Module #18: **Relations**

Rosen 5th ed., ch. 7

~32 slides (in progress), ~2 lectures

Binary Relations

- Let A, B be any two sets.
- A *binary relation* R from A to B, written (with signature) $R{:}A \leftrightarrow B$, is a subset of $A \times B$.
  - *E.g.*, let $\mathbf{<} : \mathbf{N} \leftrightarrow \mathbf{N} :\equiv \{(n,m) \mid n < m\}$
- The notation $a\ R\ b$ or aRb means $(a,b) \in R$.
  - *E.g.*, $a \mathbf{<} b$ means $(a,b) \in \mathbf{<}$
- If aRb we may say "a is related to b (by relation R)", or "a relates to b (under relation R)".
- A binary relation R corresponds to a predicate function $P_R{:}A \times B \rightarrow \{\mathbf{T},\mathbf{F}\}$ defined over the 2 sets A,B; *e.g.*, "eats" $:\equiv \{(a,b)|$ organism a eats food $b\}$

Complementary Relations

- Let $R{:}A \leftrightarrow B$ be any binary relation.
- Then, $\not{R}{:}A \leftrightarrow B$, the *complement* of R, is the binary relation defined by

  $$\not{R} :\equiv \{(a,b) \mid (a,b) \notin R\} = (A \times B) - R$$

  Note this is just $\overline{R}$ if the universe of discourse is $U = A \times B$; thus the name *complement.*
- Note the complement of $\not{R}$ is R.

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

Inverse Relations

- Any binary relation $R{:}A{\leftrightarrow}B$ has an *inverse* relation $R^{-1}{:}B{\leftrightarrow}A$, 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→Foods is defined by
  $aRb \Leftrightarrow a$ *eats* b, then:
  $b\ R^{-1}\ a \Leftrightarrow b$ *is eaten by* a. (Passive voice.)

Relations on a Set

- A (binary) relation from a set A to itself is called a relation *on* the set A.
- *E.g.*, the "**<**" relation from earlier was defined as a relation *on* the set **N** of natural numbers.
- The *identity relation* $\mathbf{I}_A$ on a set A is the set $\{(a,a)|a \in A\}$.

Reflexivity

- A relation R on A is *reflexive* if $\forall a \in A$, aRa.
  - *E.g.*, the relation $\geq :\equiv \{(a,b) \mid a \geq b\}$ is reflexive.
- A relation is *irreflexive* iff its complementary relation is reflexive.
  - Note "*irreflexive*" ≠ "*not reflexive*"!
  - Example: $<$ is irreflexive.
  - Note: "likes" between people is not reflexive, but not irreflexive either. (Not everyone likes themselves, but not everyone dislikes themselves either.)

Symmetry & Antisymmetry

- A binary relation R on A is symmetric iff $R = R^{-1}$, that is, if $(a,b) \in R \leftrightarrow (b,a) \in R$.
  - *E.g.*, = (equality) is symmetric. < is not.
  - "is married to" is symmetric, "likes" is not.
- A binary relation R is *antisymmetric* if $(a,b) \in R \rightarrow (b,a) \notin R$.
  - < is antisymmetric, "likes" is not.

Transitivity

- A relation R is *transitive* iff (for all a,b,c)
  $$(a,b)\in R \wedge (b,c)\in R \rightarrow (a,c)\in R.$$
- A relation is *intransitive* if it is not transitive.
- Examples: "is an ancestor of" is transitive.
- "likes" is intransitive.
- "is within 1 mile of" is… ?

Totality

- A relation $R{:}A \leftrightarrow B$ is *total* if for every $a \in A$, there is at least one $b \in B$ such that $(a,b) \in R$.
- If R is not total, then it is called *strictly partial*.
- A *partial relation* is a relation that *might* be strictly partial. Or, it might be total. (In other words, all relations are considered "partial.")

