UNIVERSITY AT ALBANY

State University of New York

Lecture 10: Relations

Dr. Chengjiang Long

Computer Vision Researcher at Kitware Inc.

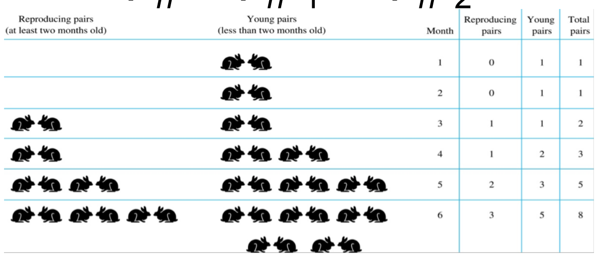
Adjunct Professor at SUNY at Albany.

Email: **clong2@albany.edu**

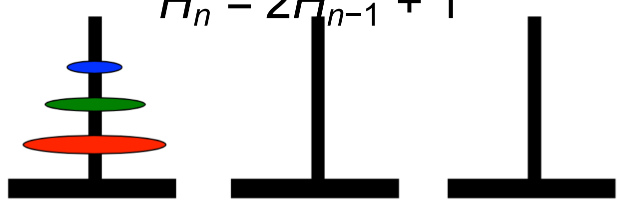
Recap Previous Lecture

- Applications of Recurrence Relations
- Solving Linear Recurrence Relations

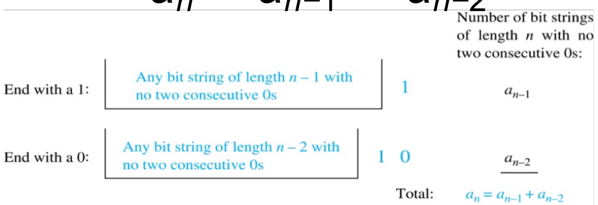
$F_n = F_{n-1} + F_{n-2}$



$H_n = 2H_{n-1} + 1$



$a_n = a_{n-1} + a_{n-2}$



characteristic equation

$r^k - c_1 r^{k-1} - c_2 r^{k-2} - \cdots - c_{k-1} r - c_k = 0$

characteristic roots

(1) k distinct roots $r_1, r_2, \ldots, r_k$.

$$a_n = \alpha_1 r_1^n + \alpha_2 r_2^m + \cdots + \alpha_k r_k^n$$

(2) has t distinct roots $r_1, r_2, \ldots, r_t$ with multiplicities $m_1, m_2, \ldots, m_t$, respectively so that $m_i \geq 1$ for $i = 0, 1, 2, \ldots, t$ and $m_1 + m_2 + \ldots + m_t = k$

$$a_n = (\alpha_{1,0} + \alpha_{1,1} n + \cdots + \alpha_{1,m_1-1} n^{m_1-1}) r_1^n + (\alpha_{2,0} + \alpha_{2,1} n + \cdots + \alpha_{2,m_2-1} n^{m_2-1}) r_2^n + \cdots + (\alpha_{t,0} + \alpha_{t,1} n + \cdots + \alpha_{t,m_t-1} n^{m_t-1}) r_t^n$$

Recap Previous Lecture

- Divide-and-Conquer Algorithms and Recurrence Relations
- Generating Functions

$f(n) = af(n/b) + g(n)$

$f(n) = af(n/b) + c$

$$f(n) \text{ is } \begin{cases} O(n^{\log_b a}) & \text{if } \quad a > 1 \\ O(\log n) & \text{if } \quad a = 1. \end{cases}$$

$f(n) = af(n/b) + cn^d$

$$f(n) \text{ is } \begin{cases} O(n^d) & \text{if } \quad a < b^d, \\ O(n^d \log n) & \text{if } \quad a = b^d, \\ O(n^{\log_b a}) & \text{if } \quad a > b^d. \end{cases}$$

