第 3 章

二元关系

二元关系思维导图如图 3.1 所示。

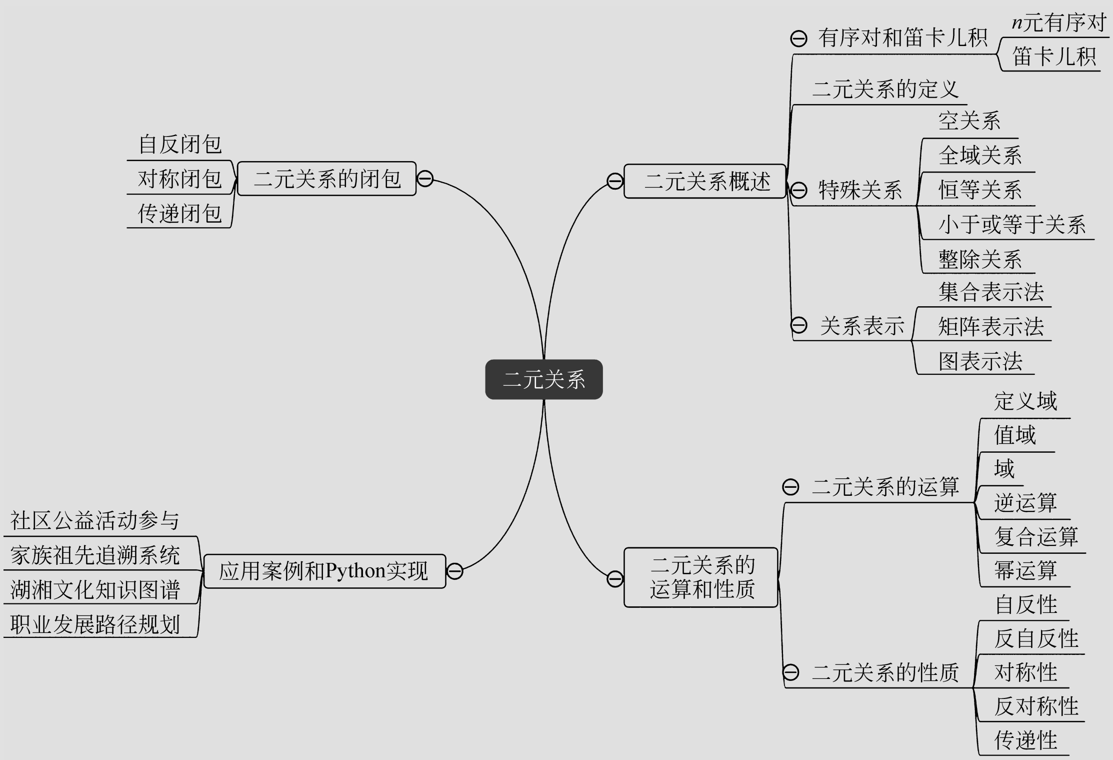

图 3.1　二元关系思维导图

二元关系作为集合论中的基本概念，其历史可以追溯到 19 世纪末乔治·康托尔创立集合论的时期。乔治·康托尔的集合论不仅为数学提供了一种全新的思维方式，还奠定了后续逻辑学和计算机科学等领域的基础。恩斯特·施罗德是早期对逻辑学和集合论做出重要贡献的数学家，尤其在关系理论的形式化方面，他的工作对后来的数学家产生了深远的

影响。

20 世纪初，随着数学家对二元关系及其性质的系统性研究，二元关系逐渐成为数学理论的一个重要分支。伯特兰·罗素和阿尔弗雷德·诺斯·怀特海德在他们的里程碑式著作《数学原理》中，对关系和函数进行了深入探讨，为逻辑学和数学的基础理论提供了坚实的基石。

进入 20 世纪中叶，随着数理逻辑和计算机科学的迅猛发展，二元关系的概念得到了进一步的扩展和应用。埃德加·F.科德在 20 世纪 70 年代提出的数据库理论中的关系模型便是基于二元关系的概念，这一理论不仅革新了信息存储和检索的方式，也极大地推动了信息技术的发展。

本章将深入探讨集合论中的二元关系领域，包括二元关系概述、二元关系的运算和性质、二元关系的闭包，以及它们的相关应用。通过本章内容的学习，读者可以掌握面向应用问题求解的二元关系相关的数学理论、工具和方法。

微课视频

3.1 二元关系概述

3.1.1 有序对和笛卡儿积

播客音频

在许多实际应用中，集合中元素之间的顺序关系至关重要。例如，在坐标系统中，点的位置是通过有序数对来确定的。

定义 3.1.1 **有序对**用来表示两个元素及其顺序关系，其由两个元素组成，记为 $\langle a,b\rangle$，a 是第一个元素，b 是第二个元素，定义为

$$\langle a,b\rangle=\{\{a\},\{a,b\}\}$$

除非 $a=b$，否则 $\langle a,b\rangle$ 和 $\langle b,a\rangle$ 为不同的有序对。另外，对于任何 a,b,c,d，$\langle a,b\rangle$ 与 $\langle c,d\rangle$ 相等当且仅当 $a=c$ 且 $b=d$。

定义 3.1.2 ***n* 元有序对**，也称 n 元组，是一个将 n 个元素按特定顺序组织起来的对象。对于任意正整数 n，当 $n>2$ 时，n 元有序对 $\langle a_1,a_2,\cdots,a_n\rangle$ 为 $\langle\langle\langle a_1,a_2\rangle,a_3\rangle,\cdots\rangle,a_n\rangle$。此定义允许人们构造任意长度的有序序列，每个新元素都扩展了前一个有序对。

定义 3.1.3 **笛卡儿积**提供了一种方法来构建来自两个或多个集合的所有可能有序对的集合。对于集合 A 和 B，笛卡儿积 $A\times B$ 是所有可能的有序对 $\langle a,b\rangle$ 的集合，其中 $a\in A$，$b\in B$，定义为

$$A\times B=\{\langle a,b\rangle \mid a\in A \text{ 且 } b\in B\}$$

集合 A,B,C 的笛卡儿积：

$$A\times B\times C=\{\langle a,b,c\rangle \mid a\in A, b\in B \text{ 且 } c\in C\}$$

n 个集合 $A_1,A_2,\cdots,A_n$ 的笛卡儿积：

$$A_1\times A_2\times\cdots\times A_n=\{\langle a_1,a_2,\cdots,a_n\rangle \mid a_1\in A_1,a_2\in A_2,\cdots,a_n\in A_n\}$$

式中，a_i 为集合 A_i 中的元素，且 $\langle a_1,a_2,\cdots,a_n\rangle$ 是 n 元有序对。

设想某人与其朋友计划在周末观看电影。他们面临的第一个决策是选择电影，第二个决策是选择观影时间。在该场景中，有序对可以被想象为一个特殊的小盒子，其不仅包含电影票和时间段这两个元素，而且非常注重它们放入的顺序。若第一个放入的是电影票，第二个是时间段，那么有序对〈电影票，时间段〉就明确地给出了决策的顺序。进一步地，若手头有一份电影列表和一份时间段列表，笛卡儿积就相当于将这两个列表结合起来，形成一个包含所有可能组合的超级大表。该表展示了每部电影与每个时间段的配对，提供了所有可能的观影选择。

例 3.1.1 (1)设 $A=\{1,2\}$ 和 $B=\{a,b\}$，计算 $A\times B$。

(2) 设 $A=\{1,2\}$ 和 $B=\{\}$，计算 $A\times B$。

(3) 设 $A=\{x,y\}$，计算 $A\times A$。

(4) 设 $A=\{1\}$，$B=\{x,y\}$，$C=\{\text{true},\text{false}\}$，计算 $A\times B\times C$。

解：(1) $A\times B=\{\langle 1,a\rangle,\langle 1,b\rangle,\langle 2,a\rangle,\langle 2,b\rangle\}$。

(2) $A\times B=\{\}$。

(3) $A\times A=\{\langle x,x\rangle,\langle x,y\rangle,\langle y,x\rangle,\langle y,y\rangle\}$。

(4) $A\times B\times C=\{\langle 1,x,\text{true}\rangle,\langle 1,x,\text{false}\rangle,\langle 1,y,\text{true}\rangle,\langle 1,y,\text{false}\rangle\}$。

Python 代码实现示例 3.1.1

```
from itertools import product
A1 = {1, 2}
B1 = {'a', 'b'}
A2 = {1, 2}
B2 = set()                                             #空集
A3 = {'x', 'y'}
A4 = {1}
B4 = {'x', 'y'}
C4 = {True, False}
cartesian_product_1 = set(product(A1, B1))             #计算集合的笛卡儿积
cartesian_product_2 = set(product(A2, B2))
cartesian_product_3 = set(product(A3, A3))
cartesian_product_4 = set(product(A4, B4, C4))
print(cartesian_product_1)
print(cartesian_product_2)
print(cartesian_product_3)
print(cartesian_product_4)
```

