INTRODUÇÃO

1-Conceito

Matemática Discreta é um ramo da Matemática que lida com objetos discretos, objetos que tomam valores separados e distintos. A mesma é melhor entendida quando contrastada com Matemática Contínua, que lida com objetos cujos valores podem variar continuamente, suavemente. Cálculo é um dos exemplos mais concretos de Matemática Contínua, enquanto que Análise Combinatória, o exemplo mais conhecido de Matemática Discreta. O estudo da Matemática Discreta permite o desenvolvimento de uma certa intuição sobre a natureza de alguns problemas com os quais a Matemática lida e sobre porque os métodos matemáticos são úteis.

2-Relações

Uma relação é do tipo um para um se cada primeira componente (s1) e cada segunda componente (s2) do par ordenado aparece uma única vez na relação. Uma relação é do tipo um para muitos se alguma primeira componente (s1) aparece em mais de um par. A relação é dita muitos para um se alguma segunda componente s2 aparecer em mais de um par. Finalmente, a ela é muitos para muitos se, pelo menos, um s1 aparece em mais de um par e, pelo menos, um s2 também aparece em mais de um par. Os tipos de relações podem ser melhor entendidas conforme abaixo:

2.1 Relação Binária

Em geral, uma relação binária é definida por uma descrição da relação, ao invés da lista dos pares ordenados. A descrição fornece uma caracterização dos elementos pertencentes à relação.

Ex: Dados dois conjuntos A e B, uma relação binária R de A em B é um subconjunto de um produto cartesiano $A \times B$, ou seja, $R \subseteq A \times B$, onde:

– A é o domínio, origem ou conjunto de partida de R

– B é o contradomínio, destino ou conjunto de chegada de R. Para $R \subseteq A \times B$, se a, b $\in$ R, então afirmamos que “a relaciona-se com b”. Podemos denotar uma relação R da seguinte forma: $R: A \rightarrow B$ e, para um elemento a, b $\in$ R, podemos denotá-lo como aRb.

2.2 Endorrelação como Grafo

Toda relação $R: A \rightarrow B$ pode ser representada a partir de um grafo direcionado com arestas ligando cada par ordenado (a, b), com origem em a e destino em b, ou seja, dado um conjunto A, uma relação do tipo R: A → A é dita uma Endorrelação ou Auto-Relação. Assim, temos que origem e destino são o mesmo conjunto e podemos denotá-la por A, R .

2.3 Relação Matriz

A relação R: A→B pode ser representada na forma de matriz, o que facilita sua implementação em sistemas computacionais.

Sejam A = {a 1, a 2, …, a n } e B = {b 1, b 2, …, b m } dois conjuntos finitos. A representação da relação R: A → B como matriz é como segue:

a) o número de linhas é n (número de elementos do domínio);

b) o número de colunas é m (número de elementos da imagem);

c) a matriz resultante possui m x n células;

d) cada uma das m x n células possuem um valor lógico associado;

e) se a i, b j ∈ R, então a posição determinada pela linha i e pela coluna j da matriz contém valor verdadeiro (1); caso contrário, seu valor será falso (0).

3-As Propriedades das Relações

3.1 Reflexiva: A relação *R* é dita reflexiva se todos os elementos se relacionam consigo mesmo. Formalmente, a relação *R* é dita reflexiva se *aRa* para todo *a* ∈ *A*, isto é, se (a, a) ∈ *R* para todo *a* ∈ *A*.

Exemplos: Em um conjunto qualquer, podemos dizer que existe relação reflexiva se os subconjuntos deste conjunto possuírem os mesmos elementos, por exemplo:

Conjunto A= { a, b, d, z }

Relações:

R_1 = { (a, b), (b, a), (b, z) } Não é reflexiva

R_2 = { (a, b), (b, d), (z, z) } Não é reflexiva

R_3 = { (a, a), (a, b), (b, b), (d, d), (z, z) } É reflexiva

3.2 Simétrica: indica que toda vez que um elemento estiver relacionado com outro, a vice-versa também estará relacionada. Em um conjunto *R* qualquer dizemos que a relação é simétrica quando *(a, b)* pertence ao conjunto *R* e obrigatoriamente o subconjunto *(b, a)* também pertença ao conjunto *R*, ao contrário a relação será antissimétrica.