$G(x) = a_0 + a_1 x + \ldots + a_k x^k + \ldots = \Sigma_{k=0}^{\infty} a_k x^k$

$$\begin{aligned}(1+x)^n &= \sum_{k=0}^{n} C(n,k)x^k \\ &= 1 + C(n,1)x + C(n,2)x^2 + \cdots + x^n\end{aligned}$$

$$\frac{1}{1-x} = \sum_{k=0}^{\infty} x^k = 1 + x + x^2 + \cdots$$

$$\frac{1}{1-ax} = \sum_{k=0}^{\infty} a^k x^k = 1 + ax + a^2x^2 + \cdots$$

Outline

- Relations
- Test and Final Project

Relations

Relations

- If we want to describe a relationship between elements of two sets A and B, we can use **ordered pairs** with their first element taken from A and their second element taken from B.
- Since this is a relation between **two sets**, it is called a **binary relation**.

- **Definition:** Let A and B be sets. A binary relation from A to B is a subset of $A \times B$.

- In other words, for a binary relation R we have $R \subseteq A \times B$. We use the notation aRb to denote that $(a, b) \in R$ and $a\underline{R}b$ to denote that $(a, b) \notin R$.

Relations

- When (a, b) belongs to R, a is said to be **related** to b by R.
- **Example:** Let P be a set of people, C be a set of cars, and D be the relation describing which person drives which car(s).
- P = {Carl, Suzanne, Peter, Carla},
- C = {Mercedes, BMW, tricycle}
- D = {(Carl, Mercedes), (Suzanne, Mercedes), (Suzanne, BMW), (Peter, tricycle)}
- This means that Carl drives a Mercedes, Suzanne drives a Mercedes and a BMW, Peter drives a tricycle, and Carla does not drive any of these vehicles.

Functions as Relations

- You might remember that a **function** f from a set A to a set B assigns a unique element of B to each element of A.
- The **graph** of f is the set of ordered pairs (a, b) such that b = f(a).
- Since the graph of f is a subset of $A \times B$, it is a **relation** from A to B.
- Moreover, for each element **a** of A, there is exactly one ordered pair in the graph that has **a** as its first element.

Functions as Relations

- Conversely, if R is a relation from A to B such that every element in A is the first element of exactly one ordered pair of R, then a function can be defined with R as its graph.

- This is done by assigning to an element $a \in A$ the unique element $b \in B$ such that $(a, b) \in R$.

Relations on a Set

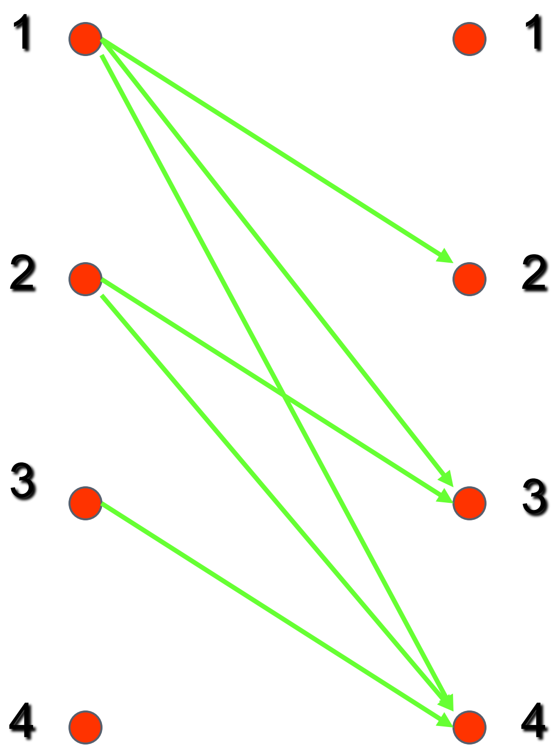
- **Definition:** A relation on the set A is a relation from A to A.

- In other words, a relation on the set A is a subset of $A \times A$.

