afirmações corretas:

a) $A \subset B$
b) $A \subseteq B$
c) $A \in B$
d) $A = B$
e) $A \subset C$
f) $A \subseteq C$
g) $A \in C$
h) $A = C$
i) $1 \in A$
j) $1 \in C$
k) $\{1\} \in A$
l) $\{1\} \in C$
m) $\emptyset \notin C$
n) $\emptyset \subset C$

Questão 3. Sejam $a = \{x \mid 2x = 6\}$ e $b = 3$. É correto afirmar que $a = b$? Por que?

Questão 4. Quais todos os subconjuntos dos seguintes conjuntos?

a) $A = \{a, b, c\}$
b) $B = \{a, \{b, c\}, D\}$, dado que $D = \{1, 2\}$

Questão 5. O conjunto vazio está contido em qualquer conjunto, inclusive nele próprio? Justifique.

Questão 6. Todo conjunto possui um subconjunto próprio? Justifique.

Questão 7. Sejam $A = \{0, 1, 2, 3, 4, 5\}$, $B = \{3, 4, 5, 6, 7, 8\}$, $C = \{1, 3, 7, 8\}$, $D = \{3, 4\}$, $E = \{1, 3\}$, $F = \{1\}$ e X um conjunto desconhecido. Para cada item abaixo, determine quais dos conjuntos A, B, C, D, E ou F podem ser iguais a X:

a) $X \subseteq A$ e $X \subseteq B$
b) $X \not\subseteq B$ e $X \subseteq C$
c) $X \not\subseteq A$ e $X \not\subseteq C$
d) $X \subseteq B$ e $X \not\subseteq C$

Questão 8. Sejam A um subconjunto de B e B um subconjunto de C. Suponha que $a \in A$, $b \in B$, $c \in C$, $d \notin A$, $e \notin B$, $f \notin C$. Quais das seguintes afirmações são verdadeiras?

a) $a \in C$
b) $b \in A$
c) $c \notin A$
d) $d \in B$
e) $e \notin A$
f) $f \notin A$

Questão 9. Bancos de dados relacionais costumam representar conjuntos de dados organizados em tabelas, colunas e registros. É possível considerar alguma semelhança entre o conceito de tabela e o conceito de conjunto? Há recursos de consulta (em linguagem SQL, por exemplo) que permitam identificar, ainda que parcialmente, relações entre tabelas e registros (ou valores) como as expressadas nesse capítulo entre elementos e conjuntos? Explique.

Questão 10. Escolha uma linguagem de programação e demonstre seus recursos para lidar com conjuntos e listas. Utilizando código-fonte (e a sintaxe da linguagem de programação) demonstre e explique as diferenças conceituais e demonstre recursos da linguagem que permitam identificar as relações de pertinência, continência, subconjunto e igualdade.

Capítulo 2

Álgebra de Conjuntos

Uma **Álgebra** é constituída de operações definidas sobre uma soleção de objetos. Neste contexto, *álgebra de conjuntos* corresponderia às operações definidas sobre todos os conjuntos.

As operações da álgebra de conjuntos podem ser:

- Não reversíveis
  - União
  - Intersecção
  - Diferença
- Reversíveis
  - Complemento
  - Conjunto das partes
  - Produto cartesiano
  - União disjunta

2.1 Operações não reversíveis

2.1.1 União

Sejam A e B dois conjuntos. A união entre eles, $A \cup B$, é definida como:

$$A \cup B = \{x \mid x \in A \vee x \in B\} \tag{2.1}$$

Considerando a lógica, o conjunto A pode ser definido como $x \in A$ e o conjunto B pode ser definido como $x \in B$. Ou seja, a propriedade de pertiência é utilizada para indicar uma proposição lógica.

A união corresponde à operação lógica *disjunção* (símbolo $\vee$).

Exemplo: união entre conjuntos

Considere os conjuntos:

Exemplo: união entre conjuntos

a) Digitos = $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
b) Vogais = $\{a, e, i, o, u\}$
c) Pares = $\{0, 2, 4, 6, ...\}$

Então:
a) Digitos $\cup$ Vogais = $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, e, i, o, u\}$
b) Digitos $\cup$ Pares = $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16, ...\}$

Suponha os conjuntos:
a) $A = \{x \in N \mid x > 2\}$
b) $B = \{x \in N \mid x^2 = x\}$

Então:
$A \cup B = \{0, 1, 3, 4, 5, 6, ...\}$

É importante observar que o resultado da união é um conjunto sem repetições de elementos.