Exemplo: *Conjunto L = { 2, 4, 6, 8 }*

Relações:

R_1 = {(2,4), (2,6), (6,8)} É antissimétrica

R_2 = {(2,4), (2,6), (4,2), (6,8)} É simétrica

R_3 = { (2,4), (2,6), (2,8), (4,2), (8,2), (6,2)} É simétrica

3.3 Transitiva:

A relação R é transitiva em um conjunto, se a seguinte relação for satisfeita: (a, b) pertence a R e (b, c) pertence a R implica que (a, c) pertença a R, caso contrário a relação não é transitiva.

Exemplo: Conjunto A

A= { a, b, c, d }

Relações:

R_1= { (a, b), (a, d), (b, c), (c, d) } Não é transitiva

R_2= { (a, b), (a, c), (b, c), (a, c) } É transitiva

R_3= a<=b, b<=c então a<=c, É transitiva

4-Fechos de relações

4.1 Fecho Reflexivo

Um fecho é reflexivo quando adicionamos os elementos (a, a), (b, b)… que não pertencem à relação R. E um fecho é simétrico quando adicionamos os elementos (b, a) tais que (a, b) pertence à R.

Exemplo: Conjunto A

A= { a, b, c, d }

Relações:

R_1= { (a, b), (b, b), (a, d), (c, d), (d, d) }

R_1= { (a, a), (a, b), (b, b), (a, d), (c, c), (c, d), (d, d) } Fecho Reflexivo

R_1= { (a, b), (b, a), (b, b), (a, d), (d, a), (c, d), (d, c), (d, d) } Fecho Simétrico

4.2 Fecho Simétrico

Em matemática, o fecho simétrico de uma relação binária R em um conjunto é a menor relação simétrica em X que contém R.

O fecho simétrico *S* de uma relação *R* em um conjunto *X* é dado por:

$$S = R \cup \{(x,y) : (y,x) \in R\}.$$

Em outras palavras, o fecho simétrico de *R* é a união de *R* com a sua relação inversa, R^{-1}

4.3 Fecho Transitivo

Seja R uma relação em um conjunto A. A relação de conectividade R consiste dos pares (a; b) de forma que existe um caminho de a para b em R.

Exemplo: Dados o conjunto A = {1, 2, 3, 4, 5} e R: A → A uma endorrelação, tal que R = {1,2, 1, 5, 2, 3, 3, 4}, temos que Fecho – { transitiva }(R) = { 1, 2, 1, 3, 1, 4, 1, 5, 2, 3, 2, 4, 3, 4 }

Algoritmo: Relação binária

Estruturas de dados utilizadas:

O programa Relação binária, desenvolvido em C#, utiliza as seguintes estruturas de dados:

1. Array bidimensional: O array bidimensional é utilizado para receber os dados em pares.
2. Lista genérica: Uma lista genérica, em C#, pode ser de qualquer tipo de dados. Neste caso, uma lista de strings.
3. Ordenação de dados: A ordenação de dados é utilizada para ordenar a lista citada anteriormente.

