CISC 1400
Discrete Structures
Chapter 4
Relations

Arthur G. Werschulz

Fordham University Department of Computer and Information Sciences

Summer, 2019

Why relations?

- Sets: rigorous way to talk about collections of objects
- Logic: rigorous way to talk about conditions and decisions
- Relations: rigorous way to talk about how objects can relate to each other
- **Example:**
  - A banking data base might have records consisting of the following attributes: customer name, address, SSN, account number, balance.
  - Problematic redundancy (joint accounts, customer with multiple accounts).
  - Break into parts to reduce redundancy:
    - Customer list: name, address, SSN, . . .
    - Account list: account number, balance
    - Depositor list: account number, SSN of owner

Outline

- Ways to describe relations between objects
- Describing a relation using English
- Properties of relations

Ways to describe relations between objects

A relation is a connection between objects in one set and objects in another (or possibly the same) set. How to describe?

- Use English. Example?
  - First set: set of names of people in this class.
  - Second set: the natural numbers
  - Relation? Associate each person with her age.
  - We might give this relation a name, such as *age*.
  - Note that the order of the sets matters.
- Use a picture.
  - Represent *domain* and *codomain* by two sets of dots.
  - Draw an arrow from dot in first set to dot in the second set if the (entity represented by the) first dot is related to the (entity represented by the) second dot.
- Use Cartesian product of the domain and codomain, along with set builder notation to represent the relation. Sometimes we use set-based notation (e.g., "$(x, y) \in r$"), sometimes prefix notation (e.g., "$r(x, y)$") and sometimes infix notation (e.g., "$x < y$").

Describing a relation

Must specify:

- the *domain* of the relation (in language terms, the "subject" of the relation),
- the *codomain* of the relation (in language terms, the "object" of the relation), and
- the connection or *rule* that links the elements in the domain to elements in the codomain.

Some terminology:

- When the domain and codomain are different, we have a relation *between* the two sets (or *from* the domain *to* the codomain).
- When the domain and codomain are the same, we have a relation *on* the given set.

Describing a relation (cont'd)

- **Example:** What elements are in the following relation?

Domain:	{Molly, Sandra, Mark}
Codomain:	{Molly, Sandra, Mark}
Rule:	(x, y) is in the relation iff x is sister of y.

  - Here, the domain and codomain are the same. We have a relation *on* the set {Molly, Sandra, Mark}.
  - Need to know family info to determine the relation!
  - (Molly, Mark) might be in the relation, but (Mark, Molly) can *not* be in the relation!
  - Suppose that Molly, Sandra, and Mark are all siblings. Then the relation consists of

    {(Molly, Sandra), (Molly, Mark), (Sandra, Molly), (Sandra, Mark)}

Describing a relation (cont'd)

What elements are in the following relation?

Domain:	the set of names of people in your family
Codomain:	{red, black, brown, blonde, flaxen, pink, green}
Rule:	(x, y) is in the relation if and only if x's hair is y.

- The domain and the codomain are different. We have a relation *from* {names of people in your family} *to* {red, black, brown, blonde, flaxen, pink, green}.
- Again, need family information.
- Might have two people with same hair color.
- Might have some "unclaimed" hair color (e.g., green). This color would not appear in the relation.
- Might have a family member without any of given hair colors. This person would not appear in the relation.

Describing a relation (cont'd)

What elements are in the following relation?

Domain:	the set $\mathbb{N}$ of natural numbers
Codomain:	$\mathbb{N}$
Rule:	$r_{\text{even}} = \{(x, y) \in \mathbb{N} \times \mathbb{N} : x + y \text{ is even}\}$.

- This is a relation on $\mathbb{N}$.
- r_{even} is an infinite list of ordered pairs from $\mathbb{N}$.
- Can't easily list r_{even}.
- Can *characterize* r_{even}.
  - two even numbers added will give an even number,
  - as will two odd numbers added,
  - but not an even and an odd number added.

So r_{even} consists of pairs from $\mathbb{N}$, in which the elements of each pair are either both even or both odd.

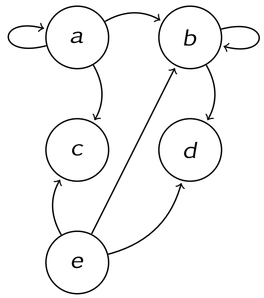
Describing a relation (cont'd)

Sometimes we use a graphical representation of a relation on a set.

Example: Consider the relation

$$\{(a,a),(a,b),(a,c),(b,b),(b,d),(e,b),(e,c),(e,d)\}$$

Pictorial representation:



Describing a relation (cont'd)

Sometimes we use a tabular representation of a relation.

