**FUNDAÇÃO CENTRO DE ANÁLISE, PESQUISA E INOVAÇÃO TECNOLÓGICA - FUCAPI**

**CURSO DE SISTEMA DE INFORMAÇÃO**

**TEMA: RELATÓRIO DE LÓGICA E MATEMÁTICA DISCRETA**

**Relações e Propriedades de uma Relação Binária e Fechos da Relação**

**Reflexivo, Simétrica, Transitiva e Anti – Simétrica**

MANAUS

2016

**FUNDAÇÃO CENTRO DE ANÁLISE, PESQUISA E INOVAÇÃO TECNOLÓGICA - FUCAPI**

**CURSO DE SISTEMA DE INFORMAÇÃO**

**ALUNOS:**

**ÁDRIAN FARIAS DE SOUZA**

**EWERTON BARBOSA DA SILVA**

**FELIPE FERNANDO DA SILVA VIEIRA**

**WELLINGTON DA CRUZ TERRA**

**TEMA: RELATÓRIO DE LÓGICA E MATEMÁTICA DISCRETA**

Trabalho solicitado pelo professora Josiane Rodrigues, do componente curricular CPT030 Lógica e Matemática Discreta, das turmas INF02NA e ENG02NA, dos cursos de Sistema de Informação e Eng de Software da FUCAPI.

**MANAUS**

**2016**

**SUMÁRIO**

**1.** **INTRODUÇÃO** 4

**2.** **REFERENCIAL TEÓRICO** 4

**2.1.** **RELAÇÕES E FUNÇÕES** 4

**2.2.** **PARES ORDENADOS** 5

**2.3.** **RELAÇÕES BINÁRIAS** 5

**2.4.** **PROPRIEDADES DAS RELAÇÕES** 6

**2.4.1.** **PROPRIEDADE REFLEXIVA** 6

**2.4.2.** **PROPRIEDADE SIMÉTRICA** 6

**2.4.3.** **PROPRIEDADE TRANSITIVA** 6

**2.4.4.** **PROPRIEDADE ANTI - SIMÉTRICA** 7

**2.5.** **FECHO DE UMA RELAÇÃO (ou Fechamento de uma Relação)** 7

**3.** **DESENVOLVIMENTO DO SOFTWARE** 8

**3.1.** **O ALGORITMO** 8

**3.1.1.** **Entrada de dados** 8

**3.1.2.** **Gerando a Relação** 8

**3.1.3.** **Verificando se a Relação é Simétrica** 9

**3.1.4.** **Gerando o Fecho Simétrico** 9

**3.1.5.** **Ordenando os dados** 10

**3.1.6.** **Apresentação dos dados de saída** 11

**3.2.** **Estruturas de Dados** 12

**4.** **REFERÊNCIAS BIBLIOGRÁFICAS** 12

**RELATÓRIO DE LÓGICA E MATEMÁTICA DISCRETA**

1. **INTRODUÇÃO**

O presente trabalho tem por objetivo desenvolver e apresentar um software que determine se uma relação é simétrica ou não e, caso negativo, apresente o seu Fecho Simétrico.

A relação de entrada deve ser determinada a partir do domínio, gerado aleatoriamente a partir dos parâmetros informados pelo usuário.

1. **REFERENCIAL TEÓRICO**
   1. **RELAÇÕES E FUNÇÕES**

A Matemática Contínua: trata de conjuntos similares ao conjunto de números reais, onde os conjuntos têm representações contínuas (do ponto de vista geométrico) e não podem ser enumerados, como por exemplo o intervalo de uma reta, uma região do plano.

Já a Matemática Discreta: trata de objetos separados e desconectados (geometricamente) de cada um, onde os conjuntos são, em geral, finitos e podem ser enumerados. Seu estudo tem como alguns de seus objetivos:

* Desenvolver a capacidade de raciocínio lógico matemático;
* Obter uma visão abrangente de uma parte significativa da computação;
* Aplicar os conceitos da disciplina como uma ferramenta matemática para investigações e aplicações precisas em computação;
* Abordar problemas aplicados e enfrentar ou propor com naturalidade novas tecnologias.
  1. **PARES ORDENADOS**

Um PAR ORDENADO, denotado por (x,y), é um par de elementos onde x é o primeiro elemento e y é o segundo elemento do par.

