Politécnico de Leiria

Escola Superior de Tecnologia e Gestão

Matemática Discreta - Componente PL

EI (D)

Ano letivo 2023/2024 - 2.º Sem. Ficha prática 5

Relações

Definição de relação

Uma **relação** R, definida de um conjunto A para um conjunto B, é um subconjunto de $A \times B$ (produto cartesiano entre A e B). Podemos definir a relação R da seguinte forma:

$$R = \{(a,b) \in A \times B : a \text{ está } R-\text{relacionado com } b\}.$$

Exemplo 1 *Se $A = \{d,e,f\}$ e $B = \{0,1,2\}$ então podemos definir uma relação R do conjunto A para o conjunto B da seguinte forma:*

$$R = \{(d,0),(d,1),(f,2),(e,1)\}.$$

Exemplo 2 *Podemos definir uma relação S, de $\mathbb{N}$ para $\mathbb{N}$, da seguinte forma:*

$$S = \left\{(a,b) \in \mathbb{N} \times \mathbb{N} : \frac{a}{b} = 2\right\}.$$

O par $(6,3)$ e o par $(10,5)$ pertencem à relação S.

Relação inversa

Para toda a relação R, definida de um conjunto A para um conjunto B, é possível definir a **relação inversa** de R, que se denota por R^{-1}. Deste modo, se

$$R = \{(a,b) \in A \times B : a \text{ é } R-\text{relacionado com } b\}$$

temos

$$R^{-1} = \{(b,a) \in B \times A : a \text{ é } R-\text{relacionado com } b\}.$$

Formas de representar uma relação

Existem diferentes **formas de representar uma relação**. Podemos representar uma relação R de um conjunto A para um conjunto B através de:

1. Diagrama de setas;
2. Matriz da relação;
3. Grafo orientado (caso $A = B$, com A um conjunto finito).

Exemplo 3 *Se $A = \{d,e,f\}$ e $B = \{0,1,2\}$ então podemos representar a relação $R = \{(d,0),(d,1),(f,2),(e,1)\}$, definida de A para B, através da seguinte matriz:*

$$M_R = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Composição de relações

Sejam A, B e C conjuntos, seja R uma relação de A para B e seja S uma relação de B para C. A relação $R \circ S$, **relação R em composição com S,** é uma relação do conjunto A para o conjunto C definida por

$$R \circ S = \{(a, c) \in A \times C : \text{a está } R - relacionado \text{ com } b \text{ e } b \text{ está } S - relacionado \text{ com } c\}.$$

Nota: $R^2 = R \circ R$, ..., $R^n = R^{n-1} \circ R$.

Podemos determinar a **matriz da relação** $R \circ S$ (**R em composição com S**) através da multiplicação da matriz da relação R, M_R, com a matriz da relação S, M_S. Primeiro calculamos $M_R \times M_S$; de seguida, nesta matriz resultante, denotada por $M_{R \circ S}$, substituímos todos os elementos não nulos por 1, obtendo assim a matriz da relação $R \circ S$.

Tipos de relações

Seja R uma relação definida sobre um conjunto A, ou seja, $R = \{(a, b) \in A \times A : aRb\}$. A relação R diz-se:

1. **reflexiva** se $\forall a \in A$, aRa;
2. **simétrica** se, $\forall a, b \in A$, $aRb \Longrightarrow bRa$;
3. **antissimétrica** se, $\forall a, b \in A$, $aRb \wedge bRa \Longrightarrow a = b$;
4. **transitiva** se $\forall a, b, c \in A$, $aRb \wedge bRc \Longrightarrow aRc$.

Relação de equivalência e relação de ordem parcial

Uma relação R, definida em A, diz-se uma **relação de equivalência** se e só se R é reflexiva, simétrica e transitiva.

Uma relação R, definida em A, diz-se uma **relação de ordem parcial** se e só se R é reflexiva, antissimétrica e transitiva. Neste caso, o par (A, R) diz-se um conjunto parcialmente ordenado.

Propriedades de fecho

Consideremos uma relação R definida sobre um conjunto A.

1. O **fecho reflexivo de** R é dado por $reflexivo(R) = R \cup \{(a, a) : a \in A\}$;
2. O **fecho simétrico de** R é dado por $simetrico(R) = R \cup R^{-1}$;
3. O **fecho transitivo de** R é dado por $transitivo(R) = \bigcup_{i=1}^{m} R^i = R \cup R^2 \cup ... \cup R^m$, onde m é o número de elementos do conjunto A.

Classes de equivalência e Conjunto quociente

Seja R uma relação de equivalência sobre um conjunto não vazio A. Para cada $a \in A$, define-se a **classe de equivalência de a em A** como o conjunto

$$[a]_R = \{x \in A : aRx\}.$$

Qualquer elemento $x \in [a]_R$ diz-se um **representante desta classe de equivalência.**

O conjunto de todas as classes de equivalência de R designa-se por **conjunto quociente de** A por R e é representado por A/R. Ou seja,

$$A/R = \{[a]_R : a \in A\}.$$

De salientar que o conjunto quociente A/R é uma partição do conjunto A.