Example: Consider the relation

$$\{(a,a),(a,b),(a,c),(b,b),(b,d),(e,b),(e,c),(e,d)\}$$

Tabular representation:

a	a	a	b	b	e	e	e
a	b	c	b	d	b	c	d

Properties of relations

- A relation *on a set* is one in which the domain and codomain are the same.
- A relation on a set may be any of the following:
  - reflexive
  - irreflexive
  - symmetric
  - antisymmetric
  - transitive

Reflexivity

A relation r on a set S is said to be *reflexive* if

$$(x,x) \in r \qquad \text{for any } x \in S.$$

- **Example:** The relation
  Domain: $\mathbb{N}$
  Codomain: $\mathbb{N}$
  Rule: $r_{\text{even}} = \{(x,y) \in \mathbb{N} \times \mathbb{N} : x+y \text{ is even}\}$.
  is reflexive.
  We need to show that $(x,x) \in r_{\text{even}}$ for all $x \in \mathbb{N}$, i.e., that $x+x$ is even for all $x \in \mathbb{N}$. But $x+x=2x$ is always even, for any $x \in \mathbb{N}$. □
- **Example:** The relation
  Domain: $\mathbb{N}$
  Codomain: $\mathbb{N}$
  Rule: $r_{\text{odd}} = \{(x,y) \in \mathbb{N} \times \mathbb{N} : x+y \text{ is odd}\}$.
  is not reflexive, since $(1,1) \notin r_{\text{odd}}$.

Reflexivity and irreflexivity

- A relation r on S is said to be *irreflexive* if

$$(x,x) \notin r \qquad \text{for any } x \in S.$$

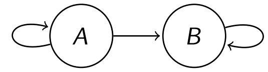
- The relation r_{odd} is irreflexive, since $x+x$ is never odd.
- **Warning:** “Irreflexive” does *not* mean “not reflexive”.
  There are relations that are neither reflexive nor irreflexive.
  Example: Let $S = \{1,2\}$. The relation

$$r = \{(1,1)\}$$

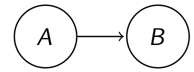
  is neither reflexive nor irreflexive.

Reflexivity and irreflexivity

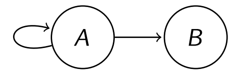
- The graph of a reflexive relation has a “loop” at every node.



- The graph of an irreflexive relation has no loops at any node.



- If a relation is neither reflexive nor irreflexive, then there will be loops at some (but not all) of its nodes.



Symmetry

A relation r on a set S is said to be *symmetric* if

$$(x,y) \in r \Rightarrow (y,x) \in r \qquad \text{for any } x,y \in S.$$

- The relation r_{even} is symmetric, since

$$\begin{aligned}(x,y) \in r_{\text{even}} &\equiv x+y \text{ is even } \equiv y+x \text{ is even}\\ &\equiv (y,x) \in r_{\text{even}}\end{aligned}$$

- The relation r_{odd} is symmetric, since

$$\begin{aligned}(x,y) \in r_{\text{odd}} &\equiv x+y \text{ is odd } \equiv y+x \text{ is odd}\\ &\equiv (y,x) \in r_{\text{odd}}\end{aligned}$$

Symmetry and antisymmetry

- A relation r on S is *antisymmetric* if

$$x,y \in S, x \neq y, (x,y) \in r \Rightarrow (y,x) \notin r.$$

- **Example:** The relation $<$ on $\mathbb{N}$ is antisymmetric, since

$$x,y \in \mathbb{N}, x \neq y, x < y \Rightarrow y \not< x.$$

- **Example:** The relation $\leq$ on $\mathbb{N}$ is antisymmetric, since

$$x,y \in \mathbb{N}, x \neq y, x \leq y \Rightarrow y \not\leq x.$$

Symmetry and antisymmetry (cont'd)

- **Example:** The $\subseteq$ relation on $\mathscr{P}(S)$ is antisymmetric, since

$$A,B \subseteq S, A \neq B, A \subseteq B \Rightarrow B \not\subseteq A$$

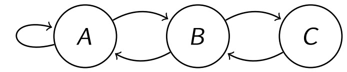
- **Warning:** "Antisymmetric" does *not* mean "not symmetric". There are relations that are neither symmetric nor antisymmetric.
  Example: Let $S = \{1,2,3\}$. The relation

$$r = \{(1,2),(2,1),(1,3)\}$$

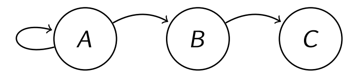
  is neither symmetric nor antisymmetric.

