Relations

Relations

- Relationships between elements of sets occur very often.
  - (Employee, Salary)
  - (Students, Courses, GPA)
- Relationships between elements of sets are represented using the structure called relation, which is just a subset of the Cartisian product of the sets.
- We use ordered pairs (or *n-tuples*) of elements from the sets to represent relationships.

Binary Relations

- Let A and B be any sets. A *binary relation R* from A to B, (i.e., with signature $R{:}A\times B$) can be identified with a subset of $A\times B$.

E.g., let **<** : $\mathbf{N}\times\mathbf{N}$ can be seen as $\{(n,m) \mid n < m\}$

- $(a,b)\in R$ means that a is related to b (by R)
- Also written as *aRb*; also *R(a,b)*
  - E.g., a<**b** and **<** (a,b) both mean (a,b)∈ **<**
- A binary relation R corresponds to a characteristic function $P_R{:}A\times B\rightarrow\{\mathbf{T},\mathbf{F}\}$

Example

A: {students at UNR}, B: {courses offered at UNR}

R: "relation of students enrolled in courses"

(Jason, CS365), (Mary, CS201) are in R

If Mary does not take CS365, then (Mary, CS365) is not in R!

If CS480 is not being offered, then (Jason, CS480), (Mary, CS480) are not in R!

Complementary Relations

- Let $R{:}A,B$ be any binary relation.
- Then, $\cancel{R}{:}A\times B$, the *complement* of R, is the binary relation defined by

$$\cancel{R} :\equiv \{(a,b) \in A\times B \mid (a,b) \notin R\} = (A\times B) - R\}$$

- Note this is just $\overline{R}$ if the universe of discourse is $U = A\times B$; thus the name *complement.*
- Note the complement of $\cancel{R}$ is R.

Example: $\not< \, = \{(a,b) \mid (a,b) \notin \, <\} = \{(a,b) \mid \neg a<b\} = \, \geq$

Inverse Relations

- Any binary relation $R{:}A\times B$ has an *inverse* relation $R^{-1}{:}B\times A$, defined by

  $$R^{-1} :\equiv \{(b,a) \mid (a,b)\in R\}.$$

  E.g., $<^{-1} = \{(b,a) \mid a<b\} = \{(b,a) \mid b>a\} = >$.

- *E.g.*, if R:People x Foods is defined by

  $a\ R\ b \Leftrightarrow a \text{ eats } b$, then:

  $b\ R^{-1}\ a \Leftrightarrow a \text{ eats } b$

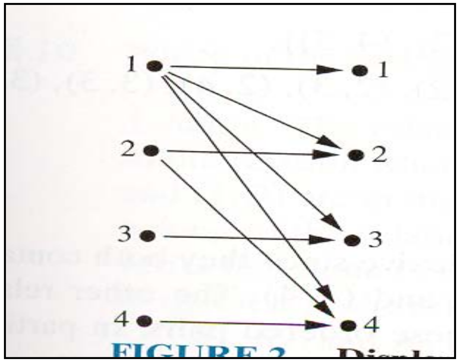
  (Compare: *b is eaten by a,* passive voice.)

Functions as Relations

A function f:A→B is a relation from A to B

A relation from A to B is not always a function f:A→B (e.g., relations could be one-to-many)

Relations are generalizations of functions!



Relations on a Set

- A (binary) relation from a set *A* to itself is called a relation *on A*. A relation on the set *A* is a relation from *A* to *A*.

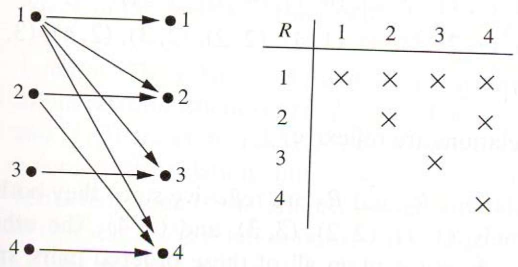
- *E.g.*, the "**<**" relation is defined as a relation *on* **N**.

Relations on a Set

A (binary) relation from a set *A* to itself is called a relation *on* the set *A*.

A: {1,2,3,4}

R={(a,b) | a divides b}



R	1	2	3	4
1	×	×	×	×
2		×		×
3			×	
4				×

FIGURE 2 **Displaying the Ordered Pairs in the Relation *R* from Example 4.**

Example

How many relations are there on a set A with *n* elements?

Reflexivity and relatives

- A relation R on A is *reflexive* iff $\forall a \in A, (aRa)$.
  - *E.g.*, the relation $\geq :\equiv \{(a,b) \mid a \geq b\}$ is reflexive.
- R is *irreflexive* iff $\forall a \in A, (\neg aRa)$
- Note "*irreflexive*" does **NOT** mean "*not reflexive*", which is just $\neg\forall a \in A, (aRa)$.
- E.g., if Adore={(j,m),(b,m),(m,b)(j,j)} then this relation is neither reflexive nor irreflexive

Reflexivity and relatives

- Theorem: A relation *R* is *irreflexive* iff its *complementary* relation *R'* is reflexive.
  - Example: < is irreflexive; ≥ is reflexive.
  - Proof: trivial
  - Is the "divide" relation on the set of positive integers reflexive?

Some examples

- Reflexive:

  =, ‘have same cardinality’, ⇔

  <=, >=, ⇒, ⊆, etc.

- Irreflexive:

  <, >, `have different cardinality’, ⊂, ‘is logically stronger than’

Symmetry & relatives

- A binary relation *R* on *A* is *symmetric* iff $\forall a,b((a,b)\in R \leftrightarrow (b,a)\in R)$.
  - *E.g.*, = (equality) is symmetric. < is not.
  - “is married to” is symmetric, “likes” is not.
- A binary relation *R* is *asymmetric* if $\forall a,b((a,b)\in R \rightarrow (b,a)\notin R)$.
  - **Examples:** < is asymmetric, “Adores” is not.
- Let R={(j,m),(b,m),(j,j)}. Is R (a)symmetric?

Symmetry & relatives

- Let R={(j,m),(b,m),(j,j)}.

R is not symmetric (because it does not contain (m,b) and because it does not contain (m,j)).

R is not asymmetric, due to (j,j)

Some direct consequences

Theorems:

1. R is symmetric iff $R = R^{-1}$,
2. R is asymmetric iff $R \cap R^{-1}$ is empty.

Symmetry & its relatives

1. R is symmetric iff $R = R^{-1}$

$\Rightarrow$ Suppose R is symmetric. Then

$(x,y) \in R \Leftrightarrow$

$(y,x) \in R \Leftrightarrow$

$(x,y) \in R^{-1}$

$\Leftarrow$ Suppose $R = R^{-1}$ Then

$(x,y) \in R \Leftrightarrow$

$(x,y) \in R^{-1} \Leftrightarrow$

$(y,x) \in R$

Symmetry & relatives

2. R is asymmetric iff $R \cap R^{-1}$ is empty.

(Straightforward application of the definitions of asymmetry and R^{-1})

- Question: Can you construct a model in which the relation “son of” is symmetric?
- More precisely: find domain A and relation R on A such that R is symmetric and R(x,y) can reasonably be read as ‘x is a son of y’

- Question: Can you construct a model in which the relation “son_of” is symmetric?
- Solution: any model in which there are no x,y such that son_of(x,y) is true
- E.g., A = {John, Mary, Sarah},
  A**x**A ⊇ R= {}

Antisymmetry

- Consider the relation x≤y
- Is it symmetrical?
- Is it asymmetrical?
- Is it reflexive?
- Is it irreflexive?

Antisymmetry

- Consider the relation $x \leq y$
- Is it symmetrical? No
- Is it asymmetrical?
- Is it reflexive?
- Is it irreflexive?

Antisymmetry

- Consider the relation x≤y
- Is it symmetrical? No
- Is it asymmetrical? No
- Is it reflexive?
- Is it irreflexive?

Antisymmetry

- Consider the relation $x\leq y$
- Is it symmetrical? No
- Is it asymmetrical? No
- Is it reflexive? Yes
- Is it irreflexive?