Functionality

- A relation $R{:}A{\leftrightarrow}B$ is *functional* (that is, it is also a partial function $R{:}A{\rightarrow}B$) if, for any $a{\in}A$, there is *at most 1* $b{\in}B$ such that $(a,b){\in}R$.
- R is *antifunctional* if its inverse relation R^{-1} is functional.
  - Note: A functional relation (partial function) that is also antifunctional is an invertible partial function.
- R is a *total function* $R{:}A{\rightarrow}B$ if it is both functional and total, that is, for any $a{\in}A$, there is *exactly* 1 b such that $(a,b){\in}R$. If R is functional but not total, then it is a *strictly partial function*.

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:

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

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

$$R^0 :\equiv \mathbf{I}_A ; \qquad R^{n+1} :\equiv R^n \circ R \quad \text{for all } n \geq 0.$$

  - Negative powers of R can also be defined if desired, by $R^{-n} :\equiv (R^{-1})^n$.

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

Relational Databases

- A *relational database* is essentially an n-ary relation R.
- 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.
- Then, the *selection operator* s_C is the operator that maps any (n-ary) relation R on A to the n-ary relation of all n-tuples from R that satisfy C.
  - *I.e.*, $\forall R \subseteq A,\ s_C(R) = R \cap \{a \in A \mid s_C(a) = \mathbf{T}\}$

Selection Operator Example

- Suppose we have a domain
  A = StudentName × Standing × SocSecNos
- Suppose we define a certain condition on A,
  $UpperLevel(name,standing,ssn) :\equiv$
  $[(standing = \text{junior}) \vee (standing = \text{senior})]$
- Then, $s_{UpperLevel}$ is the selection operator that takes any relation R on A (database of students) and produces a relation consisting of *just* the upper-level classes (juniors and seniors).

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,
  - That is, where $1 \leq i_k \leq n$ for all $1 \leq k \leq m$.
- Then the *projection operator* on n-tuples

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

is defined by:

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

Projection Example

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

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

- This operator can be usefully applied to a whole relation $R \subseteq Cars$ (database of cars) to obtain a list of model/color combinations available.

Join Operator

- Puts two relations together to form a sort of combined relation.
- If 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, C can also be sequences of elements rather than single elements.

Join Example

- Suppose R_1 is a teaching assignment table, relating *Professors* 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 (*professor*,*course*,*room*,*time*).

§7.3: Representing Relations

- Some ways to represent *n*-ary relations:
  - With an explicit list or table of its tuples.
  - With a function from the domain to {**T**,**F**}.
    - Or with an algorithm for computing this function.
- Some special ways to represent binary relations:
  - With a zero-one matrix.
  - With a directed graph.