Vejamos as propriedades da união:

- **Elemento neutro**: $A \cup \emptyset = \emptyset \cup A = A$
- **Idempotência**: $A \cup A = A$
- **Comutativa**: $A \cup B = B \cup A$
- **Associativa**: $A \cup (B \cup C) = (A \cup B) \cup C$

2.1.2 Intersecção

Sejam dois conjuntos A e B. A intersercção entre eles, $A \cap B$ é definida como:

$$A \cap B = \{x \mid x \in A \wedge x \in B\} \tag{2.2}$$

A união corresponde à operação lógica *conjunção* (símbolo $\wedge$).

Exemplo: intersecção

Considere os conjuntos:
a) Digitos = $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$
b) Vogais = $\{a, e, i, o, u\}$
c) Pares = $\{0, 2, 4, 6, ...\}$

Então:
a) Digitos $\cap$ Vogais = $\emptyset$
b) Digitos $\cap$ Pares = $\{0, 2, 4, 6, 8\}$

Também podemos utilizar **Diagrama de Venn** para demonstrar a Intersecção, como ilustra a Figura 2.1.

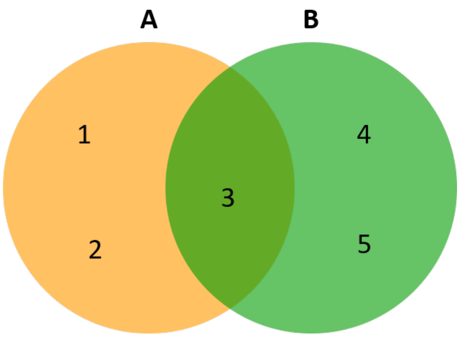

Figura 2.1: Diagrama de Venn demonstrando a intersecção entre conjuntos A e B

A Figura 2.1 considera os conjuntos $A = \{1, 2, 3\}$ e $B = \{3, 4, 5\}$ e mostra o resultado de $A \cap B$. A área mais ao centro, colorida como uma mistura das cores dos conjuntos, corresponde à intersecção (não é necesssário utilizar cores, entretanto).

Sejam A e B dois conjuntos não vazios. Se $A \cap B = \emptyset$, então A e B são chamados *conjuntos disjuntos*, *conjuntos independentes*, ou *conjuntos mutuamente exclusivos*.

Propriedades da intersecção:

- **Elemento neutro**: $A \cap U = U \cap A = A$
- **Idempotência**: $A \cap A = A$
- **Comutativa**: $A \cap B = B \cap A$
- **Associativa**: $A \cap (B \cap C) = (A \cap B) \cap C$

2.2 Propriedades envolvendo união e intersecção

As propriedades a seguir envolvem as operações de união e intersecção:

- **Distributividade da intersecção sobre a união**: $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
- **Distributividade da união sobre a intersecção**: $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
- **Absorção**: $A \cap (A \cup B) = A$ e $A \cup (A \cap B) = A$.

2.3 Operações reversíveis

Entende-se por **operação reversível** uma operação a partir de cujo resultado pode-se recuperar os operandos originais.

2.3.1 Complemento

Considere o conjunto universo U. O complemento de um conjunto $A \subseteq U$, denotado por $\sim A$ é definido como:

$$\sim A = \{x \in U \mid x \notin A\} \tag{2.3}$$

Exemplos: complemento

1. Considere o conjunto universo definido por Digitos $= \{0, 1, 2, ..., 9\}$. Seja $A = \{0, 1, 2\}$. Então, $\sim A = \{3, 4, 5, 6, 7, 8, 9\}$.

2. Suponha conjunto universo $\mathbb{N}$. Seja $A = \{0, 1, 2\}$. Então $\sim A = \{x \in \mathbb{N} \mid x > 2\}$.

3. Para qualquer conjunto universo U, valem:
a) $\sim \emptyset = U$
b) $\sim U = \emptyset$

4. Suponha que U é o conjunto universo. Então, para qualquer conjunto $A \subseteq U$, valem:
a) $A \cup \sim A = U$
b) $A \cap \sim A = \emptyset$

No último exemplo, observe que:

- a união de um conjunto com seu complemento sempre resulta no conjunto universo ($p \vee \sim p =$ Verdadeiro); e
- a intersecção de um conjunto com seu complemento sempre resulta no conjunto vazio ($p \wedge \sim p =$ Falso).