A ordem é relevante em um par ordenado;

Logo, os conjuntos {a,b} e {b,a} são iguais, mas os pares ordenados (a,b) e (b,a) são diferentes.

A representação de pontos em um plano cartesiano é um exemplo comum de pares ordenados: o ponto (2,1) é diferente do ponto (1,2);

Os pares ordenados (x,y) e (z,w) são iguais se e somente se x = z e y = w .

* 1. **RELAÇÕES BINÁRIAS**

Uma relação binária é definida como sendo um subconjunto do produto cartesiano entre o conjunto **A** e o conjunto **B**. Isto é, uma relação **S** é um conjunto de pares ordenados. Um subconjunto de **A x A** pode ser chamado simplesmente de relação binária em **A**.

Uma relação binária pode ser definida por uma descrição da relação, ao invés da lista dos pares ordenados. Um conjunto **S** pode ter determinadas propriedades:

* Reflexividade.
* Simetria.
* Transitividade.
* Anti - Simetria.

Determinados pares ordenados de objetos em um conjunto de pares ordenados se destacam dos demais porque seus elementos satisfazem alguma relação que os componentes dos demais pares, em geral, não satisfazem.

Exemplos:

- Sejam os conjuntos S = {1,2} e T = {2,3}

- O produto cartesiano é o conjunto S X T = {(1,2), (1,3), (2,2), (2,3)}

Relação de Igualdade:

O conjunto de pares que atende essa relação é unitário: {(2,2)}

Relação formada pelos pares com primeiro número menor que o segundo:

{(1,2), (1,3), (2,3)}

Uma Relação Binária pode ser descrita com palavras ou simplesmente pela enumeração dos pares ordenados que a satisfaça:

Sejam os conjuntos S = {1,2} e T = {2,3,4}

S X T = { (1,2), (1,3), (1,4), (2,2), (2,3), (2,4) }

Uma relação R definida por: x R y, tal que x = y/2, poderia também ser definida pela declaração que {(1,2), (2,4)} é o conjunto dos pares ordenados que satisfaçam R.

A descrição que fornece uma propriedade característica dos elementos da relação é chamada é chamada de Predicado Binário da relação, a qual é realizada por certos pares ordenados do produto cartesiano.

Relações Binárias em um conjunto S podem ter certas propriedades . Seja R uma Relação Binária em S.

* 1. **PROPRIEDADES DAS RELAÇÕES**
     1. **PROPRIEDADE REFLEXIVA**

Reflexiva é uma relação na qual todo elemento está relacionado consigo mesmo. Na igualdade sobre os números reais isso é claramente verificado.

R é REFLEXIVA se:

(∀x)(x ∈ S, → (x,x) ∈R)

• Exemplo 1: Sejam as relações sobre o conjunto S = {1, 2, 3, 4}

R1= {(1,1), (1,2), (2,1), (2,2), (3,4), (4,1), (4,4)}

R2 = {(1,1), (1,2),(2,1)}

R3 = {(1,1), (1,2), (1,4), (2,1), (2,2), (3,3), (4,1), (4,4)}

R4 = {(2,1), (3,1), (3,2), (4,1), (4,2), (4,3)}

R5 = {(1,1), (1,2), (1,3), (1,4), (2,2), (2,3), (2,4), (3,3), (3,4), (4,4)}

R6 = {(3,4)}

As relações R3 e R5 são Reflexivas, as demais não

* + 1. **PROPRIEDADE SIMÉTRICA**

Essa propriedade indica que toda vez que um elemento estiver relacionado com outro, a vice-versa também estará relacionada. Numa relação de parentesco entre duas pessoas isso é um fato verdadeiro.

R é SIMÉTRICA se:

(∀x)(∀y) [(x ∈ S) ∧ (y ∈ S) ∧ (x,y) ∈ R → (y,x) ∈ R].

As relações R2 e R3 do Exemplo 1 acima são Simétricas

* + 1. **PROPRIEDADE TRANSITIVA**

Transitiva significa que se um número relaciona-se com um segundo número e este com um terceiro, pode-se obter a relação do primeiro com o terceiro. A relação menor sobre os números naturais ilustra essa propriedade.