Symmetry and antisymmetry

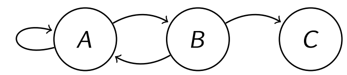
- In the graph of a symmetric relation, all the (non-loop) edges are "two-way streets".



- In the graph of an antisymmetric relation, all of the (non-loop) edges are "one-way streets".



- In a relation is neither symmetric nor antisymmetric, some streets are "two-way", some are "one-way".



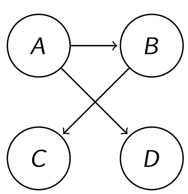
Transitivity

A relation r on a set S is said to be *transitive* if

$$(x,y) \in r \text{ and } (y,z) \in r \Rightarrow (x,z) \in r \qquad \text{for any } x,y,z \in S$$

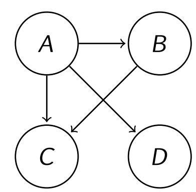
In other words, the relation allows for shortcuts.

Transitive or intransitive?



Transitivity (cont'd)

Transitive or intransitive?

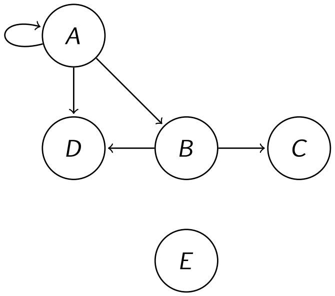


No "easy test" for transitivity:

- Try all possibilities.
- Use knowledge of the relation.

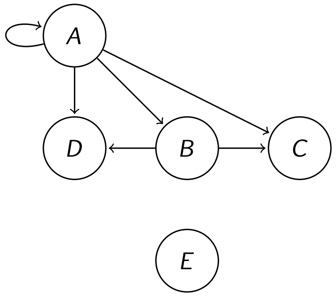
Transitivity (cont'd)

Transitive or non-transitive?



Transitivity (cont'd)

Transitive or non-transitive?



Transitivity (cont'd)

- **Example:** The $<$ relation on $\mathbb{Z}$ is transitive, since

$$x < y \text{ and } y < z \Rightarrow x < z$$

- **Example:** The $\neq$ relation on $\mathbb{N}$ is not transitive.
  Let $x = 1, y = 2, z = 1$.
  Then $x \neq y$ and $y \neq z$, but we do not have $x \neq z$.
- **Example:** The $\subseteq$ relation on $\mathscr{P}(S)$ is transitive.
  Suppose that $A \subseteq B$ and $B \subseteq C$; is $A \subseteq C$?
  Need to show that $x \in A \Rightarrow x \in C$ for all $x \in A$.
  So let $x \in A$.
  Since $x \in A$ and $A \subseteq B$, we know $x \in B$.
  Since $x \in B$ and $B \subseteq C$, we know $x \in C$. $\square$

Let's try some examples!

Here are some relations. Are they reflexive? irreflexive? symmetric? antisymmetric? transitive?

- The relation "is a sibling of" on the set of all people.
- The relation "has the same age as" on the set of all people.
- The relation $\{(3,4),(6,8),(3,9),(4,3),(9,-2),(4,9)\}$ on $\mathbb{Z}$.
- The relation $\leq$ on $\mathbb{Z}$.
- The relation $<$ on $\mathbb{Z}$.
- The relation $\subseteq$ on $\mathscr{P}(S)$.
- The relation $\subset$ on $\mathscr{P}(S)$.
- The relation "beats" on {rock, paper, scissors}.
- The "friend" relation on Facebook.

Relational Databases

- First proposed by E. F. Codd (IBM) in 1970s.
- At the heart of Oracle, Microsoft Access, Microsoft SQL Server, IBM's dBase.
- Main ideas:
  - Store data in tables.
  - Each table has rows and columns.
  - In each table, special column called the *key*, used to identify rows. (Slight simplification.) Examples: SSN, FIDN, account number, ...
  - Key entry for each row of table must be unique.
  - Can look up row in a table by specifying its key.