Também vale a noção de *duplo complemento* (ou *dupla negação*):

$$\sim\sim A = A \tag{2.4}$$

A propriedade denominada **DeMorgan** vale-se do complemento, envolvendo as operações de união e interseção (Tabela 2.4).

Tabela 2.4: DeMorgan na álgebra de conjuntos e na Lógica

DeMorgan na Álgebra de conjuntos	Propriedade DeMorgan na Lógica
$\sim (A \cup B) = \sim A \cap \sim B$	$\neg(p \vee q) \Leftrightarrow \neg p \wedge \neg q$
$\sim (A \cap B) = \sim A \cup \sim B$	$\neg(p \wedge q) \Leftrightarrow \neg p \vee \neg q$

2.3.2 Diferença

Sejam os conjuntos A e B. A diferença dos conjuntos A e B, denotada por $A - B$ é definida como:

$$A - B = A \cap \sim B \tag{2.5}$$

ou

$$A - B = \{x \mid x \in A \land x \notin B\} \tag{2.6}$$

Exemplos: diferença

---

1. Suponha os conjuntos: Digitos $= \{0, 1, 2, ..., 9\}$, Vogais $= \{a, e, i, o, u\}$ e
Pares $= \{0, 2, 4, 6, ...\}$. Utilizando a diferença, temos:
a) Digitos $-$ Vogais $=$ Digitos
b) Digitos $-$ Pares $= \{1, 3, 5, 7, 9\}$

2. Para qualquer conjunto universo U e qualquer $A \subseteq U$, valem:
a) $\emptyset - \emptyset = \emptyset$
b) $U - \emptyset = U$
c) $U - A = \sim A$
d) $U - U = \emptyset$

2.3.3 Conjunto das partes

Para qualquer conjunto A sabe-se que:

- $A \subseteq A$
- $\emptyset \subseteq A$
- Para qualquer elemento $a \in A$, é visível que $\{a\} \subseteq A$

A operação unária chamada *conjunto das partes*, ao ser aplicada ao conjunto A, resulta no conjunto de todos os subconjuntos de A. Suponha um conjunto A. O conjunto das partes de A (ou conjunto potência), denotado por $\mathrm{P}(A)$ ou 2^A, é definido por:

$$\mathrm{P}(A) = \{X \mid X \subseteq A\} \tag{2.7}$$

Exemplos: conjunto das partes

Sejam os conjuntos $A = \{a\}$, $B = \{a, b\}$, $C = \{a, b, c\}$ e $D = \{a, \emptyset, \{a, b\}\}$, então:
1. $\mathrm{P}(\emptyset) = \{\emptyset\}$
2. $\mathrm{P}(A) = \{\emptyset, \{a\}\}$
3. $\mathrm{P}(B) = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$
4. $\mathrm{P}(C) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a, b, c\}\}$
5. $\mathrm{P}(D) = \{\emptyset, \{a\}, \{\emptyset\}, \{\{a, b\}\}, \{a, \emptyset\}, \{a, \{a, b\}\}, \{\emptyset, \{a, b\}\}, \{a, \emptyset, \{a, b\}\}\}$

2.3.4 Produto Cartesiano

A operação **produto cartesiano** é uma operação binária que, quando aplicada a dois conjuntos A e B, resulta em um conjunto constituído de sequências de duas componentes (tuplas), sendo que a primeira componente de cada sequência é um elemento de A, e a segunda componente, um elemento de B.

Uma sequência de n componentes, denominada *n-upla ordenada* (lê-se: ênupla ordenada), consiste de n objetos (não necessariamente distintos) em uma ordem fixa. Por exemplo, uma 2-upla (tupla) ordenada é denominada *par ordenado*. Um par ordenado no qual a primeira componente é x e a segunda é y é definido como $\langle x, y \rangle$ ou (x, y).