Antisymmetry

- Consider the relation $x \leq y$
- Is it symmetrical? No
- Is it asymmetrical? No
- Is it reflexive? Yes
- Is it irreflexive? No

Antisymmetry

- Consider the relation x≤y
  - It is not symmetric. (For instance, 5≤6 but not 6≤5)
  - It is not asymmetric. (For instance, 5 ≤5)
  - The pattern: the only times when (a,b)∈ ≤ and (b,a)∈ ≤ are when a=b
- This is called antisymmetry

  Can you say this in predicate logic?

Antisymmetry

- A binary relation *R* on *A* is *antisymmetric* iff $\forall a,b((a,b)\in R \wedge (b,a)\in R) \rightarrow a=b)$.
- Examples: $\leq$, $\geq$, $\subseteq$
- Another example: the earlier-defined relation Adore={(j,m),(b,m),(m,b)(j,j)}

- How would you define transitivity of a relation? What are its ‘relatives’?

Transitivity & relatives

- A relation *R* is *transitive* iff (for all *a*,*b*,*c*)

  $((a,b)\in R \land (b,c)\in R) \rightarrow (a,c)\in R.$

- A relation is non*transitive* iff it is not transitive.

- A relation *R* is in*transitive* iff (for all *a*,*b*,*c*)

  $((a,b)\in R \land (b,c)\in R) \rightarrow \neg(a,c)\in R.$

Transitivity & relatives

- What about these examples:
  - “x is an ancestor of y”
  - “x likes y”
  - “x is located within 1 mile of y”
  - “x +1 =y”
  - “x beat y in the tournament”
  - “x is stronger than y”

Transitivity & relatives

- What about these examples:
  - “is an ancestor of” is transitive.
  - “likes” is neither trans nor intrans.
  - “is located within 1 mile of”
    is neither trans nor intrans
  - “x +1 =y” is intransitive
  - “x beat y in the tournament” is neither trans nor intrans
  - “x is stronger than y” is transitive.

Exploring the difference between relations and functions

Totality:

- A relation $R{:}A\times B$ is *total* if for every $a\in A$, there is at least one $b\in B$ such that $(a,b)\in R$.
  - N.B., it does not follow that R^{-1} is total
  - It does not follow that R is a function.

Functionality

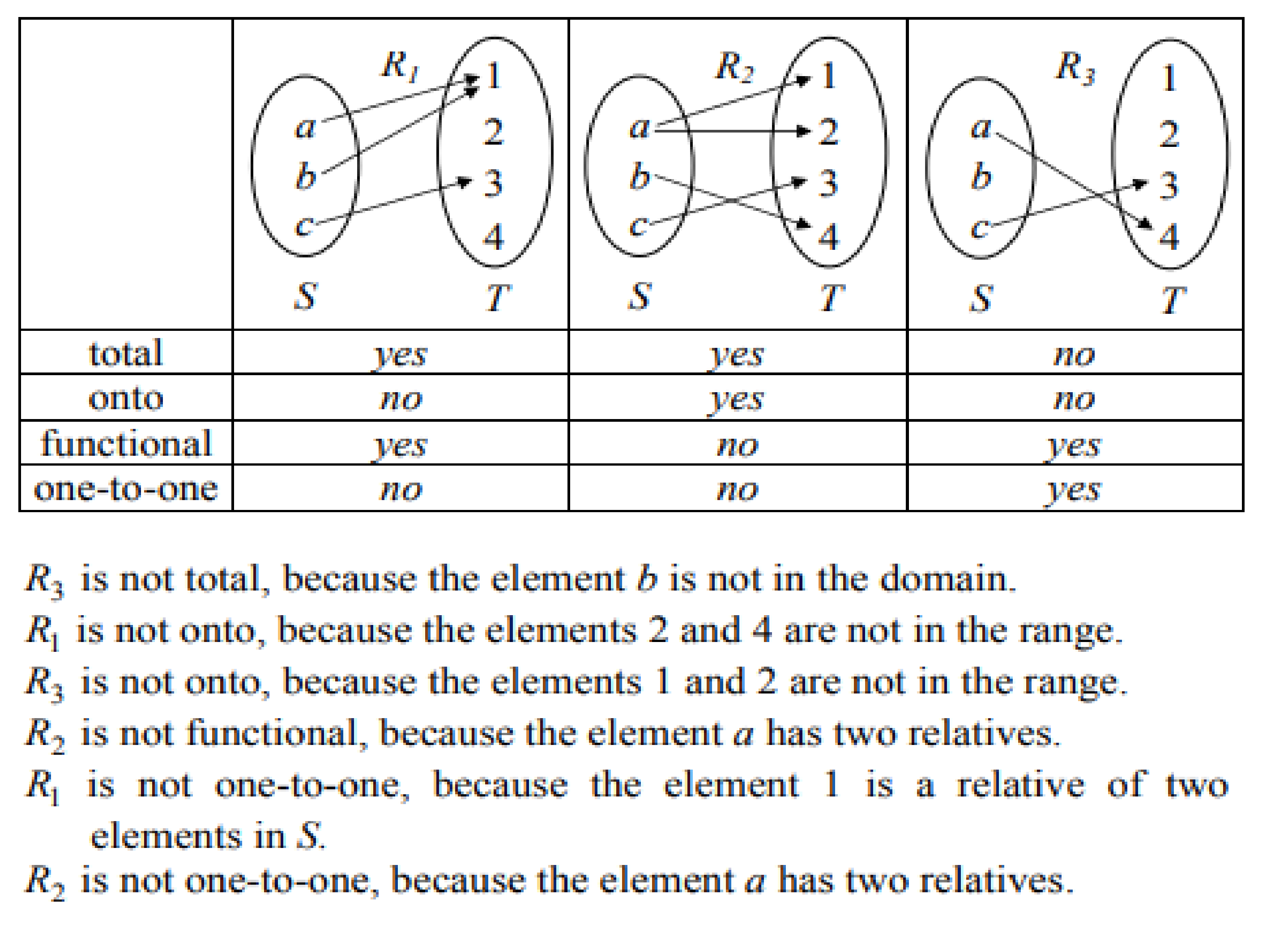
Functionality:

- A relation R: $A \times B$ is *functional* iff, for every $a \in A$, there is *at most one* $b \in B$ such that $(a,b) \in R$.
  - A functional relation R: $A \times B$ does not have to be total (there may be $a \in A$ such that $\neg\exists b \in B\,(aRb)$).
- Say that “R is functional”, using predicate logic

Functionality

- R: $A \times B$ is *functional* iff, for every $a \in A$, there is *at most one* $b \in B$ such that $(a,b) \in R$.
  $\forall a \in A$: $\neg\exists\ b_1, b_2 \in B\ (b_1 \neq b_2 \wedge aRb_1 \wedge aRb_2)$.
- If R is functional and total relation, then R can be seen as a function R: $A \rightarrow B$
  Hence one can write $R(a)=b$ as well as aRb, R(a,b), and $(a,b) \in R$. Each of these mean the same.

Functionality



R_1
a
b
c
1
2
3
4
S
T
R_2
a
b
c
1
2
3
4
S
T
R_3
a
b
c
1
2
3
4
S
T

	R_1	R_2	R_3
total	*yes*	*yes*	*no*
onto	*no*	*yes*	*no*
functional	*yes*	*no*	*yes*
one-to-one	*no*	*no*	*yes*

R_3 is not total, because the element b is not in the domain.

R_1 is not onto, because the elements 2 and 4 are not in the range.

R_3 is not onto, because the elements 1 and 2 are not in the range.

R_2 is not functional, because the element a has two relatives.

R_1 is not one-to-one, because the element 1 is a relative of two elements in S.

R_2 is not one-to-one, because the element a has two relatives.

Functionality

- *Definition:* *R* is *antifunctional* iff its inverse relation R^{-1} is functional.

  (**Exercise:** Show that iff *R* is functional and antifunctional, and both it and its inverse are total, then it is a bijective function.)

Combining what you've learned about functions and relations

Consider the relation R:**N**→**N** defined as

R = {(x,y) | x∈**N** ∧ y∈**N** ∧ y=x+1}.

Questions:

1. Is R total? Why (not)?
2. Is R functional? Why (not)?
3. Is R an injection? Why (not)?
4. Is R a surjection? Why (not)?

Combining Relations

- Two relations can be combined in a similar way to combining two sets.

$$R_1 \cup R_2$$

$$R_1 \cap R_2$$

$$R_1 - R_2$$

$$R_2 - R_1$$

Composite Relations

- Let $R{:}A\times B$, and $S{:}B\times C$. Then the *composite* $S\circ R$ of R and S is defined as:

  $S\circ R = \{(a,c) \mid \exists b{:}\ aRb \wedge bSc\}$

- Does this remind you of something?

