UNIVERSIDADE DE VILA VELHA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

GUILHERME GASPERAZZO ZAMPROGNO
KAUÃ ARAUJO DE SOUZA
KAUÃ PUTTIN PESTANA

APLICAÇÕES DE ÁLGEBRA PARA COMPUTAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

VILA VELHA
2024

GUILHERME GASPERAZZO ZAMPROGNO
KAUÃ ARAUJO DE SOUZA
KAUÃ PUTTIN PESTANA

APLICAÇÕES DE ÁLGEBRA PARA COMPUTAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Trabalho apresentado como forma de avaliação do primeiro bimestre de 2024/2, na disciplina de Álgebra para Computação do curso de Ciência da Computação, na Universidade de Vila Velha.

Professor: Erlon Pinheiro

VILA VELHA
2024

1. INTRODUÇÃO

A Álgebra para Computação é uma área da matemática que trata de assuntos como teoria dos conjuntos, relações e operações entre conjuntos, funções e suas propriedades, entre outros assuntos importantes para o desenvolvimento de tecnologias. Na Ciência da Computação, assume papel fundamental em áreas como estruturas de dados, bancos de dados, grafos, modelagem de redes, criptografias e até mesmo criação de novas linguagens de programação.

2. TEORIA DE CONJUNTOS E ESTRUTURAS DE DADOS

A Teoria dos Conjuntos define o que são conjuntos, subconjuntos e conjuntos das partes, além de estabelecer as operações que podem ser realizadas neles.

2.1 Representação dos conjuntos

A representação de um conjunto **é feita utilizando uma letra maiúscula**, os elementos do conjunto estão sempre entre chaves e separados por vírgula. Por exemplo o conjunto M é o conjunto dos múltiplos de 10: M = {0, 10, 20, 30, 40, 50...}.

2.2 Subconjuntos

Chamamos de subconjunto de um conjunto, um conjunto onde todos os seus elementos pertencem ao conjunto principal ou conjunto universo, por exemplo:
A = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

Subconjuntos:

B = {1, 2, 3}

C = {2, 4, 6, 8, 10}

2.3 Operações binárias e unárias em conjuntos

Uma operação binária em um conjunto S é uma operação entre dois elementos x e y de S, denotada por $x \circ y$. Esta operação está bem definida se $x \circ y$ existir, for único e

pertencer a S. Exemplos: adição, subtração e multiplicação no conjunto dos números inteiros Z.

Uma operação unária em S, denotada por x#, está bem definida se, para qualquer x $\in$ S, x# existir, for único e pertencer a S.

2.4 Operações de união e interseção, complemento e produto cartesiano

- **União de Conjuntos:** Combinação dos elementos de diferentes conjuntos, resultando em um novo conjunto que contém os elementos de ambos os conjuntos, sem repetição. A$\cup$B.
- **Interseção de Conjuntos:** Elementos que são pertencem a ambos os conjuntos. A$\cap$B.
- **Diferença de Conjuntos:** Elementos que pertencem a um conjunto, mas não a outro. A$-$B.
- **Conjunto Complementar:** Todos os elementos de um conjunto universo S que não pertencem a um conjunto A, é o conjunto complementar de A.

2.5 OPERAÇÕES EM CONJUNTOS EM SQL

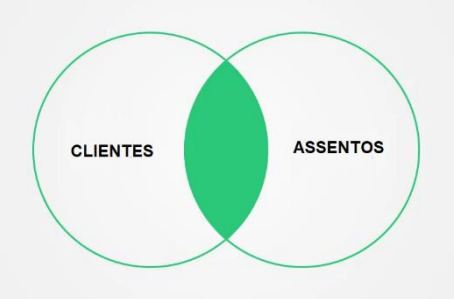
As operações em conjuntos, definidas pela Teoria dos Conjuntos, são muito utilizadas no gerenciamento de banco de dados, com a SQL (Structured Query Language) para combinar dados de duas tabelas diferentes a partir de colunas que possuem alguma coluna em comum entre elas. Isso permite a realização de análises mais aprofundadas, como a garantia de consistência e unicidade dos dados. Além disso, essas operações permitem a criação de relatórios avançados para melhoria de produtos ou serviços.