例 3.1.2 设集合 $A=\{x,y\}$，求 $A\times P(A)$的集合。

解：(1) 计算幂集 $P(A)$：对于 $A=\{x,y\}$，有 $P(A)=\{\varnothing,\{x\},\{y\},\{x,y\}\}$。

(2) 计算 $A\times P(A)$：$A\times P(A)=\{\langle x,\varnothing\rangle,\langle x,\{x\}\rangle,\langle x,\{y\}\rangle,\langle x,\{x,y\}\rangle,\langle y,\varnothing\rangle,\langle y,\{x\}\rangle,\langle y,\{y\}\rangle,\langle y,\{x,y\}\rangle\}$。

Python 代码实现示例 3.1.2

```
from itertools import product, chain
A = {'x', 'y'}
P_A = set()                                            #计算幂集 P(A)
for i in range(len(A) + 1):
    for subset in product(A, repeat=i):
        P_A.add(frozenset(subset))
cartesian_product = set(product(A, P_A))               #计算笛卡儿积 A×P(A)
print(cartesian_product)
```

例 3.1.3 高中生在升入大学之前通常要选择自己的专业方向，这是人生的一个重要决策。各种专业对于社会的发展都有其独特的价值和重要性。考虑两个集合：

$$C=\{\text{经济学},\text{工程学},\text{医学}\},D=\{\text{公办大学},\text{民办大学}\}$$

解：学生可能选择的学校和专业组合如下：$C\times D=\{\langle$经济学，公办大学$\rangle$，$\langle$经济学，民办大学$\rangle$，$\langle$工程学，公办大学$\rangle$，$\langle$工程学，民办大学$\rangle$，$\langle$医学，公办大学$\rangle$，$\langle$医学，民办大学$\rangle\}$。

笛卡儿积的基本性质如下。

(1) 一般情况下，有序对$\langle a,b\rangle$不同于$\langle b,a\rangle$，则 $A\times B\neq B\times A$。

(2) 一般情况下，$\langle\langle a,b\rangle,c\rangle$和$\langle a,\langle b,c\rangle\rangle$是不同类型的有序对，则$(A\times B)\times C\neq A\times(B\times C)$。

(3) 并集和交集满足分配律，即 $A\times(B\cup C)=(A\times B)\cup(A\times C)$和 $A\times(B\cap C)=(A\times B)\cap(A\times C)$。

(4) 任何集合与空集的笛卡儿积都是空集，即 $A\times\varnothing=\varnothing$。

(5) 若集合 A 和 B 是有限集，那么 $A\times B$ 的基数为$|A|\times|B|$。

(6) 若 $A\subseteq C$ 且 $B\subseteq D$，则 $A\times B\subseteq C\times D$。

例 3.1.4 证明 $A\times(B\cup C)=(A\times B)\cup(A\times C)$。

证明：(1) 设 $\langle a,x\rangle$ 是集合 $A\times(B\cup C)$ 中的任意元素。根据笛卡儿积的定义，有 $a\in A$ 且 $x\in B\cup C$。由于 $x\in B\cup C$，根据并集的定义，$x\in B$ 或 $x\in C$。若 $x\in B$，则 $\langle a,x\rangle\in A\times B$；若 $x\in C$，则 $\langle a,x\rangle\in A\times C$。在任何情况下，$\langle a,x\rangle\in(A\times B)\cup(A\times C)$，即 $A\times(B\cup C)\subseteq(A\times B)\cup(A\times C)$。

(2) 设 $\langle a,x\rangle$ 是集合 $(A\times B)\cup(A\times C)$ 中的任意元素。根据并集的定义，$\langle a,x\rangle\in A\times B$ 或 $\langle a,x\rangle\in A\times C$。若 $\langle a,x\rangle\in A\times B$，则 $a\in A$ 且 $x\in B$；类似地，若 $\langle a,x\rangle\in A\times C$，则 $a\in A$ 且 $x\in C$。在任何情况下，$a\in A$ 且 $x\in B$ 或 $x\in C$，即 $x\in B\cup C$。因此，$\langle a,x\rangle\in A\times(B\cup C)$，即 $(A\times B)\cup(A\times C)\subseteq A\times(B\cup C)$。

综上所述，$A\times(B\cup C)=(A\times B)\cup(A\times C)$。

例 3.1.5 若 $A\subseteq C$ 且 $B\subseteq D$，则 $A\times B\subseteq C\times D$。

证明：从 $A\times B$ 中选取任意的元素 $\langle a,b\rangle$。根据笛卡儿积的定义，$a\in A$ 且 $b\in B$。由于 $A\subseteq C$，可得 $a\in C$。同理，由于 $B\subseteq D$，可得 $b\in D$。因此，$a\in C$ 且 $b\in D$，即 $\langle a,b\rangle\in C\times D$。由于 $\langle a,b\rangle$ 是任意从 $A\times B$ 中选取的，并且对于所有这样的 $\langle a,b\rangle$，都证明了 $\langle a,b\rangle\in C\times D$，因此 $A\times B\subseteq C\times D$。

应用示例

有序对和笛卡儿积在定义二维或三维空间中的点时有实际应用。

(1) 在计算机图形学和几何建模中，笛卡儿积可以生成二维空间中的所有点。设 X 和 Y 分别表示 x 轴和 y 轴的坐标范围，通过计算笛卡儿积 $X\times Y$，可以得到坐标平面中的所有点，用于绘制图形或进行几何运算。设图像的宽度为 W 和高度为 H，笛卡儿积 $\{0,1,\cdots,W-1\}\times\{0,1,\cdots,H-1\}$ 表示图像中的所有像素点。

(2) 在三维空间中，点由有序三元组 $\langle x,y,z\rangle$ 表示。通过 3 个集合的笛卡儿积，可以生成三维空间中的所有点。设 $X=\{1,2\}$，$Y=\{3,4\}$，$Z=\{5,6\}$，笛卡儿积 $X\times Y\times Z=\{\langle 1,3,5\rangle,\langle 1,3,6\rangle,\langle 1,4,5\rangle,\langle 1,4,6\rangle,\langle 2,3,5\rangle,\langle 2,3,6\rangle,\langle 2,4,5\rangle,\langle 2,4,6\rangle\}$，其中的有序三元组表示三维空间中的点。

3.1.2 二元关系的定义

二元关系描述了两个集合元素之间的关系。在不同的领域，二元关系以不同的形式出现，可以帮助人们理解和表达元素间的各种联系。

定义 3.1.4 设有集合 A 和 B，**二元关系** R 包含所有来自 A 和 B 的有序对，记为 $R\subseteq A\times B$，也称 R 为 A 到 B 上的关系。

(1) 地理位置集合中，若位置 a 和位置 b 是相邻的，则有序对 $\langle a,b\rangle$ 属于关系 R。

(2) 人的集合中，若 a 是 b 的父亲，则有序对 $\langle a,b\rangle$ 属于父子关系 R。

(3) 设社交网络用户集合为 $A=\{$张三，李四，王五$\}$，定义关系 R 表示“是好友”。若张三和李四是好友，则有序对 $\langle$张三，李四$\rangle\in R$。由于此关系是在 A 内部定义的，因此可将“是好友”的关系表示为 $R\subseteq A\times A$。

(4) 设张三和李四是好友,李四和王五也是好友,但张三和王五不是好友,有 $R=\{\langle$张三,李四$\rangle,\langle$李四,王五$\rangle\}$,此关系 R 是有向的。

例 3.1.6　设集合 $A=\{1,2,3,4\}$和 $B=\{a,b,c\}$,关系 $R=\{\langle 1,a\rangle,\langle 2,b\rangle,\langle 3,c\rangle,\langle 4,a\rangle,\langle 1,b\rangle\}$是否为从 A 到 B 上的关系?

解:在关系 R 中,每个有序对的形式为$\langle a,b\rangle$,其中 $a\in A$ 且 $b\in B$。因此,R 是从 A 到 B 上的关系。

Python 代码实现示例 3.1.3

```
A1 = {1, 2, 3, 4}
B1 = {'a', 'b', 'c'}
R1 = {(1, 'a'), (2, 'b'), (3, 'c'), (4, 'a'), (1, 'b')}
is_relation_1 = all(a in A1 and b in B1 for a, b in R1)    #检查关系是否为从 A 到 B
print(is_relation_1)
```

例 3.1.7　设集合 $A=\{m,n,o\}$和 $B=\{1,2,3,4\}$,关系 $R=\{\langle m,1\rangle,\langle n,2\rangle,\langle o,3\rangle,\langle m,5\rangle\}$是否为从 A 到 B 上的关系?