Relational Databases (cont'd)

Basic information for our social network is stored in the *Friends* table:

Name	City	Hometown	Sex	Birthday	Status
Alex	Topeka	Topeka	F	02/15/1996	S
Alyssa	Hartford	Albany	F	02/01/1964	M
Angela	Charlotte	Denver	F	06/15/1967	S
Anna	Hartford	Hartford	F	5/19/1989	U
Chryssi	Boston	Boston	F	12/23/1985	S
Ellen	Hartford	Boston	F	04/01/1958	M
Erik	South Park	South Park	M	08/01/1997	S
Frank	Harrisburg	Phoenix	M	12/12/1969	D
Grace	Hartford	Boston	F	02/25/1962	U
Joanna	Topeka	Topeka	F	02/15/1996	S
John	Augusta	Atlanta	M	10/25/1991	S
⋮	⋮	⋮	⋮	⋮	⋮

Relational Databases (cont'd)

The *Education* table for a social network might look like the following:

Name	University	Class	Degree	Major
Ellen	Suffolk University	1986	JD	Criminal Law
Ellen	Harvard University	1980	BA	English
Frank	Dartmouth	1996	PhD	Physics
Frank	Dartmouth	1990	BS	Physics
Grace	Fordham University	2006	MS	Computer Science
Grace	Boston College	1984	BS	Computer Science
Larry	CUNY	2007	MBA	Finance
Larry	NYU	2005	BA	Literature
Lauren	Vassar College	1985	MA	Sociology
Lauren	Duke	1983	BA	English
⋮	⋮	⋮	⋮	⋮

Relational Databases (cont'd)

Note the following:

- The *Friends* table is a 6-ary relation on

$$\text{Name} \times \text{City} \times \text{Hometown} \times \text{Sex} \times \text{Birthday} \times \text{Status}$$

- The *Name* column can be used as a key for *Friends* table.
- The *Education* table is a 5-ary relation on

$$\text{Name} \times \text{University} \times \text{Class} \times \text{Degree} \times \text{Major}$$

- No column of the *Education* table serves as a key.

Relational Databases (cont'd)

- Extract info from a table by
  - looking up tables for key entries, or
  - looking across several tables and cross-indexing.
- For example:
  - Look up entry in *Friends* table for the key *Ellen* to find her birthday (04/01/1958).
  - Cross-index *Ellen* in the *Education* table to find that she got her JD degree from Suffolk University in 1986.
  - Hence Ellen was 1986 – 1958 = 28 years old when she received her JD degree.

Relational Databases (cont'd)

- Most common way to extract info from a relational database with the Structured Query Language, or SQL.
- SQL, invented in the 1970s, is based on *relational algebra*, a combination of relations and logic.
- The SQL **select** operator is used to extract info from a relational database.

Relational Databases (cont'd)

Example: Find the birthdate and sex of all married people in the database:

> **select** *Birthdate, Sex*
> **from** *Friends*
> **where** *Status* = M;

Using mathematical notation: The *Friends* relation may be described as

$$r_{\mathrm{f}} = \{(n_i, c_i, h_i, s_i, b_i, st_i) : i \in \{0, \ldots, 25\}\}$$

and the results of the SQL **select** operation can be written as

$$\{(b_i, s_i) : (n_i, c_i, h_i, s_i, b_i, st_i) \in r_{\mathrm{f}} \wedge st_i = \mathrm{M}\}$$

Relational Databases (cont'd)

Example: Use cross-referencing to find the name, birthdate, and graduation date of everyone in the social network:

> **select** *Name, Birthdate, Class*
> **from** *Friends* **join** *Education*
> **on** *Friends.Name* = *Education.Name*;

Represent this as a relation?
Write *Education* as a 5-ary relation:

$$r_{\mathrm{e}} = \{(n_i, u_i, cl_i, d_i, m_i) : i \in \{0, \ldots, 19\}\}.$$

("cl" stands for "class").
Then

$$\{(n, b_i, cl_j) : (n, c_i, h_i, s_i, b_i, st_i) \in r_{\mathrm{f}} \wedge (n, u_j, cl_j, d_j, m_j) \in r_{\mathrm{e}}\}.$$

Relational Databases (cont'd)

Example: Create a "friends of friends" table, which we can use to suggest new friends for our members.
Solution: Suppose that *FriendOf* is a two-column table that represents current friendships, something like

Name1	Name2
Alex	Lena
Alex	Joanna
⋮	⋮
Lena	Alex
Lena	Anna
Lena	Joanna
⋮	⋮

Since Alex is a friend of Lena and Lena is a friend of Joanna, then Alex is a FOAF of Joanna.

Relational Databases (cont'd)

Example (cont'd): Suppose that *FriendOfA* and *FriendOfB* are two copies of the *FriendOf* table. Then (a first approximation to) a solution is given by

select *FriendOfA.Name1*, *FriendOfB.Name2*
from *FriendOfA* **join** *FriendOfB*
on *FriendOfA.Name2* = *FriendOfB.Name1*
as *FriendSuggestions*;

This needs a little fine-tuning, to avoid the following bogus friend suggestions:

- yourself, and
- someone who's already a friend.