Composite Relations

- Let $R{:}A\times B$, and $S{:}B\times C$. Then the *composite* $S\circ R$ of R and S is defined as:

  $S\circ R = \{(a,c) \mid \exists b{:}\ aRb \wedge bSc\}$

- Does this remind you of something?
- Function composition ...
- ... except that $S\circ R$ accommodates the fact that S and R may not be functional

Composite Relations

- Function composition is a special case of relation composition: Suppose S and R are functional. Then we have (using the definition above, then switching to function notation)

  $S \circ R(a,c)$ *iff* $\exists b$: $aRb \wedge bSc$
  iff $R(a)=b$ *and* $S(b)=c$ *iff* $S(R(a))=c$

Suppose

- Adore={(a,b),(b,c),(c,c)}
- Detest={(b,d),(c,a),(c,b)}
- Adore°Detest=
- Detest°Adore=

Suppose

- Adore={(a,b),(b,c),(c,c)}
- Detest={(b,d),(c,a),(c,b)}
- Adore°Detest= {(c,b),(c,c)}
- Detest°Adore=

Suppose

- Adore={(a,b),(b,c),(c,c)}
- Detest={(b,d),(c,a),(c,b)}
- Adore°Detest= {(c,b),(c,c)}
- Detest°Adore= {(a,d),(b,a),(b,b),(c,a),(c,b)}

Example

R is a relation from {1,2,3} to {1,2,3,4}

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

S is a relation from {1,2,3,4} to {0,1,2}

S = {(1,0),(2,0),(3,1),(3,2),(4,1)}

$$R \circ S = \{(1,0),(1,1),(2,1),(2,2),(3,0),(3,1)\}$$

Composite Relations

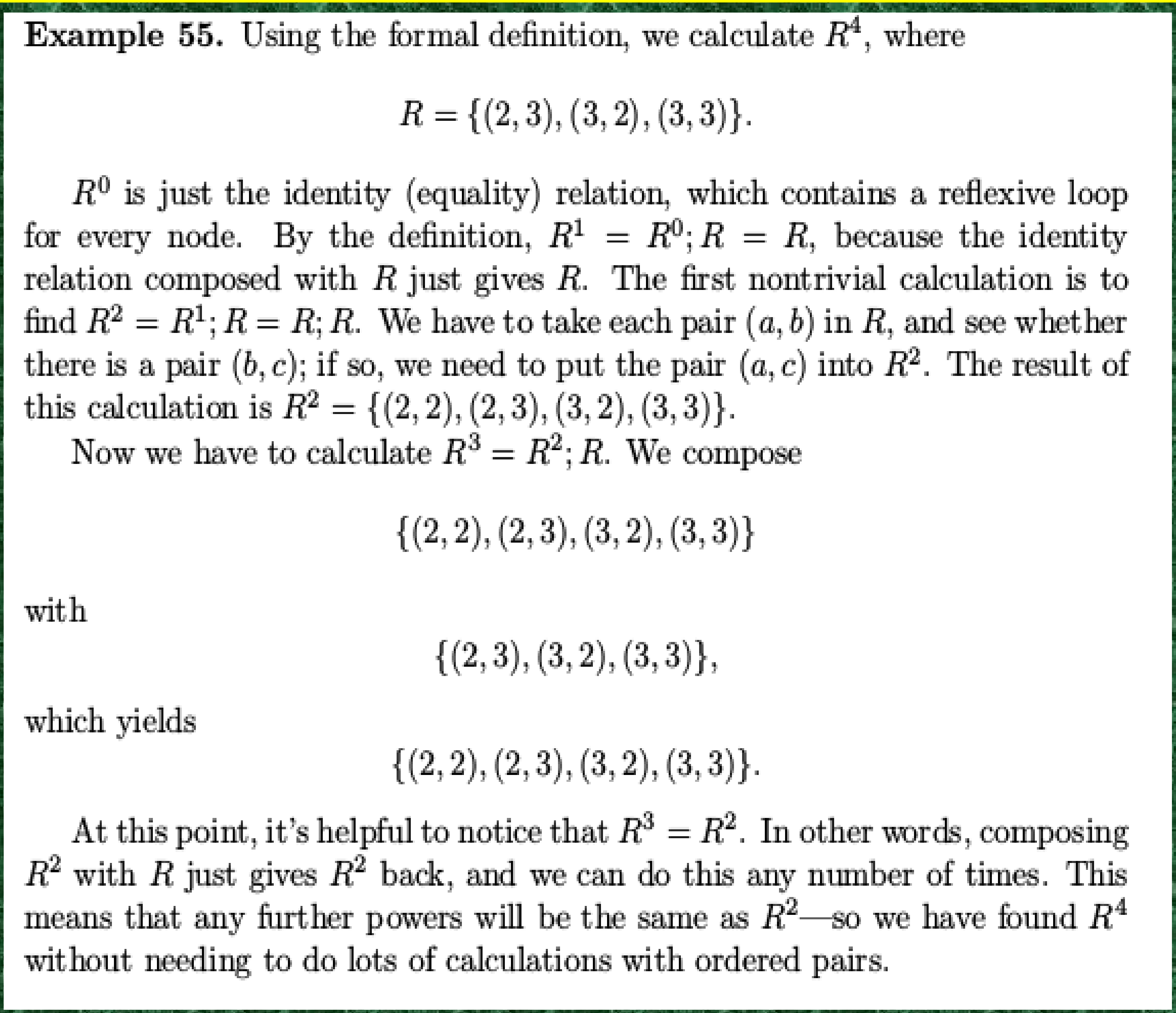
- Let $R{:}A\leftrightarrow B$, and $S{:}B\leftrightarrow C$. Then the *composite* $S\circ R$ of R and S is defined as:

  $S\circ R = \{(a,c) \mid aRb \wedge bSc\}$

- Function composition $f\circ g$ is an example.
- The nth power R^n of a relation R on a set A can be defined recursively by:

$$R^1 :\equiv \mathrm{R}\,; \qquad R^{n+1} :\equiv R^n\circ R \qquad \text{for all } n\geq 0.$$

Composite Relations



Example 55. Using the formal definition, we calculate R^4, where

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

R^0 is just the identity (equality) relation, which contains a reflexive loop for every node. By the definition, $R^1 = R^0; R = R$, because the identity relation composed with R just gives R. The first nontrivial calculation is to find $R^2 = R^1; R = R; R$. We have to take each pair (a, b) in R, and see whether there is a pair (b, c); if so, we need to put the pair (a, c) into R^2. The result of this calculation is $R^2 = \{(2,2), (2,3), (3,2), (3,3)\}$.

Now we have to calculate $R^3 = R^2; R$. We compose

$$\{(2,2), (2,3), (3,2), (3,3)\}$$

with

$$\{(2,3), (3,2), (3,3)\},$$

which yields

$$\{(2,2), (2,3), (3,2), (3,3)\}.$$

At this point, it's helpful to notice that $R^3 = R^2$. In other words, composing R^2 with R just gives R^2 back, and we can do this any number of times. This means that any further powers will be the same as R^2—so we have found R^4 without needing to do lots of calculations with ordered pairs.

§7.2: *n*-ary Relations

- An *n*-ary relation *R* on sets $A_1, \ldots, A_n$, is a subset

$$R \subseteq A_1 \times \ldots \times A_n.$$

- This is a straightforward generalisation of a binary relation. For example:
- 3-ary relations:
  - a is between b and c;
  - a gave b to c

§7.2: *n*-ary Relations

- An *n*-ary relation *R* on sets $A_1,\ldots,A_n$, is a subset

$$R \subseteq A_1 \times \ldots \times A_n.$$

- The sets A_i are called the *domains* of *R*.
- The *degree* of *R* is *n*.
- *R* is *functional in the domain* A_i if it contains at most one *n*-tuple $(\ldots, a_i, \ldots)$ for any value a_i within domain A_i.

§7.2: *n*-ary Relations

- *R* is *functional in the domain* A_i if it contains at most one *n*-tuple $(\ldots, a_i, \ldots)$ for any value a_i within domain A_i.
- Generalisation: being functional in a combination of two or more domains.