Matrizes

Tal como referido na Ficha 2, em Python existem várias formas de criar vetores e matrizes e operar com os mesmos. Nesta UC, para definir e manipular vetores e matrizes, será usado o *package Numpy*.

```
In [1]: import numpy as np
```

Os vetores e matrizes podem ser definidos através do objeto *array*, que pertence a este *package*. Enquanto que, para definir um vetor pode ser utilizado um array unidimensional, para definir uma matriz deverá ser usado um array bidimensional:

Definição de vetor / matriz	
`A=np.array([[`x_{00},x_{01},...,x_{0n}`],...,[`x_{m0},x_{m1},...,x_{mn}`]]`	Define a matriz `A` $(m+1)\times(n+1)$
`A.size`	Devolve o n.º de elementos da matriz `A`
`A.shape`	Devolve a dimensão da matriz `A` (n.º de linhas × n.º de colunas)

Exemplos:

```
In [3]: B=np.array([[1,2,3],[4,5,6]])
        print(B)

[[1 2 3]
 [4 5 6]]
```

```
In [4]: B=np.array([[1,2,3],[4,5,6]])
        B.shape

Out[4]: (2, 3)
```

```
In [5]: B=np.array([[1,2,3],[4,5,6]])
        B.size

Out[5]: 6
```

- Na tabela seguinte são apresentados comandos que permitem obter informação sobre os elementos de uma matriz, bem como manipular o formato da mesma:

Características de uma matriz	
`A[a,b]`	Se `a` e `b` são inteiros não negativos, apresenta o elemento que se encontra na linha `a`+1 e coluna `b`+1
`A[a,:]`	Apresenta todos os elementos da linha `a`+1
`A[:,b]`	Apresenta todos os elementos da coluna `b`+1
`A[:,:]`	Apresenta toda a matriz `A`
`A[i:j,:]`	Apresenta os elementos da matriz `A` que se encontram da `(i+1)`−ésima inha até à `j`−ésima linha
`np.delete(A,i,axis)`	Se `axis`= 0, devolve uma cópia da matriz `A` após eliminação da linha `i`+1 Se `axis`= 1, devolve uma cópia da matriz `A` após eliminação da coluna `i`+1
`np.array_equal(A,B)` ou `(A==B).all()`	Verifica se as matrizes `A` e `B` são iguais (devolve valor lógico)

- Na tabela seguinte são apresentados comandos que permitem a criação de certos tipos de matrizes específicas:

Matrizes específicas	
`np.eye(m)`	Matriz `m`×`m` com 1's na diagonal e 0's nas restantes posições - Matriz identidade
`np.zeros((m,n))`	Matriz `m`×`n` com todos os elementos iguais a 0 - Matriz nula
`np.ones((m,n))`	Matriz `m`×`n` com todos os elementos iguais a 1

Exemplos:

```
In [6]: B

Out[6]: array([[1, 2, 3],
               [4, 5, 6]])
```

```
In [7]: print(B)

        [[1 2 3]
         [4 5 6]]
```

```
In [8]: B[-2,-1]

Out[8]: 3
```

```
In [9]: B[1,1]

Out[9]: 5
```

```
In [10]: B[0:2,1:3]

Out[10]: array([[2, 3],
                [5, 6]])
```

```
In [11]: B[1,:]

Out[11]: array([4, 5, 6])
```

```
In [12]: B1=np.concatenate((B,np.array([[0,0,0]])),axis=0)
         print("B=",B)
         print("----------")
         print("B1=",B1)

         B= [[1 2 3]
          [4 5 6]]
         ----------
         B1= [[1 2 3]
          [4 5 6]
          [0 0 0]]
```

In [14]:
```
B3=np.delete(B,2,1)
print("B=",B)
print("----------")
print("B3=",B3)
```

```
B= [[1 2 3]
 [4 5 6]]
----------
B3= [[1 2]
 [4 5]]
```

In [15]:
```
B4=np.delete(B,1,1)
print("B=",B)
print("----------")
print("B4=",B4)
```

```
B= [[1 2 3]
 [4 5 6]]
----------
B4= [[1 3]
 [4 6]]
```

In [16]:
```
np.zeros((2,3))
```

Out[16]:
```
array([[0., 0., 0.],
       [0., 0., 0.]])
```

In [17]:
```
np.ones((2,2))
```

Out[17]:
```
array([[1., 1.],
       [1., 1.]])
```

In [18]:
```
Id3=np.eye(3)
print(Id3)
```

```
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```

- Na tabela seguinte são apresentados comandos que permitem efetuar operações com matrizes:

Funções Matriciais	
(`A` e `B` matrizes **com dimensões adequadas**)	
`A.T`	Transposta da matriz `A`
`A@B`	Multiplicação matricial entre `A` e `B`
`np.trace(A)`	Soma dos elementos da diagonal principal da matriz `A`
`np.linalg.matrix_power(A,k)`	Potência de índice `k` da matriz `A`

- **Exemplos:**