- **Example:** Let A = {1, 2, 3, 4}. Which ordered pairs are in the relation R = {(a, b) | a < b} ?

Relations on a Set

- **Solution:** R = {(1, 2), (1, 3), (1, 4), (2, 3), (2, 4),}



R	1	2	3	4
1		x	x	x
2			x	x
3				x
4				

Relations on a Set

- **How many different relations can we define on a set A with n elements?**

- A relation on a set A is a subset of A×A.
- How many elements are in A×A ?

- There are n^2 elements in A×A, so how many subsets (= relations on A) does A×A have?

- The number of subsets that we can form out of a set with m elements is 2^m. Therefore, 2^{n^2} subsets can be formed out of A×A.

- **Answer:** We can define 2^{n^2} different relations on A.

Properties of Relations

- We will now look at some useful ways to classify relations.
- **Definition:** A relation R on a set A is called **reflexive** if $(a, a) \in R$ for every element $a \in A$.
- Are the following relations on {1, 2, 3, 4} reflexive?

R = {(1, 1), (1, 2), (2, 3), (3, 3), (4, 4)} No.

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

R = {(1, 1), (2, 2), (3, 3)} No.

Definition: A relation on a set A is called **irreflexive** if $(a, a) \notin R$ for every element $a \in A$.

Properties of Relations

- **Definitions:**

- A relation R on a set A is called **symmetric** if $(b, a) \in R$ whenever $(a, b) \in R$ for all $a, b \in A$.

- A relation R on a set A is called **antisymmetric** if $a = b$ whenever $(a, b) \in R$ and $(b, a) \in R$.

- A relation R on a set A is called **asymmetric** if $(a, b) \in R$ implies that $(b, a) \notin R$ for all $a, b \in A$.

Properties of Relations

- Are the following relations on {1, 2, 3, 4} symmetric, antisymmetric, or asymmetric?

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

R = {(1, 1)} sym. and antisym.

R = {(1, 3), (3, 2), (2, 1)} asym.

R = {(4, 4), (3, 3), (1, 4)} antisym.

Properties of Relations

- **Definition:** A relation R on a set A is called **transitive** if whenever $(a, b) \in R$ and $(b, c) \in R$, then $(a, c) \in R$ for $a, b, c \in A$.

- Are the following relations on {1, 2, 3, 4} transitive?

R = {(1, 1), (1, 2), (2, 2), (2, 1), (3, 3)} Yes.

R = {(1, 3), (3, 2), (2, 1)} No.

R = {(2, 4), (4, 3), (2, 3), (4, 1)} No.

Counting Relations

- **Example:** How many different reflexive relations can be defined on a set A containing n elements?

- **Solution:** Relations on R are subsets of A×A, which contains n^2 elements.
- Therefore, different relations on A can be generated by choosing different subsets out of these n^2 elements, so there are 2^{n^2} relations.
- A **reflexive** relation, however, **must** contain the n elements (a, a) for every $a \in A$.
- Consequently, we can only choose among $n^2 – n = n(n – 1)$ elements to generate reflexive relations, so there are $2^{n(n – 1)}$ of them.

Counting Relations

- Relations are sets, and therefore, we can apply the usual **set operations** to them.

- If we have two relations R_1 and R_2, and both of them are from a set A to a set B, then we can combine them to $R_1 \cup R_2$, $R_1 \cap R_2$, or $R_1 - R_2$.

- In each case, the result will be **another relation from A to B**.

Counting Relations

- **Definition:** Let R be a relation from a set A to a set B and S a relation from B to a set C.
- The **composite** of R and S is the relation consisting of ordered pairs (a, c), where $a \in A$, $c \in C$, and for which there exists an element $b \in B$ such that $(a, b) \in R$ and $(b, c) \in S$. We denote the composite of R and S by **$S \circ R$**.

- In other words, if relation R contains a pair (a, b) and relation S contains a pair (b, c), then $S \circ R$ contains a pair (a, c).