R é TRANSITIVA se:

(∀x) (∀y) (∀z) [(x ∈ S) ∧ (y ∈ S) ∧ (z ∈ S) ∧ ((x,y) ∈ R) ∧ ((y,z) ∈ R) → ((x,z) ∈ R)]

As relações R4, R5 e R6 do Exemplo 1 acima são Transitivas

* + 1. **PROPRIEDADE ANTI - SIMÉTRICA**

A propriedade anti-simétrica não é o inverso da propriedade simétrica e sim estabelece que não seja possível inverter os elementos, ou seja, (y, x) pertence R, a somente quando x=y.

R é ANTI-SIMÉTRICA se:

(∀x)(∀y) [(x ∈ S) ∧ (y ∈ S) ∧ ((x,y) ∈ R) ∧ ((y,x) ∈ R)) → y = x ]

* 1. **FECHO DE UMA RELAÇÃO (ou Fechamento de uma Relação)**

Definição Formal:

Uma Relação Binária R\* em um Conjunto S é um Fecho de uma relação R em S com respeito à propriedade P (Transitiva, Simétrica ou Reflexiva) se:

1. R\* tem a propriedade P

2. R pertence R\*

3. R\* é um subconjunto de qualquer outra relação em S que inclui R e tem a propriedade P

A relação S = {1, 2, 3} e R uma relação binária definida pelo conjunto

R = {(1,1), (1,2), (1,3), (3,1), (2,3)}

R não é reflexiva, não é simétrica e não é transitiva.

Se uma relação R em um conjunto S não tem certa propriedade, podemos tentar estender R para obter a relação R\* em S que tenham a propriedade.

A nova relação R\* conterá todos os pares ordenados que R contém mais os pares ordenados adicionais necessários para que a propriedade desejada se verifique

No exemplo acima, o fechamento de R em relação à REFLEXIVIDADE é

R\* = {(1,1), (1,2), (1,3), (3,1), (2,3), (2,2), (3,3)}

No exemplo acima, o fechamento de R em relação à SIMETRIA é

R\* = {(1,1), (1,2), (1,3), (3,1), (2,3), (2,1), (3,2)}

No exemplo acima, o fechamento de R em relação à TRANSITIVIDADE é

R\* = {(1,1), (1,2), (1,3), (3,1), (2,3), (3,2), (3,3), (2,1)} (passo 1)

R\*= {(1,1), (1,2), (1,3), (3,1), (2,3), (3,2), (3,3), (2,1), (2,2)} (passo 2)

Essa maneira de determinar o fecho Transitivo de uma relação verificando os pares ordenados na relação original, incluindo novos pares se necessário, verificando a relação obtida, incluindo novos pares se necessário e assim por diante, até obtermos a relação transitiva, é um método de força bruta. O uso de algoritmos sobre grafos direcionados são mais eficientes.

1. **DESENVOLVIMENTO DO SOFTWARE**
   1. **O ALGORITMO**

O software foi desenvolvido respeitando 5 etapas principais, que são:

1. Entrada de dados pelo usuário e validação de acordo com as especificações solicitadas;
2. Geração do domínio da Relação através do sorteio de números aleatórios dentro dos parâmetros informado pelo usuário;
3. Verificação da propriedade simétrica na Relação que foi gerada;
4. Geração do Fecho Simétrico caso a Relação gerada aleatoriamente não seja simétrica;
5. Ordenação dos dados de saída, de acordo com o enunciado;
6. Apresentação dos dados de saída;
   * 1. **Entrada de dados**

De acordo com o enunciado do problema, o primeiro (variável m) e último (variável n) elemento do domínio devem estar dentro do intervalo 0≤m,n≤100 e a quantidade de pares ordenados (q) da Relação a ser gerada a partir dos elementos do Domínio, sendo que esta quantidade deve estar no intervalo 1≤q≤10000.

Além desses limites iniciais, no momento da leitura da quantidade de pares ordenados da Relação, deve-se levar em conta o número máximo de pares ordenados possíveis de serem gerados, a partir da quantidade de termos contidos no Domínio. Esse número é dado pela fórmula do Arranjo com repetição, uma vez que os números podem se repetir no sorteio dos pares, ex: (2,2). Essa quantidade máxima é dada pela fórmula:

AR^n_r = n^r

Arranjo de r elementos, organizados n a n.