§7.2: *n*-ary Relations

- An *n*-ary relation *R* on sets $A_1,\ldots,A_n$, written R:$A_1,\ldots,A_n$, is a subset

$$R \subseteq A_1 \times \ldots \times A_n.$$

- The *degree* of *R* is *n*.
- Example: R consists of 5-tuples (A,N,S,D,T)

A: airplane flights, N: flight number,

S: starting point, D: destination, T: departure time

Databases

- The time required to manipulate information in a database depends in how this information is stored.
- Operations: add/delete, update, search, combine etc.
- Various methods for representing databases have been developed.
- We will discuss the “*relational model*”.

Relational Databases

- A database consists of ***records***, which are *n-tuples*, made up of ***fields***.
- A *relational database* represents records as an *n-ary* relation *R*.

  (STUDENT_NAME, ID, MAJOR, GPA)
- Relations are also called "tables" (e.g., displayed as tables often)

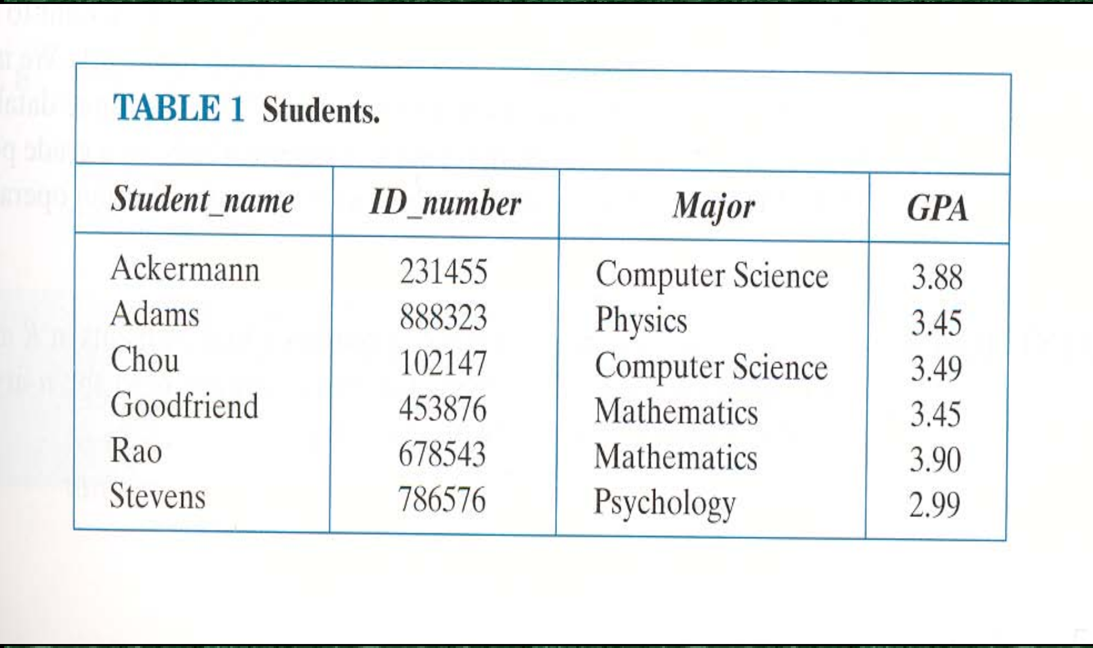


TABLE 1 **Students.**

Student_name	*ID_number*	*Major*	*GPA*
Ackermann	231455	Computer Science	3.88
Adams	888323	Physics	3.45
Chou	102147	Computer Science	3.49
Goodfriend	453876	Mathematics	3.45
Rao	678543	Mathematics	3.90
Stevens	786576	Psychology	2.99

Relational Databases

- A domain A_i of an *n-ary* relation is called *primary key* when no two *n-tuples* have the same value on this domain (e.g., ID)
- A *composite key* is a subset of domains $\{A_i, A_j, \ldots\}$ such that an n-tuple $(\ldots, a_i, \ldots, a_j, \ldots)$ is determined uniquely for each composite value $(a_i, a_j, \ldots) \in A_i \times A_j \times \ldots$

Relational Databases

- A *relational database* is essentially just a set of relations.
- A domain A_i is a (*primary) key* for the database if the relation *R* is functional in A_i.
- A *composite key* for the database is a set of domains $\{A_i, A_j, \ldots\}$ such that *R* contains at most 1 *n*-tuple $(\ldots,a_i,\ldots,a_j,\ldots)$ for each composite value $(a_i, a_j,\ldots)\in A_i\times A_j\times\ldots$

Selection Operators

- Let A be any n-ary domain $A=A_1\times\ldots\times A_n$, and let $C{:}A\rightarrow\{\mathbf{T},\mathbf{F}\}$ be any *condition* (predicate) on elements (n-tuples) of A.
- The *selection operator* s_C maps any n-ary relation R on A to the relation consisting of all n-tuples from R that satisfy C:

  $s_C(R) = \{a\in R \mid \mathrm{C}(a) = \mathbf{T}\}$

Selection Operator Example

- Let $A = \text{StudentName} \times \text{Standing} \times \text{SocSecNos}$
- Define a condition Upperlevel on *A:*

  $$UpperLevel(name,standing,ssn) \Leftrightarrow$$
  $$((standing = \text{junior}) \vee (standing = \text{senior}))$$

- Then, $s_{UpperLevel}$ takes any relation *R* on *A* and produces the subset of R involving of *just* the junior and senior students.

Projection Operators

- Let $A = A_1 \times \ldots \times A_n$ be any n-ary domain, and let $\{i_k\}=(i_1,\ldots,i_m)$ be a sequence of indices all falling in the range 1 to n,

- Then the *projection operator* on n-tuples

$$P_{\{i_k\}} : A \rightarrow A_{i_1} \ldots \times A_{i_m}$$

is defined by:

$$P_{\{i_k\}}(a_1,\ldots,a_n) = (a_{i_1},\ldots,a_{i_m})$$

Projection Example

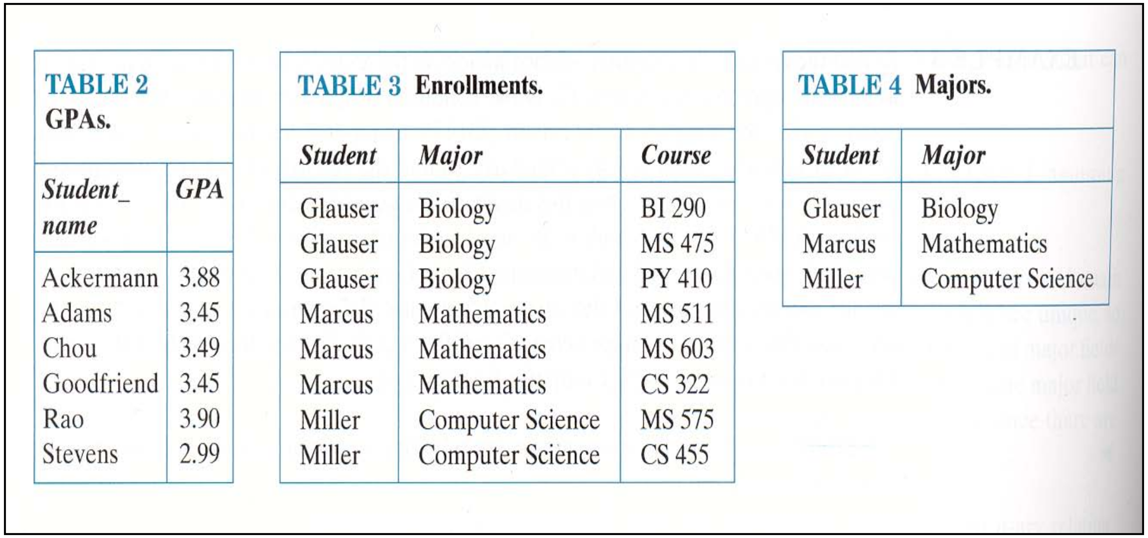


TABLE 2 GPAs.

Student_ name	*GPA*
Ackermann	3.88
Adams	3.45
Chou	3.49
Goodfriend	3.45
Rao	3.90
Stevens	2.99

TABLE 3 Enrollments.