Counting Relations

- **Example:** Let D and S be relations on A = {1, 2, 3, 4}.
- D = {(a, b) | b = 5 - a}   "b equals (5 – a)"
- S = {(a, b) | a < b}   "a is smaller than b"

- D = {(1, 4), (2, 3), (3, 2), (4, 1)}
- S = {(1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4)}
- S∘D = {(2,4), (3,3), (3,4), (4,2), (4,3), (4,4)}

D maps an element a to the element (5 – a), and afterwards S maps (5 – a) to all elements larger than (5 – a), resulting in **S∘D = {(a,b) | b > 5 – a}** or **S∘D = {(a,b) | a + b > 5}.**

Combining Relations

- We already know that **functions** are just **special cases** of **relations** (namely those that map each element in the domain onto exactly one element in the codomain).

- If we formally convert two functions into relations, that is, write them down as sets of ordered pairs, the composite of these relations will be exactly the same as the composite of the functions (as defined earlier).

Combining Relations

- **Definition:** Let R be a relation on the set A. The powers R^n, n = 1, 2, 3, …, are defined inductively by
- $R^1 = R$
- $R^{n+1} = R^n \circ R$

- In other words:
- $R^n = R \circ R \circ \ldots \circ R$ (n times the letter R)

Combining Relations

- **Theorem:** The relation R on a set A is transitive if and only if $R^n \subseteq R$ for all positive integers n.
- **Remember the definition of transitivity:**
- **Definition:** A relation R on a set A is called transitive if whenever (a, b)$\in$R and (b, c)$\in$R, then (a, c)$\in$R for a, b, c$\in$A.
- The composite of R with itself contains exactly these pairs (a, c).
- Therefore, for a transitive relation R, $R \circ R$ does not contain any pairs that are not in R, so $R \circ R \subseteq R$.
- Since $R \circ R$ does not introduce any pairs that are not already in R, it must also be true that $(R \circ R) \circ R \subseteq R$, and so on, so that $R^n \subseteq R$.

n-ary Relations

- In order to study an interesting application of relations, namely **databases**, we first need to generalize the concept of binary relations to **n-ary relations**.

- **Definition:** Let $A_1, A_2, \ldots, A_n$ be sets. An **n-ary relation** on these sets is a subset of $A_1 \times A_2 \times \ldots \times A_n$.
- The sets $A_1, A_2, \ldots, A_n$ are called the **domains** of the relation, and n is called its **degree**.

n-ary Relations

- **Example:**
- Let R = {(a, b, c) | a = 2b $\wedge$ b = 2c with a, b, c$\in$**N**}
- What is the degree of R?
- The degree of R is 3, so its elements are triples.
- What are its domains?
- Its domains are all equal to the set of integers.
- Is (2, 4, 8) in R?
- No.
- Is (4, 2, 1) in R?
- Yes.

Representing Relations Using Matrices

- We already know different ways of representing relations. We will now take a closer look at two ways of representation: **Zero-one matrices** and **directed graphs**.
- If R is a relation from $A = \{a_1, a_2, \ldots, a_m\}$ to $B = \{b_1, b_2, \ldots, b_n\}$, then R can be represented by the zero-one matrix $M_R = [m_{ij}]$ with
- $m_{ij} = 1$, if $(a_i, b_j) \in R$, and
- $m_{ij} = 0$, if $(a_i, b_j) \notin R$.
- Note that for creating this matrix we first need to list the elements in A and B in a **particular, but arbitrary order**.

Representing Relations Using Matrices

- **Example:** How can we represent the relation $R = \{(2, 1), (3, 1), (3, 2)\}$ as a zero-one matrix?