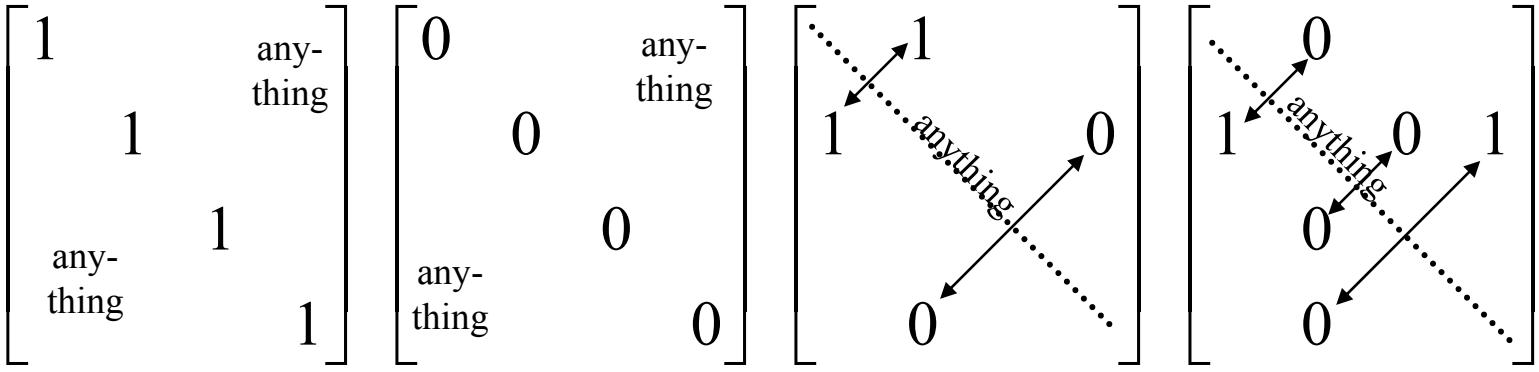
Using Zero-One Matrices

- To represent a relation R by a matrix $\mathbf{M}_R = [m_{ij}]$, let $m_{ij} = 1$ if $(a_i,b_j) \in R$, else 0.
- *E.g.*, Joe likes Susan and Mary, Fred likes Mary, and Mark likes Sally.
- The 0-1 matrix representation of that "Likes" relation:

	Susan	Mary	Sally
Joe	1	1	0
Fred	0	1	0
Mark	0	0	1

Zero-One Reflexive, Symmetric

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



Reflexive: all 1's on diagonal	*Irreflexive*: all 0's on diagonal	*Symmetric*: all identical across diagonal	*Antisymmetric*: all 1's are across from 0's

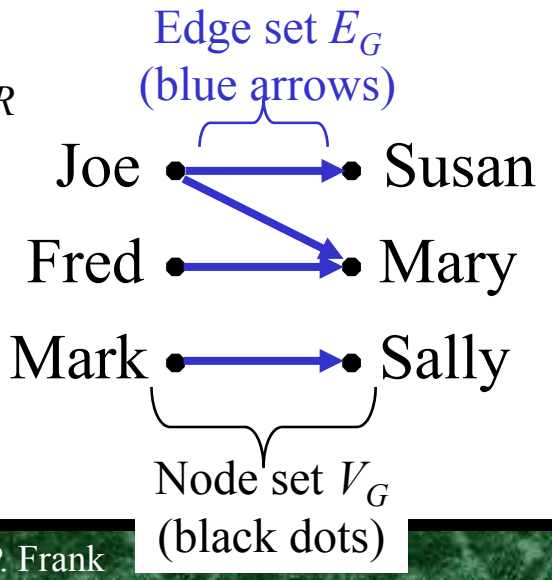
Using Directed Graphs

- 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,links)*. Visually represented using dots for nodes, and arrows for edges. Notice that a relation $R:A \leftrightarrow B$ can be represented as a graph $G_R=(V_G=A \cup B, E_G=R)$.

$\mathbf{M}_R$

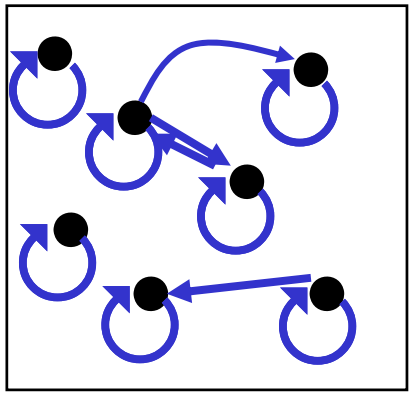
	Susan	Mary	Sally
Joe	1	1	0
Fred	0	1	0
Mark	0	0	1

G_R

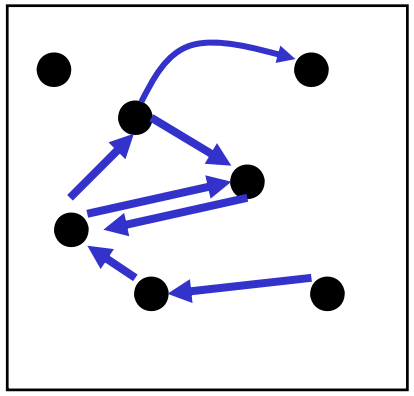


Digraph Reflexive, Symmetric

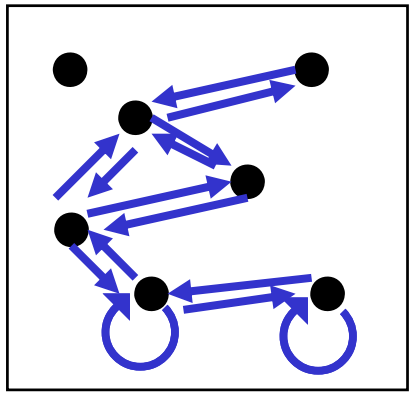
It is extremely easy to recognize the reflexive/irreflexive/ symmetric/antisymmetric properties by graph inspection.



