Relations
Mathematical Structures for CS (CSCI 3030)

Prof. Harley Eades (heades@gru.edu).

Relationships are the driving force of life itself. Humans begin to make relationships the day they are born, and continue to do so through their entire life. This lifelong relationship building process is called learning. One could argue that the job of a Computer Scientist is to study relationships. The input to an algorithm is **related** to the output, and thus, defining an algorithm defines a relationship. In this section will describe how relationships can be defined mathematically. Furthermore, we will discuss some special properties relationships can have.

First, we give the definition of a binary relation.

Definition 1. *A* ***binary relation*** *on the sets A and B is a subset $r \subseteq A \times B$ of the cartesian product of A and B.*

The intuition of a relation is to relate the objects of A to the objects of B just as we did in the previous section modeling truth tables. There the relationship was propositional formulas to truth values. The definition above defines binary relations, but relations relating more than two sets can be constructed, but we only consider the binary case here.

The definition of binary relation is with respect to two sets A and B which may be distinct. However, it is more common in computer science that A and B are actually the same set.

Definition 2. *A* ***binary relation over*** *A is a binary relation $r \subseteq A \times A$.*

Binary relations over a set A are the most common, and we will restrict our attention to these types of binary relations for the remainder of this section.

Example 3. *The game rock-paper-scissors-lizard-spock:*

$$\begin{array}{rl}\mathcal{G} = \{ & (ROCK, SCISSORS), (ROCK, LIZARD),\\ & (PAPER, ROCK), (PAPER, SPOCK),\\ & (SCISSORS, PAPER), (SCISSORS, LIZARD),\\ & (LIZARD, PAPER), (LIZARD, SPOCK),\\ & (SPOCK, ROCK), (SPOCK, SCISSORS)\}\end{array}$$

It is very common to denote the fact that $(a, b) \in R$ for some binary relation R over A by aRb that is read "a is related to b with respect to R." This is called **infix** notation. Sometimes we also denote the same relation by $R(a, b)$ which is called **prefix** notation. For example, we can state that Spock beats rock by SPOCK $\mathcal{G}$ ROCK or by $\mathcal{G}$ (SPOCK, ROCK).

Depending on the relationship one wishes to make a relation may need to satisfy different properties. For example, the less-than-or-equal-to relation over the set of natural numbers has the property that for any $n \in \mathbb{N}$, $n \leq n$. However, the strictly less than relation over the natural numbers does not satisfy that property. We now state all of the most important properties of binary relations:

i. **Reflexivity**. A binary relation R over A is **reflexive** if and only if for any $a \in A$, aRa.

ii. **Symmetry**. A binary relation R over A is **symmetric** if and only if for any $a, b \in A$, aRb implies bRa.

iii. **Transitivity**. A binary relation R over A is **transitive** if and only if for any $a, b, c \in A$, aRb and bRc implies aRc.

iv. **Antisymmetry**. A binary relation R over A is **antisymmetric** if and only if for any $a, b \in A$, aRb and bRa implies $a = b$.

There are many example relations that one can come up with that have some of the above properties or even none.

Example 4.

i. Suppose $A = \{a, b, c\}$. Then the relation R over A defined as $R = \{(a, b), (b, c), (c, c), (c, a)\}$ is not reflexive, transitive, symmetric or even antisymmetric.

ii. Suppose $A = \{a, b, c\}$. Then the relation R over A defined as $R = \{(a, a), (b, c), (a, b), (b, c), (c, c), (c, a)\}$ is reflexive, but not transitive, symmetric or antisymmetric.

iii. Suppose $A = \{a, b, c\}$. Then the relation R over A defined as $R = \{(a, b), (b, a), (b, c), (c, b), (c, c), (a, c)(c, a)\}$ is not reflexive, transitive, or antisymmetric, but it is symmetric.

iv. Suppose $A = \{a, b, c\}$. Then the relation R over A defined as $R = \{(a, b), (b, c), (a, c), (c, c)\}$ is not reflexive, symmetric or antisymmetric, but it is transitive.

v. The relation $isMarriedTo$ over the set of all people is symmetric, but not antisymmetric.

Now we consider a few proofs.