Portanto, no caso do algoritmo em questão, número máximo de pares ordenados (organizados 2 a 2), é obtido pela fórmula:

**q2**, onde q é o número de elementos do Domínio e 2 é a quantidade de elementos do par ordenado.

* + 1. **Gerando a Relação**

Uma vez validados os parâmetros de entrada, inicialmente o algoritmo aloca dinamicamente um espaço na memória para acomodar uma matriz de q linhas por 2 colunas.

A partir daí, utilizando a fórmula m+rand()%(n-m+1), a cada interação, vai sorteando um número aleatório dentro do intervalo entre m e n, e atribuindo aos elementos da matriz.

Ao mesmo tempo em que sorteia os pares ordenados, o algoritmo verifica se o par gerado não está repetido, evitando a ocorrência de dois pares (0,1) por exemplo.

Caso verifique que o par gerado é repetido, através da condição:

if ((j==1)&&(rel[i][0]==rel[k][0])&&(rel[i][1]==rel[k][1]))

o algoritmo descarta os valores e sorteia novamente até que o par seja diferente de um já existente.

Esse sorteio ocorre até que sejam gerados “q” pares ordenados diferentes, gerando a relação **R**.

* + 1. **Verificando se a Relação é Simétrica**

A partir da geração da relação, o algoritmo percorre todos os pares ordenados que foram gerados para verificar se é verdadeira a condição:

(x,y) ∈ R→(y,x) ∈ R

Para controle, a cada interação, se é encontrado um par ordenado que tem um correspondente simétrico, a variável aux incrementa de 1 unidade. Sendo que ao final das interações, essa variável terá armazenada a quantidade de pares da Relação que possuem um par simétrico.

Essa informação é importante pois o algoritmo segue o seguinte raciocínio para definir se a Relação é simétrica:

Se aux = q (número de pares simétricos = número de elementos da relação), então a relação é simétrica e a variável booleana “simetrico” recebe o valor “true”.

Se aux é diferente de q, então a variável booleana “simetrico” recebe o valor “false” e o fecho simétrico da relação terá (q + (q-aux)) elementos, ou seja, o fecho deverá possuir todos os elementos da relação (q elementos) mais todos os pares ordenados faltantes para tornarem simétricos aqueles pares da Relação que ainda não possuem um par simétrico (q-aux).

* + 1. **Gerando o Fecho Simétrico**

Caso a relação não seja simétrica estrutura de decisão: if (!simetrico), o programa aloca dinamicamente um espaço na memória para a matriz que será utilizada para guardar os elementos do Fecho. Essa matriz terá “QtdElemFecho” linhas por 2 colunas (par ordenado).

O raciocínio utilizado é o seguinte:

-Como o Fecho da Relação sempre contém todos os elementos da Relação original, o algoritmo inicia copiando o primeiro elemento da Relação para ser o primeiro elemento do Fecho e incrementa o contador k, preparando para quando for fazer a próxima cópia de elementos para dentro da matriz fechoS. Utiliza também uma variável lógica “achou” para identificar se o algoritmo encontrou o par simétrico ao atual. O valor “false” é sempre atribuído a essa variável sempre que for copiado um novo valor para a matriz fechos, preparando para a comparação do próximo par ordenado da Relação.

fechoS[k][0]=rel[i][0];

fechoS[k][1]=rel[i][1];

k=k+1;

achou=false;

A partir daí, para cada linha da matriz de Relação gerada (rel), cada elemento (x,y) é comparado com todos os outros elementos para verificar se existe o par (y,x) ou se o par ordenado é constituído por dois valores iguais. Em ambos os casos, se o teste for “verdadeiro”, não é necessário gerar o par simétrico e a variável “achou” recebe o valor “verdadeiro”.

for(j=0;j<q;j++)

{

if (!achou)

if ((rel[i][0]==rel[i][1])||(((rel[i][0]==rel[j][1])&&(rel[i][1]==rel[j][0]))))

{

achou=true;

}

else

achou=false;

}

Caso o teste seja falso, gera-se o par simétrico e incrementa-se o contador, preparando o índice da próxima linha da matriz para receber o próximo elemento, caso ele exista.

if (!achou)

{

fechoS[k][0]=rel[i][1];

fechoS[k][1]=rel[i][0];

k=k+1;

}

* + 1. **Ordenando os dados**

Após o Fecho simétrico ter sido gerado, utiliza-se dois loops de oredenação para que os dados estejam organizados de acordo com o enunciado do problema, ou seja, os pares devem ser ordenados duas vezes. A primeira em ordem crescente do primeiro elemento do par ordenado e a segunda vez ordenar em ordem crescente do segundo elemento do para ordenado.