Student	*Major*	*Course*
Glauser	Biology	BI 290
Glauser	Biology	MS 475
Glauser	Biology	PY 410
Marcus	Mathematics	MS 511
Marcus	Mathematics	MS 603
Marcus	Mathematics	CS 322
Miller	Computer Science	MS 575
Miller	Computer Science	CS 455

TABLE 4 Majors.

Student	*Major*
Glauser	Biology
Marcus	Mathematics
Miller	Computer Science

Note that fewer rows may result when a projection is applied !

Projection Example

- Suppose we have a domain *Cars*=*Model*× *Year*× *Color*. (note *n*=3).
- Consider the index sequence $\{i_k\}$= 1,3. (*m*=2)
- Then the projection $P_{\{i_k\}}$ maps each tuple (a_1,a_2,a_3) = (*model*,*year*,*color*) to its image:

$$\left(a_{i_1},a_{i_2}\right)=\left(a_1,a_3\right)=\left(model,color\right)$$

- This operator can be applied to a relation *R*⊆*Cars* to obtain a list of the model/color combinations available.

Join Operator

- Puts two relations together to form a combined relation which is their composition:
- Iff the tuple (A,B) appears in R_1, and the tuple (B,C) appears in R_2, then the tuple (A,B,C) appears in the join $J(R_1,R_2)$.
  - A, B, and C can also be sequences of elements.

Join Example

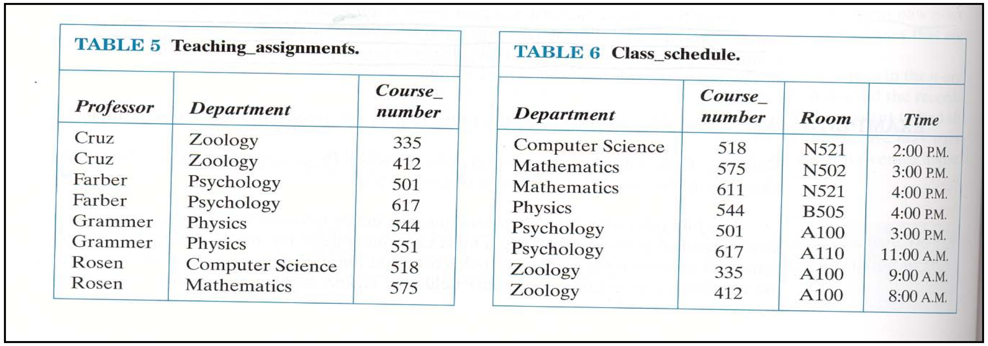


TABLE 5 Teaching_assignments.

Professor	*Department*	*Course_ number*
Cruz	Zoology	335
Cruz	Zoology	412
Farber	Psychology	501
Farber	Psychology	617
Grammer	Physics	544
Grammer	Physics	551
Rosen	Computer Science	518
Rosen	Mathematics	575

TABLE 6 Class_schedule.

Department	*Course_ number*	*Room*	*Time*
Computer Science	518	N521	2:00 P.M.
Mathematics	575	N502	3:00 P.M.
Mathematics	611	N521	4:00 P.M.
Physics	544	B505	4:00 P.M.
Psychology	501	A100	3:00 P.M.
Psychology	617	A110	11:00 A.M.
Zoology	335	A100	9:00 A.M.
Zoology	412	A100	8:00 A.M.

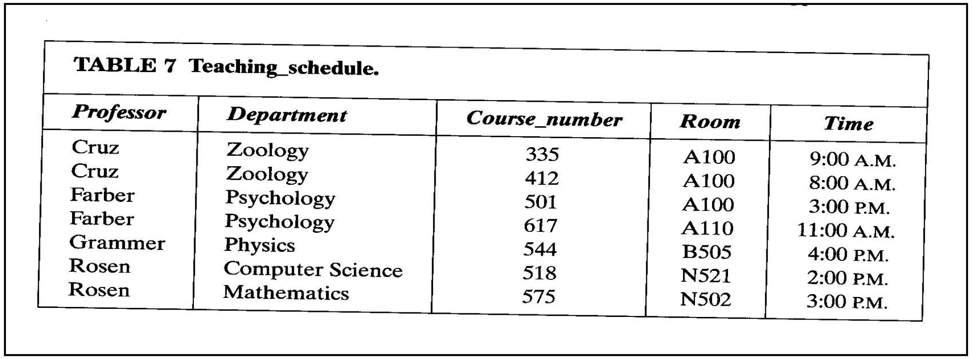


TABLE 7 Teaching_schedule.

Professor	*Department*	*Course_number*	*Room*	*Time*
Cruz	Zoology	335	A100	9:00 A.M.
Cruz	Zoology	412	A100	8:00 A.M.
Farber	Psychology	501	A100	3:00 P.M.
Farber	Psychology	617	A110	11:00 A.M.
Grammer	Physics	544	B505	4:00 P.M.
Rosen	Computer Science	518	N521	2:00 P.M.
Rosen	Mathematics	575	N502	3:00 P.M.

Join Example

- Suppose R_1 is a teaching assignment table, relating *Lecturers* to *Courses*.
- Suppose R_2 is a room assignment table relating *Courses* to *Rooms*, *Times*.
- Then $J(R_1,R_2)$ is like your class schedule, listing (*lecturer*,*course*,*room*,*time*).
- (Joins are similar to *relation composition*. For precise definition, see Rosen, p.486)

SQL Example

SELECT *Departure_ Time*

FROM *Flights*

WHERE *destination="Detroit"*

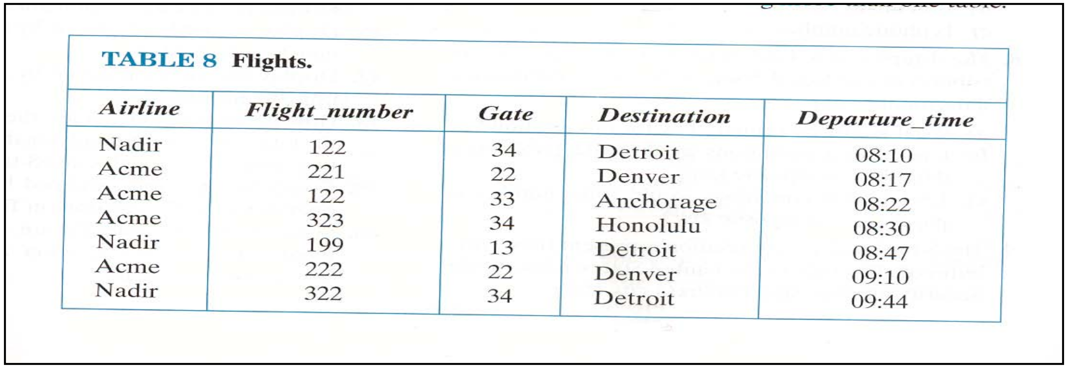


TABLE 8 **Flights.**

Airline	*Flight_number*	*Gate*	*Destination*	*Departure_time*
Nadir	122	34	Detroit	08:10
Acme	221	22	Denver	08:17
Acme	122	33	Anchorage	08:22
Acme	323	34	Honolulu	08:30
Nadir	199	13	Detroit	08:47
Acme	222	22	Denver	09:10
Nadir	322	34	Detroit	09:44

Find projection P5 of the selection of 5-tuples that satisfy the constraint "destination=Detroit"

§7.3: Representing Relations

- Before saying more about the n-th power of a relation, let's talk about representations
- Some ways to represent *n*-ary relations:
  - With a list of n-tuples.
  - With a function from the (n-ary) domain to **{T,F}**.
- Special ways to represent binary relations:
  - With a zero-one matrix.
  - With a directed graph.