2.5.1 Operações Join em SQL

Supondo que estamos gerenciando uma base de dados de compra de passagem aérea que contém as tabelas **clientes** e **assentos.** A tabela clientes armazena o *id_cliente* e seus dados, enquanto a tabela assentos armazena a identificação do assento, o *id_cliente* que adquiriu o assento, e o tipo do assento. As tabelas podem ser combinadas com o comando **JOIN** a partir da chave em comum nas duas, que, nesse caso, é o *id_cliente*:

- O **INNER JOIN** pode ser utilizado para listar os clientes que compraram passagens e os assentos que foram ocupados, deixando de fora os clientes cadastrados que não compraram passagens para aquele voo em específico e os assentos que não foram ocupados, o que se assemelha muito à operação de interseção de conjuntos, onde apenas os elementos presentes simultaneamente nos conjuntos são retornados. Exemplo:

```
SELECT c.nome, a.id_assento
FROM clientes c
INNER JOIN assentos a ON c.id_cliente = a.id_cliente;
```



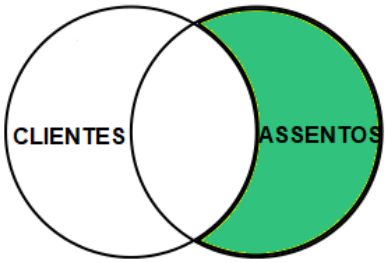
- Para visualizar todos os clientes e assentos, mesmo os clientes que não fizeram reserva de assentos e os assentos não ocupados, o comando utilizado seria o **FULL OUTER JOIN**, que funciona como a operação de união na Teoria dos Conjuntos. Exemplo:

```
SELECT c.nome, a.id_assento
FROM clientes c
FULL OUTER JOIN assentos a ON c.id_cliente = a.id_cliente;
```

- Se for necessário verificar quais assentos ainda não foram ocupados, é possível utilizar o **RIGHT JOIN** exclusivo, que busca os elementos armazenados no conjunto da direita no Diagrama de Venn, excluindo a

interseção com o conjunto da direita, que seria a diferença nas operações de conjunto: ASSENTOS -CLIENTES.

```
SELECT a.id_assento
FROM clientes c
RIGHT JOIN assentos a ON c.id_cliente = a.id_cliente
WHERE c.id_cliente IS NULL;
```



As operações realizadas em tabelas por meio da linguagem SQL podem ainda se relacionar com a classificação quanto à relação binária. Nesse caso, um cliente poderia fazer a reserva de mais de um assento, porém o assento só pode pertencer a um cliente, portanto, a classificação da relação binária nesse caso é um-para-vários.

3. RELAÇÕES BINÁRIAS E SUAS APLICAÇÕES NA COMPUTAÇÃO

Uma relação binária descreve a conexão entre dois elementos de dois conjuntos distintos ou do mesmo conjunto. Em termos matemáticos, uma relação binária entre dois conjuntos **A e B** é definida como um subconjunto do produto cartesiano **A × B**. O produto cartesiano é o conjunto de todos os pares ordenados possíveis formados por elementos de **A e B**. Dessa forma, cada par ordenado **(a, b)** que faz parte da relação binária estabelece uma ligação entre os elementos **a ∈ A** e **b ∈ B**. Agora já entendido o conceito de relação binária, podemos explorar algumas das principais aplicações na computação, especificamente em redes de computadores e grafos.

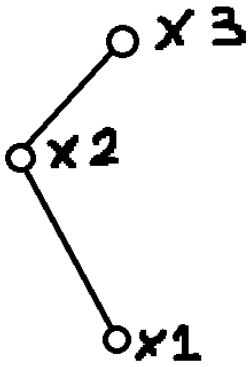
3.1 RELAÇÃO BINÁRIA E MODELAGEM DE GRAFOS EM REDES DE COMPUTADORES E REDES SOCIAIS

A teoria dos grafos é amplamente utilizada em redes de computadores para representar a conectividade entre dispositivos (como roteadores, computadores ou switches) de forma visual e matemática. Nessa modelagem, os dispositivos são representados como nós e as conexões entre eles são as arestas. A relação binária entre dois dispositivos **A e B** pode ser descrita como "conectado diretamente" ou "não conectado diretamente".

3.1.1 ESTRUTURAS DE REDE COMO GRAFO

Em uma rede de computadores, as conexões entre dispositivos podem ser diretas ou indiretas. A relação binária entre dois dispositivos **X1 e X2** é representada como um par ordenado **(X1, X2),** onde a presença de uma aresta no grafo indica uma conexão direta entre eles. Se não houver conexão, o par não faz parte da relação.

- **Exemplo prático**: Imagine uma rede simples com três roteadores **X1, X2 e X3**. Se **X1** estiver conectado a **X2**, mas não a **X3**, a relação binária seria representada pelos pares **(X1, X2) e (X2, X3)**, mas o par **(X1, X3)** não estaria presente.