解:在关系 R 中,有序对$\langle m,5\rangle$的第二个元素 5 不在集合 B 中。因此,R 不是从 A 到 B 上的关系。

Python 代码实现示例 3.1.4

```
A2 = {'m', 'n', 'o'}
B2 = {1, 2, 3, 4}
R2 = {('m', 1), ('n', 2), ('o', 3), ('m', 5)}
is_relation_2 = all(a in A2 and b in B2 for a, b in R2)    #检查关系是否为从 A 到 B
print(is_relation_2)
```

定理 3.1.1　若集合 A 有 n 个元素,则 A 上的二元关系的总数是 2^{n^2}。

证明:设集合 A 有 n 个元素,记为 $A=\{a_1,a_2,\cdots,a_n\}$。集合 A 上的笛卡儿积 $A\times A$ 包含所有可能的有序对$\langle a_i,a_j\rangle$,其中 $a_i,a_j\in A$ 且 $i,j=1,2,\cdots,n$。根据笛卡儿积的定义,$A\times A$ 包含 $n\times n=n^2$ 个元素。

任何 A 上的二元关系 R 都是 $A\times A$ 的子集,即 $R\subseteq A\times A$。对于 $A\times A$ 中的每个元素,要么属于关系 R,要么不属于关系 R。因此,对于 $A\times A$ 中的 n^2 个元素,有 2^{n^2} 种不同的选择方式。因此,定理 3.1.1 成立。

定义 3.1.5　设 n 个集合 $A_1,A_2,\cdots,A_n$,**n 元关系** R 是笛卡儿积 $A_1\times A_2\times\cdots\times A_n$ 的子集:

$$R\subseteq A_1\times A_2\times\cdots\times A_n$$

式中,$\langle a_1,a_2,\cdots,a_n\rangle$是 n 元有序对,$a_i\in A_i$ 且 $i=1,2,\cdots,n$。

例 3.1.8　设集合 $A=\{a,b\}$,$B=\{1,2\}$,$C=\{X,Y\}$,定义在 $A\times B\times C$ 上的三元关系 R:

$$R=\{\langle a,1,X\rangle,\langle b,2,Y\rangle\}$$

表示 R 包含$\langle a,1,X\rangle$和$\langle b,2,Y\rangle$两个三元组。

Python 代码实现示例 3.1.5

```
A = {'a', 'b'}                                           #定义 3 个集合 A, B, C
B = {1, 2}
C = {'X', 'Y'}
R_full = {(a, b, c) for a in A for b in B for c in C}   #所有可能的三元关系 R
```

应 用 示 例

二元关系为建模社交网络提供了基础，并且在分析用户行为、传播信息、推荐系统等方面有重要作用。

(1) 二元关系可以用于分析好友关系图，检测社交网络中的社区结构，识别紧密联系的用户群体。

(2) 基于用户的关注关系，二元关系可以用于推荐潜在的关注对象或感兴趣的内容。

(3) 二元关系可以用于建立信息传播模型，模拟信息在关注关系图中的传播过程，预测信息扩散范围。

(4) 二元关系可以用于分析阻止关系，监控和管理不当行为，保护用户隐私和安全。

3.1.3 特殊关系

在二元关系中，空关系、全域关系、恒等关系、小于或等于关系以及整除关系等为一些特殊类型的关系。

定义 3.1.6 **空关系**指在集合 A 和 B 上不包含任何有序对的关系。若 R 是一个空关系，则 $R \subseteq A \times B$ 且 $R = \varnothing$。

例如，若 $A=\{1,2\}$ 和 $B=\{a,b\}$，则 $A \times B$ 上的空关系是 $\varnothing$。

定义 3.1.7 **全域关系**指包含所有从集合 A 和 B 的可能有序对的关系。若 E 是一个全域关系，则 $E = A \times B$。

例如，若 $A=\{1,2\}$ 和 $B=\{a,b\}$，则 $A \times B$ 上的全域关系是 $\{\langle 1,a\rangle,\langle 1,b\rangle,\langle 2,a\rangle,\langle 2,b\rangle\}$。

定义 3.1.8 **恒等关系**指仅包含形如 $\langle a,a\rangle$ 的有序对的关系。若 I_A 是集合 A 上的恒等关系，则 $I_A=\{\langle a,a\rangle \mid a \in A\}$。

例如，若 $A=\{1,2\}$，则 A 上的恒等关系是 $\{\langle 1,1\rangle,\langle 2,2\rangle\}$。

定义 3.1.9 **小于或等于关系**指在任何有序集合 R 上，满足任意 $a,b \in R$，$\langle a,b\rangle \in R$ 当且仅当 $a \leqslant b$，记为 $R=\{\langle a,b\rangle \in R \times R \mid a \leqslant b\}$。

例如，实数 2 和 3，由于 $2 \leqslant 3$，因此有序对 $\langle 2,3\rangle$ 存在于小于或等于关系中。

定义 3.1.10 **整除关系**指在整数集 $\mathbf{Z}$ 上，满足任意 $a,b \in \mathbf{Z}$ 且 $b \neq 0$，$\langle a,b\rangle \in R$ 当且仅当 b 整除 a，即存在整数 k 使得 $a=kb$，记为 $R=\{\langle a,b\rangle \in \mathbf{Z} \times \mathbf{Z} \mid b \neq 0$ 且 $a=kb, k \in \mathbf{Z}\}$。

例如，整数 4 和 2，由于 4 可以被 2 整除，因此有序对 $\langle 4,2\rangle$ 存在于整除关系中。

例 3.1.9 求下列各集合之间的关系。

(1) 集合 $E=\{a,b,c\}$ 和 $F=\{1,2\}$，求 $E \times F$ 上的全域关系。

(2) 集合 $G=\{0,1,2,3,4,5\}$,求 G 上的恒等关系。

(3) 集合 $H=\{1,2,3,4,5\}$,求 H 上的小于或等于关系。

(4) 集合 $I=\{2,3,4,6\}$,求 I 上的整除关系。

解:(1) 全域关系是$\{\langle a,1\rangle,\langle a,2\rangle,\langle b,1\rangle,\langle b,2\rangle,\langle c,1\rangle,\langle c,2\rangle\}$。

(2) 恒等关系是$\{\langle 0,0\rangle,\langle 1,1\rangle,\langle 2,2\rangle,\langle 3,3\rangle,\langle 4,4\rangle,\langle 5,5\rangle\}$。

(3) 小于或等于关系包括$\{\langle 1,1\rangle,\langle 1,2\rangle,\cdots,\langle 1,5\rangle,\langle 2,2\rangle,\cdots,\langle 2,5\rangle,\cdots,\langle 5,5\rangle\}$。

(4) 整除关系包括$\{\langle 2,2\rangle,\langle 2,4\rangle,\langle 2,6\rangle,\langle 3,3\rangle,\langle 3,6\rangle,\langle 4,4\rangle,\langle 6,6\rangle\}$。

Python 代码实现示例 3.1.6

```
from itertools import product
#定义集合
E = {'a', 'b', 'c'}
F = {1, 2}
G = {0, 1, 2, 3, 4, 5}
H = {1, 2, 3, 4, 5}
I = {2, 3, 4, 6}
universal_relation_EF = set(product(E, F))        #E×F上的全域关系
identity_relation_G = {(g, g) for g in G}         #G上的恒等关系
leq_relation_H = {(h1, h2) for h1 in H for h2 in H if h1 <= h2} #H上的小于或等于关系
div_relation_I = {(i1, i2) for i1 in I for i2 in I if i2 % i1 == 0} #I上的整除关系
print(universal_relation_EF)
print(identity_relation_G)
print(leq_relation_H)
print(div_relation_I)
```

例 3.1.10　设正在设计一个数据库,其中包含两个字段:学生姓名和他们选择的课程。若数据库中有 3 名学生和 2 门课程,列出全域关系。

解:设学生为{张三,李四,王五},课程为{离散数学,人工智能},全域关系为{〈张三,离散数学〉,〈张三,人工智能〉,〈李四,离散数学〉,〈李四,人工智能〉,〈王五,离散数学〉,〈王五,人工智能〉}。

例 3.1.11　在一个社交网络中,每个用户都有一个唯一的用户 ID。给出社交网络上用户 ID 之间的恒等关系示例,并解释其应用意义。

解:设用户 ID 集合为{001,002,003},恒等关系是{〈001,001〉,〈002,002〉,〈003,003〉}。这种关系用于表示用户自己与自己的关系,如自己的个人资料页面。

例 3.1.12　一家公司的员工按资历等级排列,资历等级用 1 到 5 表示。若只考虑资历等级,列出员工之间的小于或等于关系,并解释其在晋升政策上的意义。

解:小于或等于关系包括所有形如$\langle x,y\rangle$的组合,其中 x 和 y 是从 1 到 5 的数字,且 $x\leqslant y$。该关系表明一个低等级的员工可以晋升到高等级,但不可逆转。

例 3.1.13　一家制造公司拥有一条生产线,其生产周期是特定数量的天数。设一组生产周期的天数$\{2,3,4,6,8,12\}$,并假定公司计划在某个 12 天的生产窗口内安排生产任务。列出所有与 12 天生产窗口相容的生产周期。