- **Solution:** The matrix M_R is given by

$$M_R = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}$$

Representing Relations Using Matrices

- What do we know about the matrices representing a **relation on a set** (a relation from A to A) ?
- They are **square** matrices.
- What do we know about matrices representing **reflexive** relations?
- All the elements on the **diagonal** of such matrices M_{ref} must be **1s**.

$$M_{ref} = \begin{bmatrix} 1 & & & & \\ & 1 & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot & \\ & & & & & 1 \end{bmatrix}$$

Representing Relations Using Matrices

- What do we know about the matrices representing **symmetric relations**?
- These matrices are symmetric, that is, $M_R = (M_R)^t$.

$$M_R = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

symmetric matrix,
symmetric relation.

$$M_R = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

non-symmetric matrix,
non-symmetric relation.

Representing Relations Using Matrices

- The Boolean operations **join** and **meet** **(you remember?)** can be used to determine the matrices representing the **union** and the **intersection** of two relations, respectively.

- To obtain the **join** of two zero-one matrices, we apply the Boolean "or" function to all corresponding elements in the matrices.

- To obtain the **meet** of two zero-one matrices, we apply the Boolean "and" function to all corresponding elements in the matrices.

Representing Relations Using Matrices

- **Example:** Let the relations R and S be represented by the matrices

$$M_R = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \qquad M_S = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

What are the matrices representing R∪S and R∩S?
Solution: These matrices are given by

$$M_{R\cup S} = M_R \vee M_S = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix} \qquad M_{R\cap S} = M_R \wedge M_S = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Representing Relations Using Matrices

- Do you remember the **Boolean product** of two zero-one matrices?
- Let $A = [a_{ij}]$ be an $m \times k$ zero-one matrix and $B = [b_{ij}]$ be a $k \times n$ zero-one matrix.
- Then the **Boolean product** of A and B, denoted by $A \circ B$, is the $m \times n$ matrix with (i, j)th entry $[c_{ij}]$, where
- $c_{ij} = (a_{i1} \wedge b_{1j}) \vee (a_{i2} \wedge b_{2i}) \vee \ldots \vee (a_{ik} \wedge b_{kj})$.
- $c_{ij} = 1$ if and only if at least one of the terms $(a_{in} \wedge b_{nj}) = 1$ for some n; otherwise $c_{ij} = 0$.

Representing Relations Using Matrices

- Let us now assume that the zero-one matrices $M_A = [a_{ij}]$, $M_B = [b_{ij}]$ and $M_C = [c_{ij}]$ represent relations A, B, and C, respectively.
- **Remember:** For $M_C = M_A \odot M_B$ we have:
- $c_{ij} = 1$ if and only if at least one of the terms $(a_{in} \wedge b_{nj}) = 1$ for some n; otherwise $c_{ij} = 0$.
- In terms of the **relations**, this means that C contains a pair (x_i, z_j) if and only if there is an element y_n such that (x_i, y_n) is in relation A and (y_n, z_j) is in relation B.
- Therefore, $C = B \circ A$ (**composite** of A and B).

Representing Relations Using Matrices

- This gives us the following rule:
- $M_{B \circ A} = M_A \circ M_B$
- In other words, the matrix representing the **composite** of relations A and B is the **Boolean product** of the matrices representing A and B.

- Analogously, we can find matrices representing the **powers of relations**:
- $M_{R^n} = M_R^{[n]}$ (n-th **Boolean power**).

Representing Relations Using Matrices

- **Example:** Find the matrix representing R^2, where the matrix representing R is given by