Uma n-upla ordenada é definida como:

$$\langle x_1, x_2, x_3, ..., x_n \rangle \tag{2.8}$$

Uma n-upla ordenada não deve ser confundida com um conjunto, pois a ordem das componentes é importante. Assim:

$$\langle x, y \rangle \neq \langle y, x \rangle \tag{2.9}$$

O **produto cartesiano** dos conjuntos A e B, denotado por $A \times B$ é definido por:

$$A \times B = \{\langle a, b \rangle \mid a \in A \wedge b \in B\} \tag{2.10}$$

O produto cartesiano de um conjunto com ele mesmo é definido por $A \times A = A^2$

Exemplos: produto cartesiano

Sejam os conjuntos $A = \{a\}$, $B = \{a, b\}$ e $C = \{0, 1, 2\}$. Então:

1. $A \times B = \{\langle a, a \rangle, \langle a, b \rangle\}$
2. $B \times C = \{\langle a, 0 \rangle, \langle a, 1 \rangle, \langle a, 2 \rangle, \langle b, 0 \rangle, \langle b, 1 \rangle, \langle b, 2 \rangle\}$
3. $A^2 = \{\langle a, a \rangle\}$

2.3.5 União disjunta

Diferentemente da *união*, que desconsidera repetições de elementos no conjunto resultante, a *união disjunta* permite que os elementos do conjunto resultante sejam duplicados, uma vez que seja identificada a sua fonte. A *união disjunta* dos conjuntos A e B, denotada por $A + B$ ou $A \dot{\cup} B$ é definida como:

$$A + B = \{\langle a, A \rangle \mid a \in A\} \cup \{\langle b, B \rangle \mid b \in B\} \tag{2.11}$$

ou

$$A + B = \{a_A \mid a \in A\} \cup \{b_B \mid b \in B\} \tag{2.12}$$

Exemplo: união disjunta

Suponha os conjuntos $\text{Silva} = \{\text{João}, \text{Maria}, \text{José}\}$ e $\text{Souza} = \{\text{Pedro}, \text{Ana}, \text{José}\}$. Então:

1. $\text{Silva} + \text{Souza} =$
$\{\langle \text{João}, \text{Silva}\rangle, \langle \text{Maria}, \text{Silva}\rangle, \langle \text{José}, \text{Silva}\rangle, \langle \text{Pedro}, \text{Souza}\rangle, \langle \text{Ana}, \text{Souza}\rangle, \langle \text{José}, \text{Souza}\rangle\}$

ou

2. $\text{Silva} + \text{Souza} = \{\text{João}_{\text{Silva}}, \text{Maria}_{\text{Silva}}, \text{José}_{\text{Silva}}, \text{Pedro}_{\text{Souza}}, \text{Ana}_{\text{Souza}}, \text{José}_{\text{Souza}}\}$

2.4 Provando propriedades

2.4.0.1 Prova da propriedade *elemento neutro da união*

Elemento neutro é definido como:

$$A \cup \emptyset = \emptyset \cup A = A \tag{2.13}$$

Assim, há duas igualdades, que podem ser analisadas considerando a validade da transitividade [da igualdade]. Assim, temos que observar alguns casos:

(A) Para provar $A \cup \emptyset = \emptyset \cup A$:

O primeiro caso (1): Seja $x \in A \cup \emptyset$. Então devemos provar que $A \cup \emptyset \subseteq \emptyset \cup A$:

- $x \in A \cup \emptyset \implies$ (definição de união)
- $x \in A \vee x \in \emptyset \implies$ (comutatividade da disjunção)
- $x \in \emptyset \vee x \in A \implies$ (definição de união)
- $x \in \emptyset \cup A$

Portanto, $A \cup \emptyset \subseteq \emptyset \cup A$.

O segundo caso (2): Seja $x \in \emptyset \cup A$. Então devemos provar que $\emptyset \cup A \subseteq A \cup \emptyset$:

- $x \in \emptyset \cup A \implies$ (definição de união)
- $x \in \emptyset \vee x \in A \implies$ (comutatividade da disjunção)
- $x \in A \vee x \in \emptyset \implies$ (definição de união)
- $x \in A \cup \emptyset$

Portanto, $\emptyset \cup A = A \cup \emptyset$.

Terceiro caso (3): De (1) e (2) concluímos que $A \cup \emptyset = \emptyset \cup A$.