解:整除关系中包含的有序对为$\{\langle 2,12\rangle,\langle 3,12\rangle,\langle 4,12\rangle,\langle 6,12\rangle,\langle 12,12\rangle\}$。这表示生产周期为 2 天、3 天、4 天、6 天和 12 天的任务可以在 12 天的生产窗口内完整地安排,不会出现未完成的生产周期。

应用示例

在计算机人工智能领域中，特殊关系如空关系、全域关系、恒等关系以及小于或等于关系等，有助于对各种算法和系统进行建模、推理和优化。

（1）空关系可以表示未发现的模式或无效的数据点。

（2）全域关系表示所有特征和标签之间的完全关联，这是特征选择和降维的重要前提。

（3）数据一致性是模型训练的基础。恒等关系可以用于检查数据集中的重复记录和数据完整性。

（4）在决策树算法中，小于或等于关系用于节点的分裂条件。通过比较特征值与分裂阈值，以决定数据点的分支方向。

3.1.4　关系表示

出于理解、分析和应用二元关系的需求，可以使用多种不同方式对二元关系进行表示，每种表示方法有其独特的优点和适用场景。关系表示方法主要有集合表示法、矩阵表示法和图表示法。前面已经给出多个集合表示法示例，本节重点介绍矩阵表示法和图表示法。

定义 3.1.11　设集合 $A=\{a_1,a_2,\cdots,a_m\}$ 和 $B=\{b_1,b_2,\cdots,b_n\}$，且二元关系 $R\subseteq A\times B$。该关系可以通过一个 $m\times n$ 的矩阵 $\boldsymbol{M}$ 来表示，其中矩阵行对应集合 A 的元素，矩阵列对应集合 B 的元素。矩阵 $\boldsymbol{M}$ 的第 i 行和第 j 列的元素 M_{ij} 定义如下。

（1）若 $\langle a_i,b_j\rangle\in R$，则 $M_{ij}=1$。

（2）若 $\langle a_i,b_j\rangle\notin R$，则 $M_{ij}=0$。

例 3.1.14　设集合 $A=\{1,2,3,4,5\}$，用矩阵法表示二元关系 $R=\{\langle 1,2\rangle,\langle 2,3\rangle,\langle 3,4\rangle,\langle 4,5\rangle\}$。

解：

$$\boldsymbol{M}_R=\begin{bmatrix}0&1&0&0&0\\0&0&1&0&0\\0&0&0&1&0\\0&0&0&0&1\\0&0&0&0&0\end{bmatrix}$$

Python 代码实现示例 3.1.7

```
A = [1, 2, 3, 4, 5]                        #集合 A 和二元关系 R
R = [(1, 2), (2, 3), (3, 4), (4, 5)]
n = len(A)                                 #获取集合 A 的基数
M_R = [0] * n for _ in range(n)            #初始化关系矩阵 MR 为 n×n 的零矩阵
for i in range(n):                         #遍历每个元素
```

```
    for j in range(n):
        if (A[i], A[j]) in R:        #检查元素(A[i], A[j])是否在关系 R 中
            M_R[i][j] = 1            #若在关系 R 中,则将对应的矩阵元素设置为 1
for row in M_R:                      #输出矩阵 M_R
    print(row)
```

例 3.1.15　一个项目有 4 项任务 $C=\{T_1,T_2,T_3,T_4\}$和 4 名员工 $D=\{E_1,E_2,E_3,E_4\}$。E_1 负责 T_1 和 T_2,E_2 负责 T_2 和 T_3,E_3 和 E_4 负责 T_4。使用矩阵表示员工和任务之间的分配关系。

解：

$$\boldsymbol{M}_R=\begin{bmatrix}1&1&0&0\\0&1&1&0\\0&0&0&1\\0&0&0&1\end{bmatrix}$$

式中,$M_{ij}=1$ 表示员工 E_i 负责任务 T_j。

Python 代码实现示例 3.1.8

```
C = ['T1', 'T2', 'T3', 'T4']          #任务集合 C
D = ['E1', 'E2', 'E3', 'E4']          #员工集合 D
task_allocation = {                   #任务分配情况
    'E1': ['T1', 'T2'],
    'E2': ['T2', 'T3'],
    'E3': ['T4'],
    'E4': ['T4']}
allocation_matrix = [[0] * len(C) for _ in range(len(D))]
                                      #初始化分配矩阵为全 0 矩阵
for i, employee in enumerate(D):
    for j, task in enumerate(C):
        if task in task_allocation[employee]:
            allocation_matrix[i][j] = 1
for row in allocation_matrix:
    print(row)
```

矩阵表示法在处理二元关系时具有显著的优势,尤其是在数据密集型和计算密集型的应用之中。

(1) 矩阵通过其行列结构,提供了一种直观的方式来表示二元关系。集合 A 的每个元素对应矩阵的一行,集合 B 的每个元素对应矩阵的一列,而矩阵中的每个元素表示关系中对应行和列元素之间的关系。

(2) 矩阵代数为二元关系提供了丰富的操作,如矩阵加法、乘法和转置等。这些操作可以方便地应用于关系的并、交、补、差等运算。

(3) 在计算机科学中,利用矩阵表示二元关系可以加速算法的执行,如在图算法、网络分析和数据库查询优化中。矩阵运算的优化和并行化是现代计算技术中的重要研究方向。

(4) 矩阵表示法在数学的多个分支,特别是线性代数中,有着坚实的理论基础。这些理论为分析和处理二元关系提供了强大的工具,如特征值分析、奇异值分解等。

(5) 矩阵表示法不仅适用于二元关系,还可以扩展到更复杂的情况,如多元关系、加权

关系等。加权关系可以通过在矩阵中使用实数值来表示元素之间的关系强度，这在社交网络分析、推荐系统等领域中具有应用价值。

定义 3.1.12 设集合 $A=\{a_1,a_2,\cdots,a_m\}$和 $B=\{b_1,b_2,\cdots,b_n\}$，且存在 $R\subseteq A\times B$。该关系可以通过图 $G=\langle V,E\rangle$来表示，其中 V 是图的顶点集合，E 是图的边集合。

（1）图 G 的顶点集 V 是集合 A 和 B 的并集，即 $V=A\cup B$，每个元素在图中只出现一次。

（2）图 G 的边集 E 由所有满足关系 R 的有序对$\langle a,b\rangle$表示。对于每个有序对$\langle a,b\rangle\in R$，图中存在一个从顶点 a 指向顶点 b 的边，这些边可以是有向的或无向的。

二元关系表示为图的步骤如下。

（1）创建顶点：对于集合 A 和 B 中的每个元素，图中有相应的顶点。若二元关系定义在同一集合上，$R\subseteq A\times A$，则集合 A 中每个元素对应一个顶点。

（2）绘制边：对于关系 $R\subseteq A\times B$ 中的每个有序对$\langle a,b\rangle$，在顶点 a 和 b 之间绘制一条边。若$\langle a,b\rangle\in R$ 且$\langle b,a\rangle\in R$，则边是无向的，表示 a 和 b 之间的相互关系；若$\langle a,b\rangle\in R$ 且$\langle b,a\rangle\notin R$，则边是有向的，表示从 a 到 b 的单向关系。

例 3.1.16 设 $A=\{1,2,3,4\}$，$R=\{\langle 1,2\rangle,\langle 2,2\rangle,\langle 2,3\rangle,\langle 1,4\rangle\}$，用图表示 R。

解：如图 3.2 所示，其包含 4 个顶点，分别标记为 1，2，3，4。从顶点 1 到顶点 4 和顶点 2 分别有一条有向边，顶点 2 有自环，以及从顶点 2 到顶点 3 有一条有向边。

Python 代码实现示例 3.1.9

```
import networkx as nx
import matplotlib.pyplot as plt
A = {1, 2, 3, 4}                 #集合 A 和关系 R
R = {(1, 2), (2, 2), (2, 3), (1, 4)}
G = nx.DiGraph()                 #创建有向图
G.add_edges_from(R)              #添加关系边
pos = nx.spring_layout(G)        #绘制图形
nx.draw_networkx(G, pos, with_labels=True, arrows=True)
plt.show()
```

例 3.1.17 设 $A=\{1,2,3,4,5\}$，$R=\{\langle 1,2\rangle,\langle 1,3\rangle,\langle 1,4\rangle,\langle 1,5\rangle,\langle 2,3\rangle,\langle 2,4\rangle,\langle 2,5\rangle,\langle 3,4\rangle,\langle 3,5\rangle\}$，用图表示 R。

解：如图 3.3 所示，其包含 5 个顶点，分别标记为 1，2，3，4，5。顶点 1 到顶点 2，3，4，5 分别有一条有向边，顶点 2 到顶点 3，4，5 分别有一条有向边，顶点 3 到顶点 4 和 5 分别有一条有向边。