§7.3: Representing Relations

- Why bother with alternative representations? Is one not enough?
- One reason: some calculations are easier using one representation, some things are easier using another
- There are even some basic ideas that are suggested by a particular representation

It’s often worth playing around with different representations

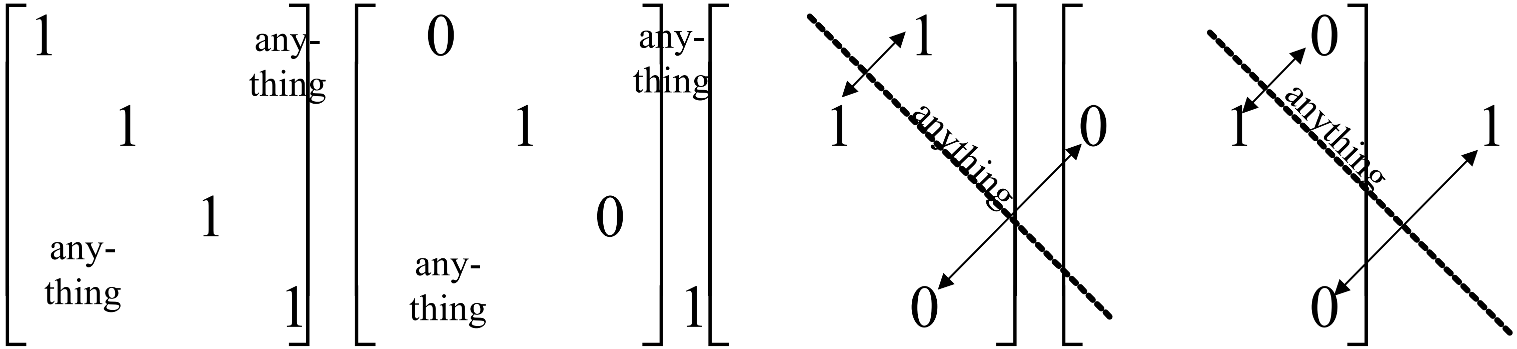
Using Zero-One Matrices

- To represent a binary relation $R{:}A\times B$ by an $|A|\times|B|$ 0-1 matrix $\mathbf{M}_R = [m_{ij}]$, let $m_{ij} = 1$ iff $(a_i,b_j)\in R$.
- *E.g.*, Suppose Joe likes Susan and Mary, Fred likes Mary, and Mark likes Sally.
- Then the 0-1 matrix representation of the relation Likes:Boys×Girls relation is:

$$\begin{array}{l} & \text{Susan} & \text{Mary} & \text{Sally} \\ \text{Joe} & 1 & 1 & 0 \\ \text{Fred} & 0 & 1 & 0 \\ \text{Mark} & 0 & 0 & 1 \end{array}$$

Zero-One Reflexive, Symmetric

- Terms: *Reflexive*, *non-Reflexive*, s*ymmetric*, *and antisymmetric*.
  - These relation characteristics are very easy to recognize by inspection of the zero-one matrix.



Reflexive: all 1's on diagonal

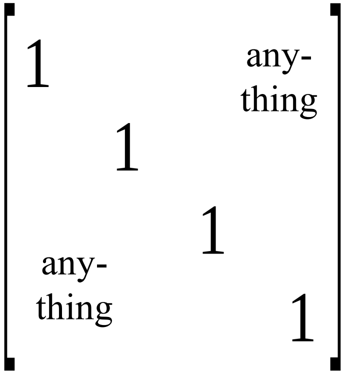
Non-reflexive: some 0's on diagonal

Symmetric: all identical across diagonal

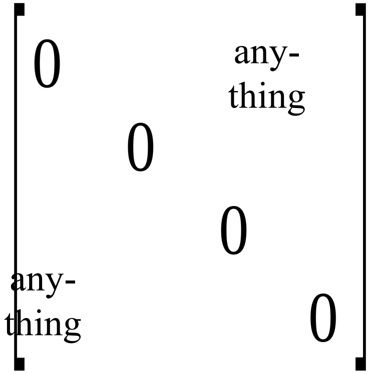
Antisymmetric: all 1's are across from 0's

Zero-One Reflexive, Symmetric

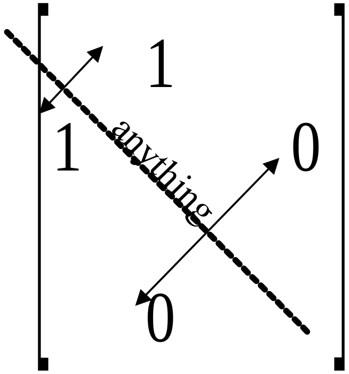
- Recall: *Reflexive*, *irreflexive*, s*ymmetric*, *and asymmetric* relations.
  - These relation characteristics are very easy to recognize by inspection of the zero-one matrix.



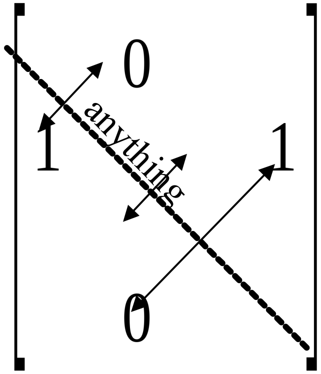
Reflexive:
only 1’s on diagona



Irreflexive:
only 0’s on diagonal



Symmetric:
all identical
across diagonal



Asymmetric:
all 1’s are across
from 0’s

Using Directed Graphs

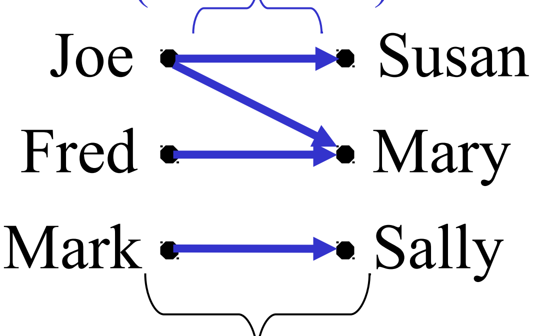
- A *directed graph* or *digraph* $G=(V_G,E_G)$ is a set V_G of *vertices (nodes)* with a set $E_G \subseteq V_G \times V_G$ of *edges (arcs)*. Visually represented using dots for nodes, and arrows for edges. A relation $R{:}A\times B$ can be represented as a graph $G_R=(V_G=A\cup B,\ E_G=R)$.

Matrix representation $\mathbf{M}_R$:

$$\begin{array}{c|ccc} & \text{Susan} & \text{Mary} & \text{Sally} \\ \hline \text{Joe} & 1 & 1 & 0 \\ \text{Fred} & 0 & 1 & 0 \\ \text{Mark} & 0 & 0 & 1 \end{array}$$

Graph rep. G_R:

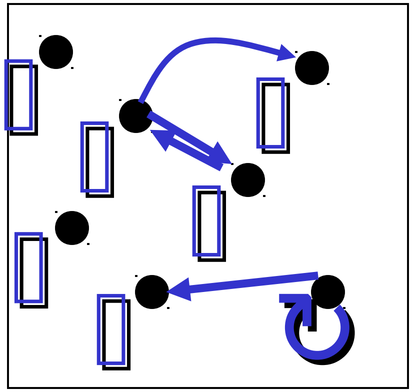
Edge set E_G (blue arrows)



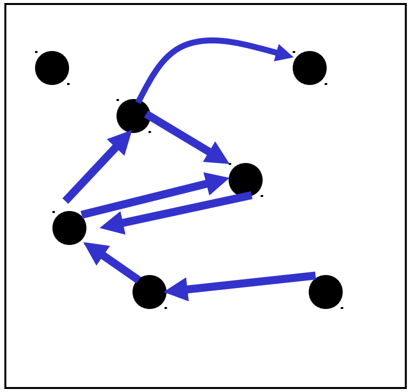
Node set V_G (black dots)

Digraph Reflexive, Symmetric

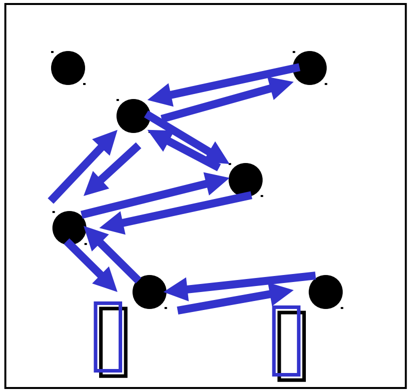
Many properties of a relation can be determined by inspection of its graph.