(B) Para provar $A \cup \emptyset = A$:

Quarto caso (4): Seja $x \in A \cup \emptyset$. Então devemos provar que $A \cup \emptyset \subseteq A$:

- $x \in A \cup \emptyset \implies$ (definição de união)
- $x \in A \vee x \in \emptyset \implies$ ($x \in \emptyset$ é sempre *false*)
- $x \in A$

Portanto, $A \cup \emptyset \subseteq A$.

Quinto caso (5): Seja $x \in A$. Então devemos provar que $A \subseteq A \cup \emptyset$:

- $x \in A \implies$ ($x \in A$ é sempre *true*, portanto podemos considerar $p \implies p \vee q$)
- $x \in A \vee x \in \emptyset$ (definição de união)
- $x \in A \cup \emptyset$

Portanto, $A \subseteq A \cup \emptyset$.

Sexto caso (6): De (4) e (5) concluímos que $A \cup \emptyset = A$.

Sétimo caso (7): Por fim, de (3) e (6) e pela transitividade da igualdade, concluímos que $A \cup \emptyset = \emptyset \cup A = A$ e provamos a propriedade do *elemento neutro* da união.

Exercício: Prove as propriedades idempotência, comutativa e associativa da união.

2.5 Relações entre a Lógica e as operações sobre conjuntos

É possível estabelecer uma relação entre a lógica e as operações da álgebra de conjuntos, como mostra a Tabela 2.9.

Tabela 2.9: Relação entre conectivos da lógica e operações sobre conjuntos

Conectivo ou relação lógicos	Operação ou relação sobre conjuntos
Negação	Complemento
Disjunção	União
Conjunção	Intersecção
Implicação	Continência
Equivalência	Igualdade

As propriedades dos conectivos e operadores lógicos são válidas na teoria dos conjuntos (Tabela 2.10):

Tabela 2.10: Relação entre propriedades dos conectivos lógicos e operações sobre conjuntos

Conectivo lógico	Operação sobre conjuntos
Idempotência: $\wedge$ e $\vee$	*Idempotência*: intersecção e união
Comutativa: $\wedge$ e $\vee$	*Comutativa*: intersecção e união
Associativa: $\wedge$ e $\vee$	*Associativa*: intersecção e união
Distributiva: $\wedge$ sobre $\vee$ e $\vee$ sobre $\wedge$	*Distributiva*: intersecção sobre união e união sobre intersecção

Conectivo lógico	Operação sobre conjuntos
Dupla negação	*Duplo complemento*
DeMorgan	*DeMorgan*
absorção	*absorção*

2.6 Exercícios

Exercício 2.1: Suponha o conjunto universo $S = \{p, q, r, s, t, u, v, w\}$ bem como os seguintes conjuntos:

- $A = \{p, q, r, s\}$
- $B = \{r, t, v\}$
- $C = \{p, s, t, u\}$

Determine:

a) $B \cap C$

b) $A \cup C$

c) $\sim C$

d) $A \cap B \cap C$

e) $\sim (A \cup B)$

f) $(A \cup B) \cap \sim C$

Exercício 2.2: Considere os conjuntos $A = \{a\}$, $B = \{a, b\}$ e $C = \{0, 1, 2\}$. Calcule os seguintes produtos cartesianos:

1. $(A \times B) \times C$
2. $A \times (B \times C)$
3. B^2
4. C^2

Exercício 2.3: Considere os conjuntos $D = \{0, 1, 2, ..., 9\}$, $V = \{a, e, i, o, u\}$, e $P = \{0, 2, 4, 6, ...\}$. Então, encontre:

1. $D + V$
2. $D + P$
3. $V + V$
4. $V + \emptyset$

Exercício 2.4: Utilizando um banco de dados relacional, crie duas tabelas: *Palavras1* e *Palavras2*, respectivamente. Utilizando linguagem SQL, crie e apresente o resultado de uma consulta que realiza o produto cartesiano entre as duas tabelas.

Exercício 2.5: Crie um programa que lê n arquivos de entrada. Cada arquivo contém uma palavra em cada linha. O programa deve ler os arquivos e gerar um arquivo de saída chamado *pc.txt* contendo o produto cartesiano entre as palavras dos arquivos de entrada. Cada linha do arquivo de saída deve representar um elemento do produto cartesiano (uma n-upla) cujos componentes devem estar separados por um espaço [em branco].