Python 代码实现示例 3.1.10

```
import networkx as nx
import matplotlib.pyplot as plt
A = {1, 2, 3, 4, 5}
R = {(1, 2), (1, 3), (1, 4), (1, 5), (2, 3), (2, 4), (2, 5), (3, 4), (3, 5)}
G = nx.DiGraph()
G.add_edges_from(R)
pos = nx.spring_layout(G)
nx.draw_networkx(G, pos, with_labels=True, arrows=True)
plt.show()
```

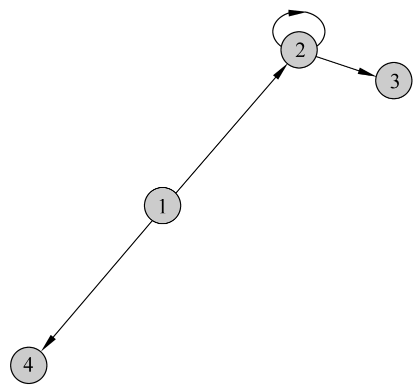

图 3.2　例 3.1.16 的关系

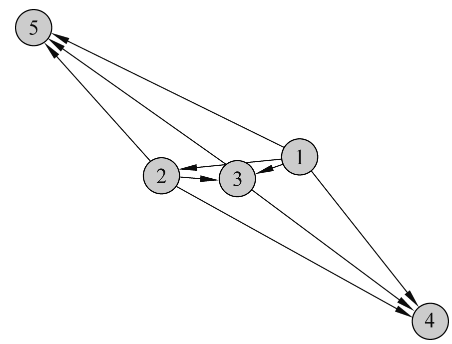

图 3.3　例 3.1.17 的关系

例 3.1.18　设观众集合 $A=\{$张三,李四,王五$\}$,电影集合 $B=\{$流浪地球,我和我的祖国,长津湖$\}$,评分集合 $C=\{1,2,3,4,5\}$,$R=\{\langle$张三,流浪地球,$4\rangle$,$\langle$张三,我和我的祖国,$5\rangle$,$\langle$李四,长津湖,$4\rangle$,$\langle$王五,流浪地球,$5\rangle$,$\langle$王五,长津湖,$4\rangle\}$,用图表示观众和电影评分关系 R。

解:如图 3.4 所示,其用来表示观众对电影的评分情况,每个观众对不同电影的评分通过边的连接和边上的数值展示出来。通过图 3.4,人们可以直观地看出每个观众对哪些电影进行了评分,以及具体的评分数值。这有助于分析观众的喜好、电影的受欢迎程度等。

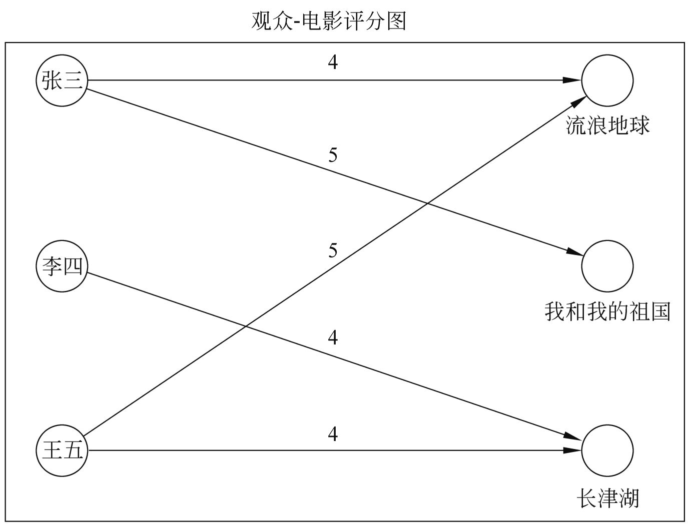

图 3.4　例 3.1.18 的关系

图是一种对二元关系进行可视化和分析的强大工具,具有多种优势和广泛应用。

(1) 图表示法提供了一种直观的方式来理解二元关系。节点代表实体,如人、物体或概念;而边则展示了这些实体之间的相互作用或联系。这不仅可以帮助人们快速把握数据的总体结构,还能揭示大型数据集中不易察觉的模式和趋势。

(2) 图表示法在社交网络分析、推荐系统、生物信息学、交通网络规划、通信网络架构等多个领域有应用。在这些领域中,图不仅用于展示数据,还用于模拟和预测实体之间的复杂

交互。

(3) 随着图形数据模型在数据库系统中的日益普及,使用图来表示二元关系变得更加方便和高效。许多现代数据库系统,如图形数据库,提供了对图结构的原生支持,包括高效的图查询语言和算法。

应用示例

计算机网络的结构可以用矩阵和图表示,这种表示方法在网络设计和流量分析中非常有用。

(1) 矩阵表示有助于网络优化问题的求解。使用矩阵表示二元关系,可以方便地实现图算法。通过矩阵运算,人们可以优化网络结构和性能。其在网络路由、负载均衡和故障检测中有实际应用。

(2) 图表示可以直观地展示网络中各节点之间的路径和连接性。通过图,人们可以轻松识别网络中的关键节点和瓶颈,直观地分析数据流在网络中的路径,识别可能的拥塞点,设计更高效的路由算法。其也可以描述网络的拓扑结构,包括星形、环形、树形和网状等不同类型的网络结构。这有助于理解网络的物理和逻辑布局,优化网络性能。

3.1.5 小结

本节讨论了有序对和笛卡儿积,二元关系的定义,空关系、全域关系、恒等关系等特殊关系,以及矩阵和图的关系表示等内容。有序对是表示两个元素之间顺序关系的基本单位;笛卡儿积是两个集合所有有序对的集合,是构建二元关系的基础。二元关系定义了两个集合之间的关系,可以用来描述元素之间的各种关联。特殊关系如空关系、全域关系和恒等关系提供了对关系的不同视角。矩阵表示法使用 0 和 1 的矩阵来表示关系,可以方便地进行关系的运算和分析;图表示法通过节点和边的结构直观地展示关系,广泛应用于计算机科学、网络分析等领域。

习题

1. 有序对$\langle a,b\rangle$与$\langle b,a\rangle$的关系是(　　)。

A. 相等　　B. 不相等　　C. 取决于 a 和 b　　D. 无法确定

2. 集合 $A=\{1,2\}$上的空关系是(　　)。

A. $\{\langle 1,1\rangle,\langle 2,2\rangle\}$　　B. $\{\langle 1,2\rangle,\langle 2,1\rangle\}$

C. $\{\}$　　D. $A\times A$

3. 集合 $A=\{a,b\}$上的恒等关系是(　　)。

A. $\{\langle a,a\rangle,\langle b,b\rangle\}$　　B. $\{\langle a,b\rangle,\langle b,a\rangle\}$

C. $\{\}$　　D. $A\times A$

4. 设集合 $A=\{1,2\}$和关系矩阵$\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$,对应的关系是(　　)。

A. $\{\langle 1,1\rangle,\langle 2,2\rangle\}$

B. $\{\langle 1,2\rangle,\langle 2,1\rangle\}$

C. $\{\langle 1,1\rangle,\langle 1,2\rangle,\langle 2,1\rangle,\langle 2,2\rangle\}$

D. $\{\}$

5. 计算集合 $E=\{1,2,3\}$ 和 $F=\{x,y\}$ 的笛卡儿积的元素个数，并列出其所有元素。

6. 计算集合 $E=\{a,b,c\}$ 上的全域关系的元素个数，并列出所有的关系对。

7. 给定关系 $H=\{\langle a,a\rangle,\langle a,b\rangle,\langle b,c\rangle\}$，构造其对应的关系矩阵。

8. 设集合 $A=\{1,2,3\}$ 和关系矩阵 $\begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 0\\ 1 & 0 & 1\end{bmatrix}$，列出对应的关系对。

向人工智能大模型探索提问

1. 问理论知识：笛卡儿积在数学中有哪些应用场景？

2. 问实践应用：在特征工程中，如何利用笛卡儿积生成组合特征？这种方法如何影响模型的表现？

3. 问工程项目：在分布式系统中，如何利用二元关系优化任务调度和资源管理？

4. 问学术科研：二元关系在网络科学中的最新研究方向是什么？有哪些重要的应用前景？

5. 问习题解答：习题 6 和 7 的正确求解过程是什么？

6. 问代码解释：Python 代码实现示例 3.1.7 和示例 3.1.8 的详细解释和分析是什么？

3.2　二元关系的运算和性质

微课视频

播客音频

3.2.1　二元关系的运算

二元关系继承了集合的并集、交集和差集等基本运算。例如，集合 $A=\{1,2,3\}$，其上定义关系 $R=\{\langle 1,2\rangle,\langle 2,3\rangle,\langle 3,3\rangle\}$ 和 $S=\{\langle 2,1\rangle,\langle 3,3\rangle\}$，则 $R\cup S=\{\langle 1,2\rangle,\langle 2,3\rangle,\langle 2,1\rangle,\langle 3,3\rangle\}$，$R\cap S=\{\langle 3,3\rangle\}$，$R-S=\{\langle 1,2\rangle,\langle 2,3\rangle\}$。然而，二元关系还有一些特有的运算和性质，这些在处理关系时非常重要。