Essa representação é extremamente útil para modelar e resolver problemas de conectividade e rota de dados. A topologia da rede pode ser descrita como um grafo não-direcionado (quando as conexões são bidirecionais) ou direcionado (quando as conexões têm um sentido, como em redes ponto-a-ponto).

3.2 CONGRUÊNCIA MODULAR

A congruência de módulo **n** é um tipo de relação de equivalência uma relação binária que é reflexiva, simétrica e transitiva. Nessa relação, dois números são considerados congruentes se, ao serem divididos por um número inteiro positivo **n**, deixarem o mesmo resto. Esse número **n** é chamado de módulo.

Por exemplo, na congruência de módulo 5, dizemos que $x \equiv y \pmod 5$ ou $x - y = 5k$, onde k é um número inteiro.

3.2.1 VERIFICAÇÃO DE CPF POR CONGRUÊNCIA DE MÓDULO N

O CPF (Cadastro de Pessoa Física) é um número identificador de pessoas formado por 11 dígitos no total. Os 9 primeiros a base do CPF, sendo que o último deles indica a região onde o CPF foi registrado. Os últimos dois dígitos são verificadores, onde utiliza-se do conceito de congruência modular para formá-los.

Para obter o primeiro dígito verificador, é seguido um algoritmo com os seguintes passos:

- Multiplica-se os nove primeiros dígitos do CPF, começando da direita para a esquerda, por números decrescentes, começando em 10 e terminando em 2;
- Soma-se de todas as multiplicações é somado;
- Divide-se o resultado da soma por 11, se o resto dessa divisão (mod 11) for 0 ou 1, o dígito verificador é 0, se for diferente, o dígito será 11 decrescido do resto da divisão.

Para a obtenção do segundo dígito verificador, utiliza-se um processo semelhante, com algumas diferenças

- Multiplica-se os dez primeiros dígitos, os nove iniciais mais o primeiro dígito verificador, por números decrescentes, começando de 11 e terminando em 2.
- Soma-se o resultado das multiplicações;
- O resultado da soma é dividido por 11, o segundo dígito é determinado a partir do resto da divisão (mod 11):
    - Se o resto for 0 ou 1, o dígito verificador será 0;
    - Se o resto for diferente de 0, o dígito será 11 decrescido do resto da divisão.

3.3 VERIFICAÇÃO DO ISBN

ISBN (International Standard Book Number): ISBN é um identificador único para livros, e não pode ser aplicado diretamente à álgebra. No entanto, a verificação do ISBN pode ser um excelente exemplo prático de um campo da disciplina de Álgebra aplicada à Computação, uma vez que a Álgebra é usada em muitos algoritmos para verificar erros.

Cada ISBN possui um dígito verificador que é resultado de uma determinada fórmula matemática que usa os demais números do conjunto. É possível enxergar essa fórmula como uma aplicação de determinados conceitos de álgebra como somatórias ponderadas e congruências modulares.

Exemplo:

Um **ISBN-10** consiste de 9 dígitos principais mais um dígito verificador.

S = 1 x 1 + 2 x 2 + 3 x 3 +...+ 9 x 9. Onde x 1, x 2... x 9 são os primeiros 9 dígitos do ISBN. O digito verificador (10º dígito) deve ser escolhido de modo que a soma S seja divisível por 11, ou seja: S ≡ 0 (mod 11). Se a soma não for divisível por 11, o ISBN é inválido

Verificação de erros: Algoritmos que usam somas modulares e dígitos verificadores são importantes em sistemas de computação, redes e até bancos de dados para detecção de erros.

Álgebra booleana: o processo de verificação acima pode ser substituído por software e programação lógica com base na álgebra booleana, que é um conceito fundamental em computação.

3.4 CODIGO DE BARRAS

A relação binária define a conexão entre dois conjuntos. Conjunto A, que contém os dados codificados, como números ou letras. Conjunto B que tem as sequências de barras e espaços que representam esses dados visualmente. Cada elemento de A tem uma relação com uma sequência única de barras e espaços de B. Esta relação binária é a base para a decodificação do código de barras, assim podendo converter o padrão físico em informação digital.

A congruência modular é aplicada no cálculo do dígito verificador de muitos sistemas de código de barras, como o **EAN-13**. O dígito verificador serve para garantir a integridade e a correta leitura dos dados codificados no código de barras. É calculado usando operações modulares (geralmente mod10).