Exemplo: Para os arquivos de entrada:

palavras1.txt
jose
maria

palavras2.txt
silva
santos

palavras3.txt
moreira
aires

o arquivo resultante seria:

pc.txt
jose silva moreira
jose silva aires
jose santos moreira
jose santos aires
maria silva moreira
maria silva aires
maria santos moreira
maria santos aires

2.7 Projeto do capítulo

Este capítulo deu continuidade à Seção 1.2 e apresentou operações e propriedades sobre conjuntos que constituem a **Álgebra de conjuntos**.

Como forma de fundamentar os conceitos apresentados este capítulo traz o **projeto do capítulo**, uma atividade prática que envolve computação e escrita de textos.

O projeto deste capítulo deve alcançar os objetivos:

1. utilizar uma linguagem de programação para criar uma estrutura de dados **Conjunto**, com

as funcionalidades:

- adicionar elemento
- remover elemento
- verificar pertinência
- verificar continência
- realizar união (com outro conjunto)
- realizar intersecção (com outro conjunto)
- realizar diferença (com outro conjunto)
- realizar complemento (em relação a outro conjunto)
- gerar o conjunto das partes
- realizar o produto cartesiano (com outro conjunto)
- realizar a união disjunta (com outro conjunto)

2. escrever um texto didático explicando e demonstrando a implementação e os cocneitos utilizados. Para cada funcionalidade, apresentar: o conceito (definição), a operação (como funciona), a demonstração (com exemplos). Esse texto deve usar notação matemática (o resultado é um arquivo **PDF**).

Importante: No *Objetivo 1* não pode ser utilizado recurso da linguagem que já implemente recursos de conjuntos e opreações sobre conjuntos. Utilize lista, vetor ou outra estrutura de dados semelhante.

Por fim, o conteúdo deve ser disponibilizado em um repositório do Github. Deve haver um arquivo `README.md` identificando e apresentando o trabalho, bem como descrevendo os procedimentos adotados.

Referências

GIT COMMUNITY. **Git**, [s.d.]. Disponível em: <https://git-scm.com/>. Acesso em: 22 jul. 2018

MENEZES, P. B. **Matemática discreta para computação e informática**. Traducao. 3. ed. Porto Alegre: Bookman, 2010.

MICROSOFT. **Visual Studio Code - Code Editing. Redefined**, [s.d.]. Disponível em: <https://code.visualstudio.com/>. Acesso em: 22 jul. 2018a

MICROSOFT. **TypeScript - JavaScript that scales**, [s.d.]. Disponível em: <https://www.typescriptlang.org/index.html>. Acesso em: 24 jul. 2018b

NODE.JS FOUNDATION. **Node.js**, [s.d.]. Disponível em: <https://nodejs.org>. Acesso em: 23 jul. 2018

NPM, INC. **npm**, [s.d.]. Disponível em: <https://www.npmjs.com/>. Acesso em: 23 jul. 2018

THE JQUERY FOUNDATION. **jQuery**, [s.d.]. Disponível em: <http://jquery.com/>. Acesso em: 22 jul. 2018

W3SCHOOLS. **JavaScript Tutorial**, [s.d.]. Disponível em: <https://www.w3schools.com/js/default.asp>. Acesso em: 22 jul. 2018a

W3SCHOOLS. **JavaScript and HTML DOM Reference**, [s.d.]. Disponível em: <https://www.w3schools.com/jsref/default.asp>. Acesso em: 22 jul. 2018b

W3SCHOOLS. **HTML5 Tutorial**, [s.d.]. Disponível em: <https://www.w3schools.com/html/default.asp>. Acesso em: 22 jul. 2018c

W3SCHOOLS. **CSS Tutorial**, [s.d.]. Disponível em: <https://www.w3schools.com/css/default.asp>. Acesso em: 22 jul. 2018d

W3SCHOOLS. **CSS Reference**, [s.d.]. Disponível em: <https://www.w3schools.com/cssref/default.asp>. Acesso em: 22 jul. 2018e

W3SCHOOLS. **HTML Element Reference**, [s.d.]. Disponível em: <https://www.w3schools.com/tags/default.asp>. Acesso em: 22 jul. 2018f