定义 3.2.1　对于集合 A 和 B 上的二元关系 $R\subseteq A\times B$，**定义域** $\mathrm{dom}(R)$ 是集合 A 中所有出现在 R 中的第一个元素的集合，记为 $\mathrm{dom}(R)=\{a\in A\mid \exists b\in B,\langle a,b\rangle\in R\}$。

定义 3.2.2　对于关系 $R\subseteq A\times B$，**值域** $\mathrm{ran}(R)$ 是集合 B 中所有出现在 R 中的第二个元素的集合，记为 $\mathrm{ran}(R)=\{b\in B\mid \exists a\in A,\langle a,b\rangle\in R\}$。

定义 3.2.3　对于集合 A 和 B 上的二元关系 R，**域** $\mathrm{fld}(R)$ 包含所有在关系 R 中作为第一个或第二个元素出现的元素，记为 $\mathrm{fld}(R)=\mathrm{dom}(R)\cup\mathrm{ran}(R)$。

例 3.2.1　设 $A=\{1,2,3\}$ 和 $B=\{3,4,5\}$，在 A 和 B 上关系 $R=\{\langle 1,3\rangle,\langle 2,3\rangle,\langle 3,4\rangle\}$。求 R 的定义域、值域和域。

解：$\mathrm{dom}(R)=\{1,2,3\}$，$\mathrm{ran}(R)=\{3,4\}$，$\mathrm{fld}(R)=\{1,2,3,4\}$。

例 3.2.2　设 $A=\{a,b,c,d,e\}$，求在其上的关系 R_1 和 R_2 的定义域、值域和域。其中，

$R_1=\{\langle a,b\rangle,\langle b,c\rangle,\langle c,d\rangle\}$，$R_2=\{\langle a,b\rangle,\langle b,a\rangle,\langle c,d\rangle,\langle d,e\rangle\}$。

解：$\mathrm{dom}(R_1)=\{a,b,c\}$，$\mathrm{ran}(R_1)=\{b,c,d\}$，$\mathrm{fld}(R_1)=\{a,b,c,d\}$。

$\mathrm{dom}(R_2)=\{a,b,c,d\}$，$\mathrm{ran}(R_2)=\{a,b,d,e\}$，$\mathrm{fld}(R_2)=\{a,b,c,d,e\}$。

Python 代码实现示例 3.2.1

```
A ={'a', 'b', 'c', 'd', 'e'}
R1 ={('a', 'b'), ('b', 'c'), ('c', 'd')}
R2 ={('a', 'b'), ('b', 'a'), ('c', 'd'), ('d', 'e')}
def domain(relation):
    return {pair[0] for pair in relation}
def range_set(relation):
    return {pair[1] for pair in relation}
def field(relation):
    return domain(relation).union(range_set(relation))
dom_R1 =domain(R1)       #计算 R1
ran_R1 =range_set(R1)
fld_R1 =field(R1)
dom_R2 =domain(R2)       #计算 R2
ran_R2 =range_set(R2)
fld_R2 =field(R2)
print("R1:")
print("定义域 dom(R1):", dom_R1)
print("值域 ran(R1):", ran_R1)
print("域 fld(R1):", fld_R1)
print("\nR2:")
print("定义域 dom(R2):", dom_R2)
print("值域 ran(R2):", ran_R2)
print("域 fld(R2):", fld_R2)
```

定理 3.2.1　集合 A 到 B 的二元关系 R 和 S，其交、并和差运算如下。

(1) $\mathrm{dom}(R\cap S)\subseteq\mathrm{dom}(R)\cap\mathrm{dom}(S)$。

(2) $\mathrm{ran}(R\cap S)\subseteq\mathrm{ran}(R)\cap\mathrm{ran}(S)$。

(3) $\mathrm{dom}(R\cup S)=\mathrm{dom}(R)\cup\mathrm{dom}(S)$。

(4) $\mathrm{ran}(R\cup S)=\mathrm{ran}(R)\cup\mathrm{ran}(S)$。

(5) $\mathrm{dom}(R-S)\subseteq\mathrm{dom}(R)$。

(6) $\mathrm{ran}(R-S)\subseteq\mathrm{ran}(R)$。

证明：设 $a\in\mathrm{dom}(R\cap S)$。根据定义，$\exists b\in B$ 使得 $\langle a,b\rangle\in R\cap S$。由交集性质，$\langle a,b\rangle\in R\cap S$，因此 $a\in\mathrm{dom}(R)$ 且 $a\in\mathrm{dom}(S)$，即 $a\in\mathrm{dom}(R)\cap\mathrm{dom}(S)$。

若 a 在 R 和 S 的定义域中，则 $\exists b_1\in B$ 使得 $\langle a,b_1\rangle\in R$，$\exists b_2\in B$ 使得 $\langle a,b_2\rangle\in S$。若 $b_1=b_2$，则 $\langle a,b_1\rangle\in R\cap S$，从而 $a\in\mathrm{dom}(R\cap S)$。因为 $R\cap S$ 要求 b 必须相同，若 $b_1\neq b_2$，则不能保证 $a\in\mathrm{dom}(R\cap S)$。

综上所述，$\mathrm{dom}(R\cap S)\subseteq\mathrm{dom}(R)\cap\mathrm{dom}(S)$。

其他几条定理读者可自行证明。

定义 3.2.4　若 R 是二元关系，$R\subseteq A\times B$，R 的**逆关系** R^{-1} 为

$$R^{-1}=\{\langle b,a\rangle \mid \langle a,b\rangle\in R\}$$

例 3.2.3　设集合 $A=\{1,2,3\}$ 和关系 $R=\{\langle 1,2\rangle,\langle 2,3\rangle,\langle 3,1\rangle\}$，求 R 的逆关系 R^{-1}，并给出 R 和 R^{-1} 的矩阵和图表示。

解：$R^{-1}=\{\langle 2,1\rangle,\langle 3,2\rangle,\langle 1,3\rangle\}$，$R$ 和 R^{-1} 的矩阵表示分别如下：

$$\boldsymbol{M}_R=\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}$$

$$\boldsymbol{M}_{R^{-1}}=\begin{bmatrix}0&0&1\\1&0&0\\0&1&0\end{bmatrix}$$

图 3.5 展示了如何通过逆关系操作得到一个新的二元关系。图 3.5 中，R^{-1}是图 R 的逆关系，即将图 R 中的每条有向边的方向反转。

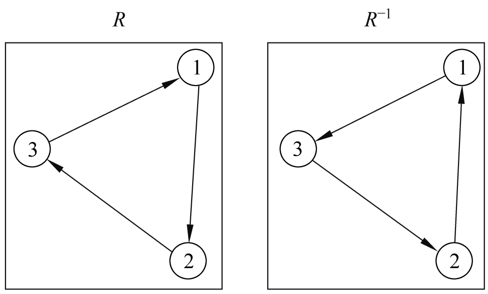

图 3.5　例 3.2.3 的关系

Python 代码实现示例 3.2.2

```
def inverse_relation(relation):
    return {(b, a) for (a, b) in relation}
def relation_to_matrix(relation, elements):
    matrix = [[0 for _ in elements] for _ in elements]
    element_indices = {element: i for i, element in enumerate(elements)}
    for (a, b) in relation:
        i, j = element_indices[a], element_indices[b]
        matrix[i][j] = 1
    return matrix
A = {1, 2, 3}                                              #集合 A 和关系 R
R = {(1, 2), (2, 3), (3, 1)}
R_inverse = inverse_relation(R)                            #计算关系 R⁻¹
matrix_R = relation_to_matrix(R, A)                        #关系 R 的矩阵表示
matrix_R_inverse = relation_to_matrix(R_inverse, A)        #关系 R⁻¹的矩阵表示
print(R_inverse), print(matrix_R), print(matrix_R_inverse)
```

定理 3.2.2　对于任意二元关系 R 和 S，有：

（1）$(R^{-1})^{-1}=R$。

（2）$\mathrm{dom}(R^{-1})=\mathrm{ran}(R)$，$\mathrm{ran}(R^{-1})=\mathrm{dom}(R)$。

（3）$(R\cap S)^{-1}=R^{-1}\cap S^{-1}$。

（4）$(R\cup S)^{-1}=R^{-1}\cup S^{-1}$。

（5）$(R-S)^{-1}=R^{-1}-S^{-1}$。

证明：（1）对于关系 R 中的所有$\langle a,b\rangle$，根据逆运算的定义，有$\langle b,a\rangle\in R^{-1}$。再次应用逆运算，可得$\langle a,b\rangle\in(R^{-1})^{-1}$，$(R^{-1})^{-1}$包含 R 中的所有元素对，则$(R^{-1})^{-1}=R$。