3.4.1 COMO FUNCIONA O CÁLCULO COM CONGRUÊNCIA MODULAR

Os dígitos do código de barras são somados com base em suas posições (pares e ímpares), com diferentes pesos atribuídos aos dígitos.

A soma total é usada para calcular o dígito verificador, que deve satisfazer uma congruência modular, como: **$S \equiv 0 \pmod{10}$**.

Onde S é o resultado da soma ponderada dos dígitos. Isso significa que o resto da divisão de S por 10 deve ser zero, garantindo que o código seja válido.

Exemplo:

Imaginando que temos um código de barras **EAN-13** com os primeiros 12 dígitos:

123456789012

Para calcular o dígito verificador primeiro temos quer somar os dígitos das posições ímpares e multiplique por 3: (1+3+5+7+9+1) × 3 = 81. Depois some os dígitos das posições pares: 2+4+6+8+0+2=22. E some os dois resultados que é: 81+22=103.

Agora devemos calcular o dígito verificador que tornará **103 + d** congruente a 0 módulo 10: **$103 + d \equiv 0 \pmod{10}$**. O valor mais próximo de 103 que é múltiplo de 10 é 110. O dígito verificador **d** será 110−103=7.

Portanto, o dígito verificador é **7**, o que significa que o código completo será **1234567890127.**

4. FUNÇÕES, TEORIA DOS NUMEROS E CRIPTOGRAFIA

A criptografia é uma técnica usada para proteger informações, transformando dados legíveis em um formato que só pode ser decifrado por alguém que tenha uma chave específica. O RSA é um dos algoritmos mais usados para criptografia e é essencial para garantir a segurança das informações trocadas na internet, como em compras online ou mensagens privadas. Em resumo, o RSA é como uma troca de segredos

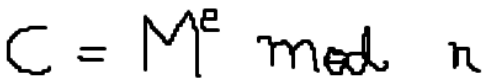
usando cadeados: qualquer pessoa pode trancar a mensagem, mas só quem tem a chave certa pode abrir e ler.

Na criptografia, as funções desempenham um papel crucial, especialmente no processo de criptografia de dados. Um exemplo é o uso de funções injetivas e bijetivas. Uma função é injetora quando cada elemento do conjunto de saída tem apenas uma correspondência única no conjunto de entrada. Na criptografia, isso significa que uma mensagem criptografada deve ter apenas uma mensagem original correspondente, garantindo que não haja ambiguidade na descriptografia.

No algoritmo RSA, a função usada para criptografar e descriptografar mensagens é uma função bijetora, o que significa que é injetora (sem colisões) e sobrejetora (cada elemento de saída é coberto). Essa função é o coração da criptografia, porque garante que cada mensagem criptografada possa ser revertida para sua mensagem original, usando a chave certa. Essa propriedade de injeção e reversibilidade é essencial para que a criptografia funcione corretamente e mantenha a segurança da informação. Em resumo, as funções na criptografia garantem que o processo de transformação (criptografia) e de retorno (descriptografia) sejam seguros e únicos, protegendo a integridade e a confidencialidade dos dados.

Exemplos:

Criptografia com chave pública:



M é a mensagem original (convertida para números). **e** é parte da chave pública do destinatário. **n** é o produto de dois números primos. **(n = p × q).**

C: Significa que para cada **M**, existe um único **C.**

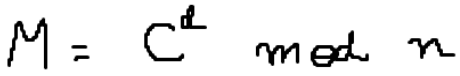
Imagine que Erlon Filho queira enviar uma mensagem **(M = 5)** para Erlon Pai. Ele usa a chave pública de Erlon Pai **(e = 3, n = 33)** para criptografar:

$C = 5^3 = 125$

125 mod 33 = 26

A mensagem criptografada é **C = 26.**

E temos a descriptografia com chave privada:



d é a parte secreta da chave privada. O destinatário usa **d** para recuperar a mensagem original **M** a partir da mensagem criptografada **C**.

D: Garante que apenas quem tem a chave privada correta pode decifrar a mensagem.

Agora que Erlon Pai recebeu a mensagem criptografada C = 26, ele usa sua chave privada d = 7 e n = 33 para descriptografar a mensagem. M = Cd mod n

M = 26^7 mod 33 □ 26^7 = 8031810176

Agora, vamos calcular 8031810176 mod 33 que nos dá:

8031810176 ÷ 33 = 243752126 **(resto 5)** □ **M = 5**

Assim, Erlon Pai recupera a mensagem original **M=5**, que foi enviada por Erlon Filho.