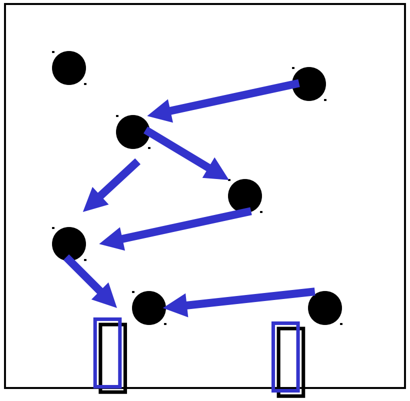
Reflexive:
Every node
has a self-loop



Irreflexive:
No node
links to itself



Symmetric:
Every link is
bidirectional



Antisymmetric:
never (a,b) and
(b,a), unless a=b

These are not symmetric & not asymmetric

These are non-reflexive & non-irreflexive

§7.4: Closures of Relations

- For any property *X*, the "*X* closure" of a set *A* is defined as the "smallest" superset of *A* that has the given property.
- The *reflexive closure* of a relation *R* on *A* is obtained by adding (*a*,*a*) to *R* for each $a \in A$. *I.e.*, it is $\boxed{R \cup I_A}$
- The *symmetric closure* of *R* is obtained by adding (*b*,*a*) to *R* for each (*a*,*b*) in *R*. *I.e.*, it is $\boxed{R \cup R^{-1}}$
- The *transitive closure* or *connectivity relation* of *R* is obtained by repeatedly adding (*a*,*c*) to *R* for each (*a*,*b*), (*b*,*c*) in *R*.
  - *I.e.*, it is

$$\boxed{R^* = \bigcup_{n \in \mathbf{Z}^+} R^n}$$

Back to the n-th power of a relation

- A *path* of length n from node a to b in the directed graph G is a sequence $(a,x_1), (x_1,x_2), \ldots, (x_{n-1},b)$ of n ordered pairs in E_G.
  - Note: there exists a path of length n from a to b in R if and only if $(a,b) \in R^n$.
- A path of length $n \geq 1$ from a to itself is a *cycle*.
- R*: the relation that holds between a and b iff there exists a finite path from a to b using R.
  - Note: R* is transitive!

§7.4: Closures of Relations

- For any property *X*, the *X* closure of a set *A* is defined as the "smallest" superset of *A* that has property *X*. More specifically,
  - The *reflexive closure* of a relation *R* on *A* is the smallest superset of R that is reflexive.
  - The *symmetric closure* of *R* is the smallest superset of R that is symmetric
  - The *transitive closure* of *R* is the smallest superset of R that is transitive

Calculating closures

- The *reflexive closure* of a relation R on A is obtained by "adding" (a,a) to R for each $a \in A$. *I*.*e*., it is $R \cup I_A$ (Check that this is the r.c.)
- The *symmetric closure* of R is obtained by "adding" (b,a) to R for each (a,b) in R. *I*.*e*., it is $R \cup R^{-1}$ (Check that this is the s.c.)
- The *transitive closure* of R is obtained by *"repeatedly"* adding (a,c) to R for each (a,b),(b,c) in R …

Calculating closures

- Adore={(a,b),(b,c),(c,c)}
- Detest={(b,d),(c,a),(c,b)}
- The *symmetric closure* of ...

  ... Adore=

  ... Detest=

Calculating closures

- Adore={(a,b),(b,c),(c,c)}
- Detest={(b,d),(c,a),(c,b)}
- The *symmetric closure* of ...

  ... Adore={(a,b),(b,c),(c,c),(b,a), (c,b)}

  ... Detest={(b,d),(c,a),(c,b), (d,b), (a,c),(b,c)}

Calculating closures

- Adore={(a,b),(b,c),(c,c)}
- Detest={(b,d),(c,a),(c,b)}
- The *transitive closure* of ...

  ... Adore=

  ... Detest=

Calculating closures

- Adore={(a,b),(b,c),(c,c)}
- Detest={(b,d),(c,a),(c,b)}
- The *transitive closure* of ...

  ... Adore={(a,b),(b,c),(c,c),(a,c)}

  ... Detest={(b,d),(c,a),(c,b),(c,d)}

Paths in Digraphs/Binary Relations

- A *path* of length n from node a to b in the directed graph G (or the binary relation R) is a sequence $(a,x_1), (x_1,x_2), \ldots, (x_{n-1},b)$ of n ordered pairs in E_G (or R).
  - An empty sequence of edges is considered a path of length 0 from a to a.
  - If any path from a to b exists, then we say that a is *connected to b*. (*"You can get there from here."*)
- A path of length $n\geq 1$ from a to a is called a *circuit* or a cycle.
- Note that there exists a path of length n from a to b in R if and only if $(a,b)\in R^n$.

Simple Transitive Closure Alg.

A procedure to compute R^* with 0-1 matrices.

```
procedure transClosure($\mathbf{M}_R$:rank-$n$ 0-1 mat.)
    $\mathbf{A} := \mathbf{B} := \mathbf{M}_R$;
    for $i := 2$ to $n$  begin
        $\mathbf{A} := \mathbf{A} \odot \mathbf{M}_R$;   $\mathbf{B} := \mathbf{B} \vee \mathbf{A}$      {join}
    end            {note A represents $R^i$}
    return B {Alg. takes $\Theta(n^4)$ time}
```

Roy-Warshall Algorithm

- Uses only $\Theta(n^3)$ operations!

```
Procedure Warshall($\mathbf{M}_R$ : rank-$n$ 0-1 matrix)
  $\mathbf{W} := \mathbf{M}_R$
  for $k$ := 1 to $n$
    for $i$ := 1 to $n$
      for $j$ := 1 to $n$
        $w_{ij} := w_{ij} \vee (w_{ik} \wedge w_{kj})$
  return W   {this represents $R^*$}
```

$w_{ij} = 1$ means there is a path from i to j going only through nodes $\leq k$

§7.5: Equivalence Relations

- Definition: An *equivalence relation* on a set *A* is any binary relation on *A* that is *reflexive, symmetric, and transitive.*

§7.5: Equivalence Relations

- Definition: An *equivalence relation* on a set *A* is any binary relation on *A* that is *reflexive, symmetric, and transitive.*
  - *E.g.*, = is an equivalence relation.
  - But many other relations follow this pattern too

§7.5: Equivalence Relations

- Definition: An *equivalence relation* on a set A is any binary relation on A that is *reflexive, symmetric, and transitive.*
  - *E.g.*, = is an equivalence relation.
  - For any function $f{:}A\rightarrow B$, the relation "have the same f value", or $=_f :\equiv \{(a_1,a_2) \mid f(a_1)=f(a_2)\}$ is an equivalence relation,
    - *e.g.*, let m="mother of" then $=_m$ = "have the same mother" is an equivalence relation

Equivalence Relation Examples

- “Strings *a* and *b* are the same length.”
- “Integers *a* and *b* have the same absolute value.”

Equivalence Relation Examples

Let's talk about relations between functions:

1. How about: $R(f,g) \Leftrightarrow f(2)=g(2)$?
2. How about: $R(f,g) \Leftrightarrow f(1)=g(1) \vee f(2)=g(2)$?

Equivalence Classes

- Let *R* be any equivalence relation.
- The *equivalence class of a under R*,

  $[a]_R :\equiv \{\ x \mid aRx \}$ (optional subscript *R*)

  - Intuitively, this is the set of all elements that are "equivalent" to *a* according to *R*.
  - Each such *b* (including *a* itself) can be seen as a *representative* of $[a]_R$.

Equivalence Classes

- Why can we talk so loosely about elements being equivalent to each other (as if the relation didn’t have a direction)?
- In some sense, it does not matter which representative of an equivalence class you take as your starting point:

If aRb then $\{ x \mid aRx \} = \{ x \mid bRx \}$

Equivalence Classes

If aRb then aRx ⇔ bRx **Proof:**

1. Suppose aRb while bRx.
Then aRx follows directly by *transitivity.*

2. Suppose aRb while aRx.
aRb implies bRa *(symmetry).* But
bRa and aRx imply bRx by *transitivity*

Equivalence Classes

We now know that

If aRb then $\{ x \mid aRx \} = \{ x \mid bRx \}$

Equally,

If aRb then $\{ x \mid xRa \} = \{ x \mid xRb \}$
(due to symmetry)

In other words, an equivalence class based on R is simply a maximal set of things related by R

Equivalence Class Examples

- "(Strings a and b) *have the same length.*"
  - Suppose a has length 3. Then $[a]$ = the set of all strings of length 3.
- "(Integers a and b) *have the same absolute value.*"
  - $[a]$ = the set $\{a, -a\}$

Equivalence Class Examples

- "Formulas φ and ψ contain the same number of brackets" (e.g. for formulas of propositional logic, using the strict syntax)
- Now what is $[((p\wedge q)\vee r)]$?