(2) 对于 R 中的任意 $\langle a,b\rangle$，b 是 R 值域中的元素，a 是 R 定义域中的元素。在 R^{-1} 中，对应的有序对是 $\langle b,a\rangle$，这说明 R 的值域变成了 R^{-1} 的定义域，R 的定义域变成了 R^{-1} 的值域，则 $\mathrm{dom}(R^{-1})=\mathrm{ran}(R)$，$\mathrm{ran}(R^{-1})=\mathrm{dom}(R)$。

(3) 若 $\langle a,b\rangle\in(R\cap S)^{-1}$，则 $\langle b,a\rangle\in R\cap S$，这说明 $\langle b,a\rangle$ 同时属于 R 和 S。因此，$\langle a,b\rangle\in R^{-1}$ 且 $\langle a,b\rangle\in S^{-1}$，即 $\langle a,b\rangle\in R^{-1}\cap S^{-1}$，$(R\cap S)^{-1}=R^{-1}\cap S^{-1}$。

(4) 类似于交的逆，若 $\langle a,b\rangle\in(R\cup S)^{-1}$，则 $\langle b,a\rangle\in R\cup S$，这说明 $\langle b,a\rangle$ 属于 R 或 S。因此，$\langle a,b\rangle$ 属于 R^{-1} 或 S^{-1}，即 $\langle a,b\rangle\in R^{-1}\cup S^{-1}$，也即 $(R\cup S)^{-1}=R^{-1}\cup S^{-1}$。

(5) 若 $\langle a,b\rangle\in(R-S)^{-1}$，则 $\langle b,a\rangle\in R-S$，那么 $\langle b,a\rangle\in R$ 且 $\langle b,a\rangle\notin S$，即 $\langle a,b\rangle\in R^{-1}$ 且 $\langle a,b\rangle\notin S^{-1}$，$\langle a,b\rangle\in R^{-1}-S^{-1}$，$(R-S)^{-1}=R^{-1}-S^{-1}$。

定义 3.2.5 设有二元关系 R 和 S。其中，R 定义在集合 A 和 B 上，$R\subseteq A\times B$；而 S 定义在集合 B 和 C 上，$S\subseteq B\times C$。R 和 S 的**复合关系** $R\circ S$ 为

$$R\circ S=\{\langle a,c\rangle\in A\times C\mid \exists b\in B,\langle a,b\rangle\in R\text{ 且}\langle b,c\rangle\in S\}$$

例 3.2.4 设 $A=\{1,2,3,4\}$，定义在其上的关系 $R=\{\langle 1,2\rangle,\langle 2,3\rangle,\langle 3,4\rangle\}$，$S=\{\langle 2,1\rangle,\langle 3,2\rangle,\langle 4,3\rangle\}$，求复合关系 $R\circ S$。

解：对 R 中的每个元素对 $\langle x,z\rangle$，检查 S 中是否存在形式为 $\langle z,y\rangle$ 的元素对。若存在，则将 $\langle x,y\rangle$ 加入复合关系 $R\circ S$ 中。R 中有 $\langle 1,2\rangle$，S 中有 $\langle 2,1\rangle$，因此 $R\circ S$ 包含 $\langle 1,1\rangle$；R 中有 $\langle 2,3\rangle$，S 中有 $\langle 3,2\rangle$，因此 $R\circ S$ 包含 $\langle 2,2\rangle$；R 中有 $\langle 3,4\rangle$，S 中有 $\langle 4,3\rangle$，因此 $R\circ S$ 包含 $\langle 3,3\rangle$。所以，复合关系 $R\circ S=\{\langle 1,1\rangle,\langle 2,2\rangle,\langle 3,3\rangle\}$。关系 R、S 和 $R\circ S$ 的矩阵分别记为 $\boldsymbol{M}_R$、$\boldsymbol{M}_S$ 和 $\boldsymbol{M}_{R\circ S}$，表示如下：

$$\boldsymbol{M}_R=\begin{bmatrix}0&1&0&0\\0&0&1&0\\0&0&0&1\\0&0&0&0\end{bmatrix}$$

$$\boldsymbol{M}_S=\begin{bmatrix}0&0&0&0\\1&0&0&0\\0&1&0&0\\0&0&1&0\end{bmatrix}$$

$$\boldsymbol{M}_{R\circ S}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&0\end{bmatrix}$$

Python 代码实现示例 3.2.3

```
R = {(1, 2), (2, 3), (3, 4)}            #关系 R 和 S
S = {(2, 1), (3, 2), (4, 3)}
def composite_relation(R, S):           #计算复合关系 R∘S
    RS = set()
    for (a, b) in R:
        for (c, d) in S:
            if b == c:
```

```
            RS.add((a, d))
    return RS
RS = composite_relation(R, S)        #计算复合关系 R∘S
print("复合关系 RS:", RS)
```

图 3.6 展示了如何通过复合操作得到一个新的二元关系，复合关系 $R\circ S$ 通过在关系 S 的输出基础上应用关系 R 来获得。例如，关系 S 中有节点 3 到 2 的边，关系 R 中有节点 2 到 3 的边，在复合关系 $R\circ S$ 中表现为 3 的自环。

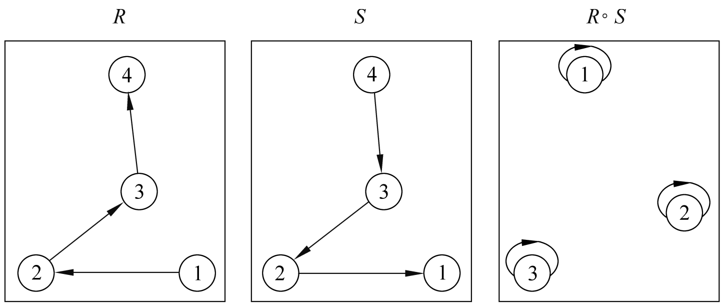

图 3.6　例 3.2.4 的关系

定义 3.2.6　矩阵复合运算是一种用矩阵表示和操作二元关系的方法。设有两个二元关系 R 和 S，分别表示为矩阵 $\boldsymbol{M}_R$ 和 $\boldsymbol{M}_S$，其中 $\boldsymbol{M}_R$ 是 $m\times n$ 矩阵，$\boldsymbol{M}_S$ 是 $n\times p$ 矩阵，$\boldsymbol{M}_R\circ\boldsymbol{M}_S$ 可以定义为一个新的 $m\times p$ 矩阵 $\boldsymbol{M}_C$，其中每个元素 $\boldsymbol{M}_C[i,j]$ 为：

$$\boldsymbol{M}_C[i,j]=\bigvee_{k=1}^{n}(\boldsymbol{M}_R[i,k]\wedge\boldsymbol{M}_S[k,j])$$

式中，$\vee$ 为逻辑或运算；$\wedge$ 为逻辑与运算（详见 5.1.3 节）。

$\boldsymbol{M}_C[i,j]$ 的值为 1 当且仅当存在至少一个元素 k 使得 $\boldsymbol{M}_R[i,k]$ 和 $\boldsymbol{M}_S[k,j]$ 都为 1。

例 3.2.5　设关系矩阵 $\boldsymbol{R}=\begin{bmatrix}1&0&1\\0&1&0\\1&1&0\end{bmatrix}$ 和 $\boldsymbol{S}=\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}$，求 $\boldsymbol{R}\circ\boldsymbol{S}$。

解：

$$\boldsymbol{R}\circ\boldsymbol{S}=\begin{bmatrix}1&1&0\\0&0&1\\0&1&1\end{bmatrix}$$

Python 代码实现示例 3.2.4

```
import numpy as np
R = np.array([[1, 0, 1],                             #矩阵 R 和 S
              [0, 1, 0],
              [1, 1, 0]])
S = np.array([[0, 1, 0],
              [0, 0, 1],
              [1, 0, 0]])
R_circ_S = (np.dot(R, S) > 0).astype(int)        #计算矩阵复合 R∘S
print(R_circ_S)
```

定理 3.2.3 空关系$\varnothing$与任何关系 R 的复合都为空，即$\varnothing \circ R=\varnothing$和 $R \circ \varnothing=\varnothing$。

证明：若存在某个元素 c，使得$\langle a, c\rangle \in \varnothing$且$\langle c, b\rangle \in R$，则$\langle a, b\rangle \in \varnothing \circ R$。但由于$\varnothing$不包含任何元素，因此不可能存在这样的 c 满足$\langle a, c\rangle \in \varnothing$。因此，$\varnothing \circ R$ 不包含任何元素，即$\varnothing \circ R=\varnothing$。同理可证 $R \circ \varnothing=\varnothing$。

定理 3.2.4 设集合 A 上有二元关系 R, S, T。

(1) $(R \circ S) \circ T=R \circ(S \circ T)$。

(2) $(R \circ S)^{-1}=S^{-1} \circ R^{-1}$。

(3) 若 I_A 为集合 A 上的恒等关系，即 $I_A \circ R=R$ 和 $R \circ I_A=R$。

证明：(1) 设 a, c 是集合 A 上的任意元素，且$\langle a, c\rangle \in(R \circ S) \circ T$。根据复合关系的定义，存在一个元素 b，使得$\langle a, b\rangle \in R \circ S$ 且$\langle b, c\rangle \in T$。由于$\langle a, b\rangle \in R \circ S$，存在一个元素 d，使得$\langle a, d\rangle \in R$ 且$\langle d, b\rangle \in S$。因此，$\langle d, c\rangle \in S \circ T$，进而得到$\langle a, c\rangle \in R \circ(S \circ T)$。由于 a, c 是任意的，因此$(R \circ S) \circ T=R \circ(S \circ T)$。