Lemma 5. *Show that the strictly less-than relation on the natural numbers is transitive.*

Proof. We must prove that $\forall l, n, m \in \mathbb{N}$, if $l < n$ and $n < m$, then $l < m$. Suppose $n, m \in \mathbb{N}$, and that $l < n$ and $n < m$. Then there must exist non-zero natural numbers $k_1, k_2 \in \mathbb{N}$ such that $n = l + k_1$ and $m = n + k_2$. This then implies that $l = n - k_1$. Since we know $k_1 > 0$ and $k_2 > 0$ by assumption we know that $n - k_1 < n + k_2$ which is equivalent to $l < m$. Therefore, the strictly less-than relation over the natural numbers is transitive. □

Lemma 6. *Show that the strictly less-than relation on the natural numbers is anti-symmetric.*

Proof. We must prove that $\forall n, m \in \mathbb{N}$, if $n < m$ and $m < n$, then $n = m$. We use proof by contradiction. Suppose $n, m \in \mathbb{N}$, $n < m$, $m < n$, and $n \neq m$. We know that $n < m$ and $m < n$, thus there must exist non-zero natural numbers $k_1, k_2 \in \mathbb{N}$ such that $m = n + k_1$ and $n = m + k_2$. This implies that $m = n - k_2$. Now using substitution we may conclude that $n - k_2 = n + k_1$ which implies that $n = n + k_1 + k_2$ which is impossible because $k_1 > 0$ and $k_2 > 0$; a contradiction. Therefore, the strictly less-than relation over the natural numbers is anti-symmetric. □

Lemma 7. *Show that the subset operator is transitive.*

Proof. Suppose A, B, and C are sets. Then we must show that if $A \subset B$ and $B \subset C$, then $A \subset C$. So suppose $A \subset B$ and $B \subset C$. Then we know that for any $x \in A$ it is the case that $x \in B$, but we also know that for any $y \in B$ it is the case that $y \in C$, and hence, we may conclude that $x \in C$. Thus, $A \subset C$. □

Lemma 8. *Show that the subset operator is anti-symmetric.*

Proof. Suppose A and B are sets. Then we must show that if $A \subset B$ and $B \subset A$, then $A = B$, but this follows by definition. □

There are a special class of binary relations that capture the notion of being equivalent.

Definition 9. *A binary relation R over A is an* ***equivalence relation*** *if and only if R is reflexive, symmetric, and transitive.*

Example 10. *Notice that the subset operator on sets is a binary relation over the collection of all sets. That is, a set A is related to B if and only if $A \subseteq B$. We can define a new relation on sets, $R = \{(A, B) \mid A \subseteq B \land B \subseteq A\}$. The relation R is an equivalence relation. In fact, it is by definition the equality for sets.*

1 Closures

So far we have studied various properties of relations like reflexivity, transitivity, and symmetry, but also a few others. What if we have a relation, but it does not meet one of these properties, but we want our relation to? As an example suppose we are civil engineers in designing the interstate between three cities. The three locations will be denoted by L_1, L_2, and L_3. We also have the relation $R = \{(L_1, L_3), (L_3, L_2)\}$. This relation stands for "connected by a highway." We might have many more, but two works for our simple example. Now our goal is to figure out which locations we should add an interstate connecting cities. We only want to connect cities by an interstate if we have to go through another city to get the final destination. This is clearly true for L_1 and L_2, because to go from L_1 to L_2 we have to go through the city L_3, which can slow drivers down. To add the interstate we need to add the pair (L_1, L_3) to R, which not makes R transitive. What if we had a lot of pairs in R, and we wanted to add all our highways? Then we would go through adding all of the new pairs making R transitive. Doing this is called computing the transitive closure of a relation.

First, what is a closure?

Definition 11. *A set is closed under an operation if by applying the operation to each element of the set always produces an element of the set.*

Consider an example. Suppose $\Sigma = \{a, b, c\}$ (think of Σ's elements as characters), and then define a a word over Σ as $w = a_1 a_2 \cdots a_n$ where each $a_i \in \Sigma$ – think of words as strings. Now define the set $\Sigma^* = \{w \mid w \text{ is a word over } \Sigma\}$ – or the set of all possible strings whose characters are from Σ. The set Σ^* is a closed set under concatenation of words. That is, given any two elements $w_1, w_2 \in \Sigma^*$, then their concatenation $w_1 w_2$ is an element of Σ^*.