$$M_R = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

Solution: The matrix for R^2 is given by

$$M_{R^2} = M_R^{[2]} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

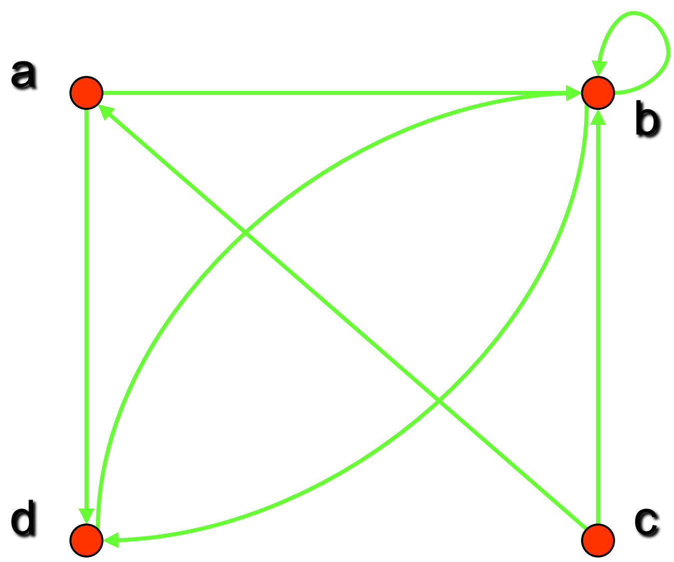
Representing Relations Using Digraphs

- **Definition:** A **directed graph**, or **digraph**, consists of a set V of **vertices** (or **nodes**) together with a set E of ordered pairs of elements of V called **edges** (or **arcs**).
- The vertex a is called the **initial vertex** of the edge (a, b), and the vertex b is called the **terminal vertex** of this edge.

- We can use arrows to display graphs.

Representing Relations Using Digraphs

- **Example:** Display the digraph with V = {a, b, c, d},
E = {(a, b), (a, d), (b, b), (b, d), (c, a), (c, b), (d, b)}.



An edge of the form (b, b) is called a **loop.**

Representing Relations Using Digraphs

- Obviously, we can represent any relation R on a set A by the digraph with A as its vertices and all pairs $(a, b) \in R$ as its edges.

- Vice versa, any digraph with vertices V and edges E can be represented by a relation on V containing all the pairs in E.

- This **one-to-one correspondence** between relations and digraphs means that any statement about relations also applies to digraphs, and vice versa.

Equivalence Relations

- **Equivalence relations** are used to relate objects that are similar in some way.
- **Definition:** A relation on a set A is called an equivalence relation if it is reflexive, symmetric, and transitive.
- Two elements that are related by an equivalence relation R are called **equivalent**.

Equivalence Relations

- Since R is **symmetric**, a is equivalent to b whenever b is equivalent to a.
- Since R is **reflexive**, every element is equivalent to itself.
- Since R is **transitive**, if a and b are equivalent and b and c are equivalent, then a and c are equivalent.

- Obviously, these three properties are necessary for a reasonable definition of equivalence.

Equivalence Relations

- **Example:** Suppose that R is the relation on the set of strings that consist of English letters such that aRb if and only if l(a) = l(b), where l(x) is the length of the string x. Is R an equivalence relation?
- **Solution:**
  - R is reflexive, because l(a) = l(a) and therefore aRa for any string a.
  - R is symmetric, because if l(a) = l(b) then l(b) = l(a), so if aRb then bRa.
  - R is transitive, because if l(a) = l(b) and l(b) = l(c), then l(a) = l(c), so aRb and bRc implies aRc.
- **R is an equivalence relation.**

Equivalence Class

- **Definition:** Let R be an equivalence relation on a set A. The set of all elements that are related to an element *a* of A is called the **equivalence class** of *a*.
- The equivalence class of *a* with respect to R is denoted by **$[a]_R$**.
- When only one relation is under consideration, we will delete the subscript R and write **$[a]$** for this equivalence class.
- If $b \in [a]_R$, b is called a **representative** of this equivalence class.

Example

- In the previous example (strings of identical length), what is the equivalence class of the word mouse, denoted by [mouse] ?

- **Solution:** [mouse] is the set of all English words containing five letters.

- For example, ‘horse’ would be a representative of this equivalence class.