Algoritmo de descrição narrativa

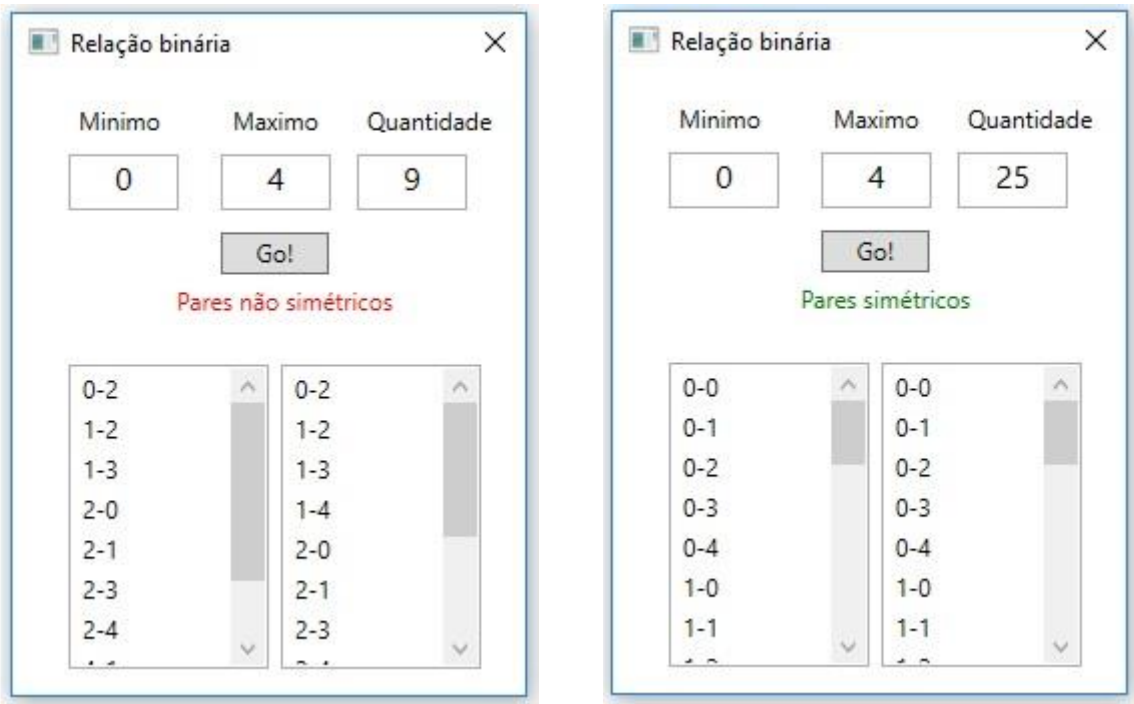

Figura 1Programa relação bináriahttps://github.com/cassiofernandes/C_Sharp/tree/master/relacao_binaria

- Receber dados do usuário (classe MainWindow.xaml.cs) - O usuário informa valores mínimos e máximos para os intervalos e uma quantidade de linhas.

1. Se algum campo estiver em branco, uma mensagem alerta para que o usuário insira valores. Caso contrário, verifica os valores inseridos.
2. Se o valor mínimo for maior que o máximo, uma mensagem alerta que o intervalo mínimo não pode ser maior que o intervalo máximo.
3. Se o intervalo máximo for superior a 10^2, uma mensagem alerta para o valor máximo. Caso contrário, verifica a quantidade máxima de linhas.
4. A quantidade máxima de linhas deve ser menor ou igual a $((\text{intervaloMaximo} - \text{intervaloMínimo}) + 1)^2$; caso contrário, uma mensagem alerta a quantidade máxima de linhas permitidas para o intervalo informado.

- Gerar relações binárias e fecho simétrico (classe ParesBinarios.cs) – Obter intervalo mínimo, máximo e quantidade de linhas.

1. Inicia-se um loop **do...while**.
2. Uma variável (linhas) recebe o valor da quantidade de linhas (Q);
3. Uma variável do tipo **array** bidimensional, de tamanho 1-2, recebe valores aleatórios, um para cada posição.
4. Em uma lista genérica (lista1), inicialmente nula (vazia), verifica-se se os valores gerados já constam na lista.
5. Se não constam, a lista recebe os valores gerados no **array**. A variável linhas é decrementada com -1 (linha--);
6. Lista1 é ordenada.
7. Em uma lista genérica (lista2), inicialmente nula (vazia), verifica-se se os valores gerados já constam na lista.
8. Se não constam, a lista2 recebe os valores gerados no array e seu inverso (se **array** = (1, 2), lista2 recebe (1, 2) e (2, 1));
9. Lista2 é ordenada.
10. Se já constam, em lista1, novos números são gerados.
11. O loop dura enquanto linha > 0;
12. Um loop **foreach,** verifica-se se todos os elementos de lista1 constam em lista2.
13. Se sim, é retornado, para a classe principal, um valor verdadeiro (**true**);
14. Se não, é retornado, para a classe principal, um valor falso (**false**);

- Exibir resultados (classe MainWindow.xaml.cs) – Obter lista1 e lista2

1. Se for retornado um valor verdadeiro, um texto exibirá a frase “ Pares simétricos”;
2. Se for retornado um valor falso, um texto exibirá a frase “ Pares não simétricos”;
3. Uma lista, na interface, é preenchida com os valores gerados em lista1.
4. Uma lista, na interface, é preenchida com os valores gerados em lista2;

Referências bibliográficas:

Disponível em: http://pt.slideshare.net/UlrichSchiel/matemtica-discreta-parte-v-relaes , acessado em 16/05/2016

Disponível em: http://pt.slideshare.net/mechjunior/apostila-matemtica-discreta , acessado em 16/05/2016

Disponível em: http://www.univasf.edu.br/~jorge.cavalcanti/Mat_Disc_Parte11.pdf , acessado em 16/05/2016