4.1 TEORIA DOS NUMEROS

A teoria dos números também tem uma aplicação direta na criptografia, especialmente no uso de números primos e congruências. O algoritmo RSA depende da fatoração de grandes números primos e da aritmética modular, ambos elementos da teoria dos números.

Números Primos e Fatoração: O RSA, por exemplo, baseia-se na dificuldade de fatorar o produto de dois grandes números primos. A segurança do sistema depende da impossibilidade prática de fatorar números enormes, o que é essencial para proteger a chave privada. Além disso, utiliza congruências para gerar as chaves e operar com elas.

Exemplo: Se n = pq, onde p e q são números primos grandes, é extremamente difícil para um hacker fatorar n para descobrir p e q, tornando o RSA seguro. Esses dois aspectos mostram como a função e a teoria dos números sustentam a criptografia moderna.

5. INDUÇÃO MATEMÁTICA

A indução matemática é um método de prova, usado para mostrar que uma certa afirmação ou propriedade é verdadeira para todos os elementos de determinado conjunto. Temos 2 passos importantes.

Caso base: Primeiro, você verifica se a afirmação é verdadeira para o menor valor possível.

Passo indutivo: Você assume que a afirmação é verdadeira para n = k e, com base nessa suposição, prova que a afirmação também é verdadeira para n = k + 1 (próximo número).

A indução matemática é amplamente utilizada em computação para provar a funcionalidade e eficiência, análise e complexidade de algoritmos, principalmente recursivos. Usamos a indução para provar que o tempo de execução de um algoritmo cresce de acordo com uma determinada função matemática, o que nos ajuda a entender o comportamento de um algoritmo em grandes entradas. Por exemplo, ao analisar o algoritmo do caixeiro viajante, podemos utilizar a indução para provar que o problema não seria resolvido em tempo hábil para humanos, ou seja, é um algoritmo ineficiente de complexidade O(n!).

6. CONCLUSÃO

Neste trabalho, vimos como a álgebra é essencial na computação. A Teoria dos Conjuntos nos ajuda a entender operações importantes como união, interseção e diferença, usadas em estruturas de dados e sistemas de banco de dados. As relações binárias, por sua vez, são fundamentais para modelar redes de computadores e redes sociais, conectando elementos de diferentes conjuntos. Também exploramos a aplicação de congruências em verificações de CPF, ISBN e Código de barras, mostrando a utilidade da álgebra em garantir a integridade de dados. A criptografia, por meio da Teoria dos Números e funções, foi mais um exemplo de como a álgebra protege a troca de informações. Por fim, a indução matemática para provar a eficiência de algoritmos. Assim, fica evidente que a álgebra é uma ótima ferramenta para o desenvolvimento de soluções tecnológicas.

7. REFERÊNCIAS BILIOGRÁFICAS

- https://pt.wikipedia.org/wiki/Teoria_das_redes_complexas
- https://csd.uwo.ca/~abrandt5/teaching/DiscreteStructures/Chapter7/index.html
- https://www.gta.ufrj.br/ensino/eel878/redes1-2018-1/trabalhos-vf/p2p/grafos.html
- https://www.inovaproj.com.br/engenharia/matemática-computacional/artigos/teoria-dos-grafos-explorando-as-conexões-e-relações-complexas
- https://www.ufrgs.br/cespri/projects/analise-de-redes-sociais-usando-a-teoria-dos-grafos/
- https://www.facebook.com/wvsbrasil/videos/495768424238127/
- https://ilos.com.br/teoria-dos-grafos-e-analise-de-redes/
- https://pt.wikipedia.org/wiki/Rede_de_computadores
- https://coens.dv.utfpr.edu.br/will/wp-content/uploads/2022/03/Apostila_Algebra_Relacional.pdf
- https://www.estrategiaconcursos.com.br/blog/banco-dados-descomplicado-algebra-relacional/
- https://www.ime.usp.br/~jef/bd05.pdf
- https://sca.profmat-sbm.org.br/profmat_tcc.php?id1=2423&id2=84479
- https://repositorio.ufms.br/bitstream/123456789/1746/1/Josiane%20Colombo%20Pedrini%20Esquinca%281%29.pdf
- http://dspace.nead.ufsj.edu.br/trabalhospublicos/bitstream/handle/123456789/61/CLAUDEMILSON%20DA%20SILVA%20OLIVEIRA_12313_assignsubmission_file_TCC_CLAUDEMILSON_versao_final.pdf?sequence=1&isAllowed=y