The set $P = \{n \in \mathbb{N} \mid n \text{ is prime }\}$ is not closed under multiplication. Simply take the elements $3 \in P$ and $5 \in P$, but $3 \times 5 = 15 \notin P$.

The relation $\mathsf{isMarriedTo}$ is closed under transitivity, but not symmetry.

Given a set that is potentially not closed under an operation then we can compute a new set – we want the smallest set possible – that has all the elements of the original set, but is closed under the operation. This is called taking the closure under the operation.

Definition 12. *The closure of a set S under an operation is the smallest set containing S that is closed under the operation.*

Examples:

1. The transitive closure of
$$R = \{(a, b), (b, c), (d, a), (c, d), (c, c)\}$$
is
$$R' = \{(a, b), (b, c), (d, a), (c, d), (c, c), (a, c), (b, d), (a, d), (d, c), (d, d)\}$$

2. The symmetric closure of the relation in the previous example is:

$$R'' = \{(a,b),(b,a),(b,c),(c,d),(d,a),(d,a),(c,d),(d,c),(c,c)\}$$

2 Mathematically Modeling Databases: Part 1

All of the relations we have discussed so far have been binary relations, but there is nothing from stopping one from having a relation over three sets, in fact, any relation, R, can be seen as a subset of the cartesian product $R \subseteq S_1 \times \cdots \times S_n$ for some n. In fact, these larger relations over potentially distinct sets are the building blocks of modern database theory. In this section we will discuss how relations can be used as a mathematical model of commercial databases.

Consider the following relation describing employees of the company Moraband:

$$\begin{array}{lll} \text{Emp} = \{ & (12348, \text{Darth}, \text{Sidious}), & (12349, \text{Darth}, \text{Zannah}), \\ & (12345, \text{Darth}, \text{Vadar}), & (12346, \text{Darth}, \text{Bane}), \\ & (12347, \text{Kylo}, \text{Ren})\} & \end{array}$$

Relations such as these can be represented by a database table:

Employee ID	First Name	Last Name
12348	Darth	Sidious
12349	Darth	Zannah
12345	Darth	Vadar
12346	Darth	Bane
12347	Kylo	Ren

Because database tables are in fact relations the order of the rows does not matter. Thus, the following table is the same as the previous one:

Employee ID	First Name	Last Name
12345	Darth	Vadar
12346	Darth	Bane
12347	Kylo	Ren
12348	Darth	Sidious
12349	Darth	Zannah

Now not every table is a database table.

Definition 13. *A database table (classically R-table) is a relation, R, with the following properties:*

- *each row in the table is a tuple of R,*
- *the ordering of rows is immaterial, and*
- *all rows are distinct from one another in content.*

Each of these follow directly from the mathematical definition of a relation. The first says that the entries of the table are the entries of the relation, the second says that the ordering of rows does not matter, but each row is a tuple of the relation which is itself a set, and hence, order does not matter. The last property says that a table cannot have two identical rows, but this would amount to having a duplicate tuple in R, but R is a set, and hence, cannot have any duplicates. Therefore, any table satisfying the above properties is a relation!

A database is now defined as a set of database tables and a set of operations on those tables. Database operations turn out to be operations on tables (relations). That is, they are algorithms that take as input one or more tables, and output a table or a value.

The a couple primitive operations are listed as follows:

- Projection: Given a database table, T, and a list of column names, return the database table T' containing all of the values for the columns in the input list. As an example the projection

  $$\textsf{project}\,\textsf{Emp}[\text{Employee ID}, \text{Last Name}]$$

  yields the following table:

Employee ID	Last Name
12345	Vadar
12346	Bane
12347	Ren
12348	Sidious
12349	Zannah

- Selection: Given a database table, T, and a predicate on the values of the T, produce the table, T', containing only those rows that satisfy the predicate. As an example the selection

  $$\textsf{select}\,\textsf{Emp}[\text{First Name} \neq \text{Darth}]$$

  yields the following table:

Employee ID	First Name	Last Name
12347	Kylo	Ren