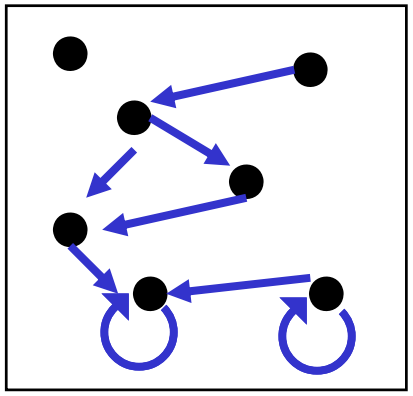
Reflexive: Every node has a self-loop



Irreflexive: No node links to itself



Symmetric: Every link is bidirectional



Antisymmetric: No link is bidirectional

Asymmetric, non-antisymmetric

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

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), …, (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}

A Faster Transitive Closure Alg.

procedure *transClosure*($\mathbf{M}_R$:rank-n 0-1 m

A := **B** := $\mathbf{M}_R$;

for $i := 1$ to $\lceil \log_2 n \rceil$ **be**

A := **A** $\odot$ **A**; {**A** represents R^{2^i} }

B := **B** $\vee$ **A** {"add" $\mathbf{M}_R^{[2^i]}$ into **B**}

end

return B {Alg. takes only $\Theta(n^3 \log n)$ time}

WARNING: Contains a Bug

Need to Fix

Roy-Warshall Algorithm

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

Procedure *Warshall*($\mathbf{M}_R$: rank-n 0-1 matrix)

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

- An *equivalence relation* (e.r.) on a set A is simply any binary relation on A that is reflexive, symmetric, and transitive.
  - *E.g.*, = itself is an equivalence relation.
  - For any function $f{:}A \to B$, 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 e.r.

Equivalence Relation Examples

- "Strings a and b are the same length."
- "Integers a and b have the same absolute value."
- "Real numbers a and b have the same fractional part (*i.e.*, $a - b \in \mathbf{Z}$)."
- "Integers a and b have the same residue modulo m." (for a given $m>1$)

Equivalence Classes

- Let R be any equiv. rel. on a set A.
- The *equivalence class* of a,
  $[a]_R :\equiv \{ b \mid aRb \}$ (optional subscript R)
  - It is the set of all elements of A that are "equivalent" to a according to the eq.rel. R.
  - Each such b (including a itself) is called a *representative* of $[a]_R$.
- Since $f(a)=[a]_R$ is a function of a, any equivalence relation R be defined using $aRb :\equiv$ "a and b have the same f value", given that f.

Equivalence Class Examples

- "Strings a and b are the same length."
  - $[a]$ = the set of all strings of the same length as a.
- "Integers a and b have the same absolute value."
  - $[a]$ = the set $\{a, -a\}$
- "Real numbers a and b have the same fractional part (*i.e.*, $a - b \in \mathbf{Z}$)."
  - $[a]$ = the set $\{\ldots, a-2, a-1, a, a+1, a+2, \ldots\}$
- "Integers a and b have the same residue modulo m." (for a given $m>1$)
  - $[a]$ = the set $\{\ldots, a-2m, a-m, a, a+m, a+2m, \ldots\}$

Partitions

- A *partition* of a set A is the set of all the equivalence classes $\{A_1, A_2, \ldots\}$ for some e.r. on A.
- The A_i's are all disjoint and their union = A.
- They "partition" the set into pieces. Within each piece, all members of the set are equivalent to each other.

§7.6: Partial Orderings

- Not sure yet if there will be time to cover this section.