A ordenação é feita usando o seguinte raciocínio:

Cada elemento do Fecho, é comparado com os demais e, caso encontre um valor menor do que o atual (que está sendo comparado), esses valores são trocados de posição (o menor valor vem para a posição mais baixa). E assim é feito com todos os elementos do Fecho. Como cada elemento é um par ordenado, compara-se apenas o primeiro elemento, porém quando é feita a troca de posição, deve-se copiar ambos os valores.

for (i = 0; i < (QtdElemFecho-1); i++)

{

k = i;

for (j = (i+1); j < QtdElemFecho; j++)

{

if(fechoS[j][0] < fechoS[k][0])

k = j;

}

if (i != k)

{

aux = fechoS[i][0];

aux2 = fechoS[i][1];

fechoS[i][0] = fechoS[k][0];

fechoS[i][1] = fechoS[k][1];

fechoS[k][0] = aux;

fechoS[k][1] = aux2;

}

}

Para o segundo loop de ordenação, utiliza-se o mesmo raciocínio, porém procura-se apenas os pares ordenados que que tenham o mesmo valor como primeiro elemento (já ordenado) e aplica-se a comparação ao segundo elemento. Isso é feito apenas alterando a estrutura condicional que compara os termos, da seguinte forma:

if((fechoS[j][0] == fechoS[k][0])&&(fechoS[j][1] < fechoS[k][1]))

Assim ele só entra no loop de ordenação para os pares que possuem valores iguais no primeiro elemento.

* + 1. **Apresentação dos dados de saída**

Conforme enunciado, caso a relação gerada seja simétrica, o software encerra apenas com a exibição do valor “1” na tela, caso não seja simétrica, o software apresento valor “0” em uma linha e mostra o fecho simétrico a partir da próxima linha, sendo um par ordenado por linha.

Nesse caso, os valores “1” ou “0”, significando se a matriz é simétrica ou não, é alcançado a partir do conteúdo da variável “simétrico”, que é atualizada na rotina de verificação da Relação sobre essa propriedade.

Como o código foi feito na linguagem C, um valor verdadeiro é representado pelo valor “1” e o valor falso é representado pelo valor “0”. Portanto basta simplesmente testar essa variável e imprimir os resultados de acordo com o resultado do teste.

if (!simetrico) //caso a relação não seja simétrica

{

printf("\n%d\n",simetrico); //imprime o valor 0 (simétrico=false=0)

for (i=0;i<QtdElemFecho;i++) //imprime o Fecho simétrico gerado

printf("%d %d\n",fechoS[i][0],fechoS[i][1]);

}

else //caso seja simétrica

printf("\n%d\n",simetrico); //imprime o valor 1 e encerra (simétrico=true=1)

* 1. **ESTRUTURAS DE DADOS**

Para o desenvolvimento desse código, foi utilizado apenas o conceito de matrizes como estruturas de dados. Somente com essa estrutura foi possível atender todos os requisitos solicitados para a saída do problema.

1. **REFERÊNCIAS BIBLIOGRÁFICAS**

<https://pt.wikipedia.org/wiki/Combinat%C3%B3ria#Arranjo_com_repeti.C3.A7.C3.A3o>, acessado em 13/05/2016

<https://pt.wikipedia.org/wiki/Rela%C3%A7%C3%A3o_bin%C3%A1ria>, acessado em 17/05/2016

<http://www.deinf.ufma.br/~vidal/mdl/12-relacoesBinarias.pdf>, acessado em 17/05/2016

<https://pt.wikipedia.org/wiki/Selection_sort#C>, acessado em 14/05/2016

**DEPARTAMENTO DE TEORIA DE INFORMÁTICA.**

(P. Blauth Menezes – site [blauth@inf.ufrgs.br](mailto:blauth@inf.ufrgs.br)).

**INTRODUÇÃO A MATEMÁTICA DISCRETA.**

(Professor Antonio Alfredo Ferreira).

**RELAÇÕES E FUNÇÕES.**

(Kate Revoredo – www.wordpcfconvert.com).