(2) 设 a, c 是集合 A 上的任意元素，且$\langle a, c\rangle \in(R \circ S)^{-1}$，这说明$\langle c, a\rangle \in R \circ S$。因此，存在元素 b，使得$\langle c, b\rangle \in R$ 且$\langle b, a\rangle \in S$，因此$\langle b, c\rangle \in R^{-1}$且$\langle a, b\rangle \in S^{-1}$，从而$\langle a, c\rangle \in S^{-1} \circ R^{-1}$。由于 a, c 是任意的，因此$(R \circ S)^{-1}=S^{-1} \circ R^{-1}$。

(3) 设 a, b 是集合 A 上的任意元素，$\langle a, b\rangle \in I_A \circ R$。根据恒等关系的定义，存在元素 c，使得$\langle a, c\rangle \in I_A$ 且$\langle c, b\rangle \in R$。由于$\langle a, c\rangle \in I_A$，可得 $a=c$，因此$\langle a, b\rangle \in R$。由于 a, b 是任意的，因此 $I_A \circ R=R$。同理可证 $R \circ I_A=R$。

定理 3.2.5 设集合 A 上有二元关系 R, S, T。

(1) $R \circ(S \cup T)=(R \circ S) \cup(R \circ T)$。

(2) $(S \cup T) \circ R=(S \circ R) \cup(T \circ R)$。

(3) $R \circ(S \cap T) \subseteq(R \circ S) \cap(R \circ T)$。

(4) $(S \cap T) \circ R \subseteq(S \circ R) \cap(T \circ R)$。

证明：(1) ① 设$\langle a, b\rangle \in R \circ(S \cup T)$，这说明存在某个元素 c，使得$\langle a, c\rangle \in R$ 且$\langle c, b\rangle \in S \cup T$。因为$\langle c, b\rangle \in S \cup T$，所以$\langle c, b\rangle \in S$ 或$\langle c, b\rangle \in T$。若$\langle c, b\rangle \in S$，则$\langle a, b\rangle \in R \circ S$；若$\langle c, b\rangle \in T$，则$\langle a, b\rangle \in R \circ T$。因此，$\langle a, b\rangle \in(R \circ S) \cup(R \circ T)$。

② 假设$\langle a, b\rangle \in(R \circ S) \cup(R \circ T)$，则$\langle a, b\rangle \in R \circ S$ 或$\langle a, b\rangle \in R \circ T$。若$\langle a, b\rangle \in R \circ S$，则存在某个元素 c，使得$\langle a, c\rangle \in R$ 且$\langle c, b\rangle \in S$。类似地，若$\langle a, b\rangle \in R \circ T$，则存在某个元素 c，使得$\langle a, c\rangle \in R$ 且$\langle c, b\rangle \in T$。在两种情况下，$\langle a, c\rangle \in R$ 且$\langle c, b\rangle \in S \cup T$，因此$\langle a, b\rangle \in R \circ(S \cup T)$。

综上所述，$R \circ(S \cup T)=(R \circ S) \cup(R \circ T)$。

(2) 该证明与(1)非常类似，只是运算顺序不同，但结论是一样的。其同样通过证明互为子集来证明两个集合相等。

(3) 设$\langle a, b\rangle \in R \circ(S \cap T)$，这说明存在某个元素 c，使得$\langle a, c\rangle \in R$ 且$\langle c, b\rangle \in S \cap T$。因为$\langle c, b\rangle \in S \cap T$，所以$\langle c, b\rangle \in S$ 且$\langle c, b\rangle \in T$。因此，$\langle a, b\rangle \in R \circ S$ 且$\langle a, b\rangle \in R \circ T$，$\langle a, b\rangle \in(R \circ S) \cap(R \circ T)$。

(4) 该证明与(3)类似，只是运算顺序不同。同样，可以证明左边是右边的子集，但无法

证明它们完全相等。

定理 3.2.6　设 R,S,T,Q 为任意的关系，其中 R 是 S 的子集，$R\subseteq S$。

(1) $R\circ T\subseteq S\circ T$。

(2) $T\circ R\subseteq T\circ S$。

证明：(1) 设 $\langle a,b\rangle\in R\circ T$。根据关系复合的定义，存在某个元素 c，使得 $\langle a,c\rangle\in R$ 且 $\langle c,b\rangle\in T$。由于 $R\subseteq S$，$\langle a,c\rangle\in R$，可得 $\langle a,c\rangle\in S$，因此有 $\langle a,c\rangle\in S$ 且 $\langle c,b\rangle\in T$。根据关系复合的定义，可得 $\langle a,b\rangle\in S\circ T$。因此，对于所有 $\langle a,b\rangle\in R\circ T$，都有 $\langle a,b\rangle\in S\circ T$，则 $R\circ T\subseteq S\circ T$ 成立。

(2) 设 $\langle a,b\rangle\in T\circ R$，这说明存在某个元素 c，使得 $\langle a,c\rangle\in T$ 且 $\langle c,b\rangle\in R$。由于 $R\subseteq S$，$\langle c,b\rangle\in R$，可得 $\langle c,b\rangle\in S$，因此有 $\langle a,c\rangle\in T$ 且 $\langle c,b\rangle\in S$。根据关系复合的定义，有 $\langle a,b\rangle\in T\circ S$。因此，对于所有 $\langle a,b\rangle\in T\circ R$，都有 $\langle a,b\rangle\in T\circ S$，则 $T\circ R\subseteq T\circ S$ 成立。

定义 3.2.7　设集合 A 上的二元关系 R，关系的**幂运算**如下。

(1) R^0 为单位关系 I_A，即 $R^0=I_A$，$I_A=\{\langle a,a\rangle|a\in A\}$。

(2) $R^1=R$，即关系的一次幂是关系本身，表示所有属于关系 R 的有序对 $\langle a,b\rangle$。

(3) $R^{n+1}=R^n\circ R$，$n\geqslant 0$。

例 3.2.6　(1) 设 $A=\{1,2,3\}$ 和关系 $R=\{\langle 1,2\rangle,\langle 2,3\rangle\}$，计算 R^2 和 R^3。

(2) 设 $B=\{a,b,c\}$ 和关系 $S=\{\langle a,b\rangle,\langle b,c\rangle,\langle c,a\rangle\}$，计算 S^2 和 S^3。

解：(1) $R^2=R\circ R=\{\langle 1,3\rangle\}$，$R^3=R^2\circ R=\varnothing$。

(2) $S^2=S\circ S$，从 S 中有：$\langle a,b\rangle$ 和 $\langle b,c\rangle$ 得到 $\langle a,c\rangle$，$\langle b,c\rangle$ 和 $\langle c,a\rangle$ 得到 $\langle b,a\rangle$，$\langle c,a\rangle$ 和 $\langle a,b\rangle$ 得到 $\langle c,b\rangle$。因此，$S^2=\{\langle a,c\rangle,\langle b,a\rangle,\langle c,b\rangle\}$。

$S^3=S^2\circ S$，从 S^2 和 S 中有：$\langle a,c\rangle$ 和 $\langle c,a\rangle$ 得到 $\langle a,a\rangle$，$\langle b,a\rangle$ 和 $\langle a,b\rangle$ 得到 $\langle b,b\rangle$，$\langle c,b\rangle$ 和 $\langle b,c\rangle$ 得到 $\langle c,c\rangle$。因此，$S^3=\{\langle a,a\rangle,\langle b,b\rangle,\langle c,c\rangle\}$。

将(2)中的关系 S 用矩阵进行表示，如下：

$$\boldsymbol{M}_S=\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}$$

则可以求 $\boldsymbol{M}_{S^2}$、$\boldsymbol{M}_{S^3}$ 的关系矩阵，分别为

$$\boldsymbol{M}_{S^2}=\boldsymbol{M}_S\circ\boldsymbol{M}_S=\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}\circ\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}=\begin{bmatrix}0&0&1\\1&0&0\\0&1&0\end{bmatrix}$$

$$\boldsymbol{M}_{S^3}=\boldsymbol{M}_{S^2}\circ\boldsymbol{M}_S=\begin{bmatrix}0&0&1\\1&0&0\\0&1&0\end{bmatrix}\circ\begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix}=\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}$$

图 3.7 展示了通过关系 S 多次后的结果。S^2 表示关系 S 的二次复合关系，即从一个顶点通过两条边到达另一个顶点的过程；在 S^3 中，每个顶点都有一条自环，表示通过关系 S 三次后，每个顶点都会回到自身。