```
In [22]: A=np.array([[1,2],[3,4]])
         print('A=',A ,'\n----\n','A^2=\n',A @ A)
```

```
A= [[1 2]
 [3 4]]
----
 A^2=
 [[ 7 10]
 [15 22]]
```

```
In [23]: A = np.array([[1 , 2 , 3],[-1,-2,-3]])
         print('A=',A ,'\n----\n','A^T=\n',A.T)
```

```
A= [[ 1  2  3]
 [-1 -2 -3]]
----
 A^T=
 [[ 1 -1]
 [ 2 -2]
 [ 3 -3]]
```

Exercícios propostos

1. Seja $A = \{1, 2, 3, 4\}$ e R a relacão em A definida por:

$$R = \{(1, 1), (1, 3), (2, 1), (2, 3), (3, 1), (3, 2), (4, 4)\}.$$

   (a) Indique a matriz que representa a relação R, M_R.

   (b) Determine a matriz da relação R^{-1}.

   (c) Determine a matriz da relação R^2 e a matriz da relação R^3.

   (d) Indique, justificando, se R é uma relação reflexiva, simétrica, antissimétrica e/ou transitiva.

   (e) Determine:

      i. o fecho reflexivo de R;

      ii. o fecho simétrico de R;

      iii. o fecho transitivo de R.

   (f) Determine a menor relação de equivalência definida em A que contém o conjunto R.

   (g) Seja S a relação de equivalência determinada na alínea anterior. Determine as classes de equivalência de S, o conjunto quociente A/S e a partição do conjunto A determinada através de A/S.

2. Sejam R e S relações definidas num conjunto $D = \{a, b, c\}$ cujas matrizes de relação M_R e M_S são dadas por:

$$M_R = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \quad \text{e} \quad M_S = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

   (a) Determine a matriz associada à relação $R \circ S$.

   (b) Indique, justificando, se c é $R \circ S-$relacionado com a.

3. Assumindo que é possível realizar a composição $R \circ S$, construa uma função com o nome relcomp(M_R,M_S), em que M_R é a matriz de uma relação R e M_S é a matriz de uma relação S, que devolva a matriz da relação $R \circ S$.

4. A partir da matriz M_R de uma relação R, construa uma função:

   (a) com o nome is_reflexiva(M_R) que verifique se a relação R é uma relação reflexiva;

   (b) com o nome is_simetrica(M_R) que verifique se a relação R é uma relação simétrica;

   (c) com o nome is_transitiva(M_R) que verifique se a relação R é uma relação transitiva.

5. A partir da função is_reflexiva(M_R), implementada na alínea (a) do exercício anterior, construa uma função com o nome fecho_reflexivo(M_R) que, caso a relação R não seja reflexiva, devolva a matriz do fecho reflexivo de R.

6. Construa uma função com o nome equival(M_R), em que M_R é a matriz de uma relação R, que verifique se a relação R é ou não uma relação de equivalência e, em caso negativo, devolva a matriz da menor relação de equivalência que contém a relação R.

7. Construa uma função com o nome orparcial(M_R), em que M_R é a matriz de uma relação R, que verifique se a relação R é ou não uma relação de ordem parcial.

Nota: Os exercícios 8 e 9 deverão ser resolvidos por análise dos algoritmos (sem recorrer ao Python); deverá justificar convenientemente as suas respostas.

8. Dada a matriz M de uma relação R, indique qual(is) das seguintes funções permite(m) determinar se R é reflexiva:

   - Função 1:

```
def reflexiva(M):
    [a,b]=M.shape
    for i in range(0,a):
        if M[i,i]==1:
            print("A relação é reflexiva")
        else:
            print("A relação não é reflexiva")
```

   - Função 2:

```
def reflexiva(M):
    [a,b]=M.shape
    if np.trace(M)==a:
        print("A relação é reflexiva")
    else:
        print("A relação não é reflexiva")
```

   - Função 3:

```
def reflexiva(M):
    [a,b]=M.shape
    if np.array_equal(np.diag(M),np.diag(np.eye(a))) :
        print("A relação é reflexiva")
    else:
        print("A relação não é reflexiva")
```

9. Dada a matriz M de uma relação R, indique qual(is) das seguintes funções permite(m) obter a matriz do fecho simétrico da relação R :

- Função 1:

```
def fecho_sim(M):
    [a,b]=M.shape
    for i in range(0,a):
        for j in range(0,b):
            if M[i,j]==1:
                M[j,i]=1
    return M
```

- Função 2:

```
def fecho_sim(M):
    [a,b]=M.shape
    S=M+M.T
    return S
```

- Função 3:

```
def fecho_sim(M):
    [a,b]=M.shape
    S=M+M.T
    S[S!=0]=1
    return S
```