Equivalence Class Examples

- Consider the equivalence relation ⇔ (i.e., logical equivalence, for example between formulas of propositional logic)
- What is $[p\wedge q]$?

Partitions

- A *partition* of a set *A* is a collection of disjoint nonempty subsets of A that have A as their union.

- Intuitively: a partition of A divides A into separate parts (in such a way that there is no remainder).

Partitions and equivalence classes

- Consider a *partition* of a set A into A_1, ..A_n
  - The A_i's are all disjoint : For all x and for all i, if $x \in A_i$ and $x \in A_j$ then $A_i = A_j$
  - The union of the A_i's = A

Partitions and equivalence classes

- A *partition* of a set *A* can be viewed as the set of all the equivalence classes $\{A_1, A_2, \ldots\}$ for some equivalence relation on *A*.
- For example, consider the set A={1,2,3,4,5,6} and its partition {{1,2,3},{4},{5,6}}
- R = { (1,1),(2,2),(3,3),(1,2),(1,3),(2,3),(2,1),(3,1),(3,2) ,(4,4), (5,5),(6,6),(5,6),(6,5) }

Partitions and equivalence classes

- We sometimes say:
  - A partition of A *induces* an equivalence relation on A
  - An equivalence relation on A *induces* a partition of A

§7.6: Partial Orderings

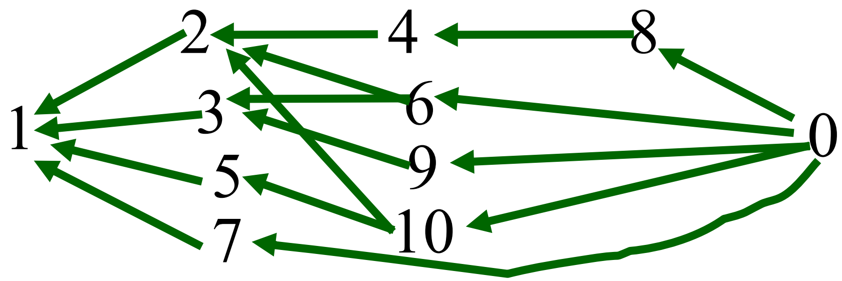
- A relation R on A is called a *partial ordering* or *partial order* iff it is reflexive, antisymmetric, and transitive.
  - We often use a symbol looking something like $\preccurlyeq$ (or analogous shapes) for such relations.
  - Examples: $\leq$, $\geq$ on real numbers, $\subseteq$, $\supseteq$ on sets.
  - Another example: the "divides" relation $\mid$ on $\mathbf{Z}^+$.
    - It is not necessarily the case that either $a \preccurlyeq b$ or $b \preccurlyeq a$.
- A set A together with a partial order $\preccurlyeq$ on A is called a *partially ordered set* or *poset* and is denoted $(A, \preccurlyeq)$.

Posets as Noncyclical Digraphs

- If a set S is partially ordered by a relation R then its graph can be simplified:
  - Looping edges need not be drawn, because they can be inferred
  - Instead of drawing edges for R(a,b), R(b,c) and R(a,c) , the latter can be omitted (because it can be inferred)
  - If direction of arrows is represented as left-to-right (or top-down) order then it’s called a Hasse diagram
    (We won’t do that here)

Posets as Noncyclical Digraphs

- There is a one-to-one correspondence between posets and the reflexive+transitive closures of noncyclical digraphs.
- Example: consider the poset ({0,…,10}, |)
  - Its "minimal" digraph:



Posets as Noncyclical Digraphs

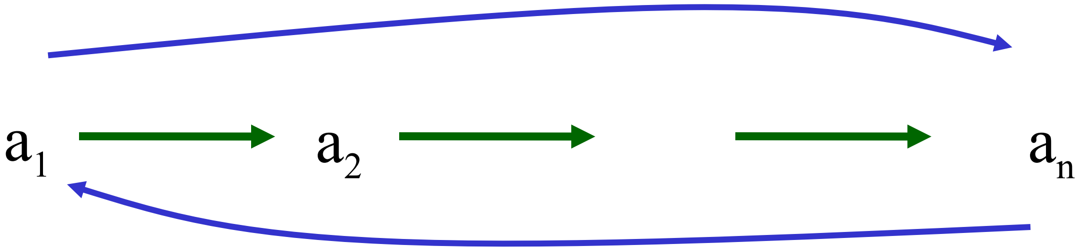
- Prove: a graph for a partial order cannot contain cycles

Posets as Noncyclical Digraphs

- **Theorem**: a graph for a partial order cannot contain cycles with length > 1.
- **Proof**: suppose there is a cycle $a_1Ra_2R\ldots Ra_nRa_1$ (with n>1). Then, with n-1 applications of transitivity, we have a_1Ra_n. But also a_nRa_1, which conflicts with antisymmetry.

Posets do not have cycles

- **Proof**: suppose there is a cycle $a_1Ra_2R\ldots Ra_nRa_1$. Then, with n-1 applications of transitivity, we have a_1Ra_n. But also a_nRa_1, which conflicts with antisymmetry.



- Can something be both a poset and an equivalence relation?

- Can something be both a poset and an equivalence relation?
  - Equiv: ref, sym, trans
  - Poset: ref, antisym, trans
- Can a relation (that is reflexive and transitive) be both sym and antisym?

- Can a relation that is reflexive and transitive be both sym and antisym?
- Yes: the empty relation R={} is an example
- But any relation $R \subseteq \{(x,x): x \in A\}$ will also qualify.
  - It’s reflexive
  - It’s symmetric and antisymmetric
  - It’s transitive
- Other relations cannot qualify. (Prove at home)

Some other types of orderings

1. A **lattice** is a poset in which every pair of elements has a least upper bound (LUB) and a greatest lower bound (GLB).

   Formally: (done in exercise)

   Example: $(Z^+, |)$ In this case,

   LUB = Least Common Multiple

   GLB = Greatest Common Denominator

   Non-example: $(\{1,2,3\}, |)$

Some other types of orderings

2. **Linearly ordered** sets (also: **totally** ordered sets): posets in which *all elements are "comparable" (i.e., related by R)*.

   Formally: $\forall x,y \in A(xRy \vee yRx)$.

   Example:

   Non-example:

Some other types of orderings

Linearly ordered sets (also: **totally** ordered sets): posets in which *all elements are comparable*. Formally:

$\forall x,y \in A(xRy \vee yRx)$.

Example: $(\mathbf{N},\leq)$

Non-example: $(\mathbf{N}, \mid)$ (where | is 'divides')

Non-example: $\subseteq$

An application of posets

- Consider (A,$\leq$), where A is a set of project tasks and a$<$b means “a must be completed before b can be completed”
- (Sometimes it’s easier
  to define $<$ than $\leq$)
- Note that (A,$\leq$) is a poset:
  ref, antisym, trans

An application of posets

- A common problem: Given $(A,\leq)$, find a *linear* order $(A,\preccurlyeq)$ that is *compatible* with $(A,\leq)$. (That is, $(A,\leq) \subseteq (A,\preccurlyeq)$)
- (We're assuming that tasks cannot be carried out in parallel)
- Algorithm for finding a compatible linear order given a finite partial order: p.526.

Some other types of orderings

2. **Well-ordered** sets: linearly ordered sets in which *every nonempty subset has a least element* (that is, an element a such that $\forall x \in A (aRx)$)

   Example: …

   Non-example: …

Some other types of orderings

2. **Well-ordered** sets: linearly ordered sets in which *every nonempty subset has a least element* (that is, an element a such that $\forall x \in A (aRx)$)

Example: (**N**,≤)
Non-examples:
(**Z**,≤) ,
(non-negative elements of **R**, ≤)

Some other types of orderings

2. Non-examples: **(Z,≤)** , **(R^+, ≤)**
   - **(Z,≤)**: **Z** itself has no least element.
   - **(Non-negative R , ≤)**:

     Nonnegative R itself does have a least element , but

     R^+ ⊆ Nonnegative R has no least element.

- Well-orderings are behind one of the most general proof techniques that exist: *mathematical induction*.

- The last 30 slides were a tiny crash course in the theory of *mathematical structures*
- Compare Rosen, chapter 7.6.