Equivalence Classes

- **Theorem:** Let R be an equivalence relation on a set A. The following statements are equivalent:
  - aRb
  - $[a] = [b]$
  - $[a] \cap [b] \neq \emptyset$
- A **partition** of a set S is a collection of disjoint nonempty subsets of S that have S as their union. In other words, the collection of subsets A_i, $i \in I$, forms a partition of S if and only if
  - $A_i \neq \emptyset$ for $i \in I$
  - $A_i \cap A_j = \emptyset$, if $i \neq j$
  - $\cup_{i \in I} A_i = S$

Example

- Let S be the set {u, m, b, r, o, c, k, s}.
  Do the following collections of sets partition S ?

{{m, o, c, k}, {r, u, b, s}}	yes.
{{c, o, m, b}, {u, s}, {r}}	no (k is missing).
{{b, r, o, c, k}, {m, u, s, t}}	no (t is not in S).
{{u, m, b, r, o, c, k, s}}	yes.
{∅, {b, o, k}, {r, u, m}, {c, s}}	no (∅ not allowed).

Equivalence Classes

- **Theorem:** Let R be an equivalence relation on a set S. Then the **equivalence classes** of R form a **partition** of S. Conversely, given a partition $\{A_i \mid i \in I\}$ of the set S, there is an equivalence relation R that has the sets A_i, $i \in I$, as its equivalence classes.

Example

- Let R be the relation $\{(a, b) \mid a \equiv b \pmod 3\}$ on the set of integers. Is R an equivalence relation?
- **Yes, R is reflexive, symmetric, and transitive.**

- What are the equivalence classes of R ?
- {{…, -6, -3, 0, 3, 6, …},
 {…, -5, -2, 1, 4, 7, …},
 {…, -4, -1, 2, 5, 8, …}}

Databases and Relations

- Consider a relational database of students, whose records are represented as 4-tuples with the fields **Student Name**, **ID Number**, **Major**, and **GPA**:

- R = {(Ackermann, 231455, CS, 3.88),
    (Adams, 888323, Physics, 3.45),
    (Chou, 102147, CS, 3.79),
    (Goodfriend, 453876, Math, 3.45),
    (Rao, 678543, Math, 3.90),
    (Stevens, 786576, Psych, 2.99)}

- Relations that represent databases are also called **tables**, since they are often displayed as tables.

Databases and Relations

- A domain of an n-ary relation is called a **primary key** if the n-tuples are uniquely determined by their values from this domain.
- This means that no two records have the same value from the same primary key.
- In our example, which of the fields **Student Name**, **ID Number**, **Major**, and **GPA** are primary keys?
- **Student Name** and **ID Number** are primary keys, because no two students have identical values in these fields.
- In a real student database, only **ID Number** would be a primary key.

Databases and Relations

- We can apply a variety of **operations** on n-ary relations to form new relations.
- **Definition:** The **projection** $P_{i_1, i_2, \ldots, i_m}$ maps the n-tuple $(a_1, a_2, \ldots, a_n)$ to the m-tuple $(a_{i_1}, a_{i_2}, \ldots, a_{i_m})$, where $m \leq n$.
- In other words, a projection $P_{i_1, i_2, \ldots, i_m}$ keeps the m components $a_{i_1}, a_{i_2}, \ldots, a_{i_m}$ of an n-tuple and deletes its (n – m) other components.
- **Example:** What is the result when we apply the projection $P_{2,4}$ to the student record (Stevens, 786576, Psych, 2.99) ?
- **Solution:** It is the pair (786576, 2.99).

Databases and Relations

- We can use the **join** operation to combine two tables into one if they share some identical fields.

- **Definition:** Let R be a relation of degree m and S a relation of degree n. The **join** $J_p(R, S)$, where $p \leq m$ and $p \leq n$, is a relation of degree $m + n - p$ that consists of all $(m + n - p)$-tuples
$(a_1, a_2, \ldots, a_{m-p}, c_1, c_2, \ldots, c_p, b_1, b_2, \ldots, b_{n-p})$,
where the m-tuple $(a_1, a_2, \ldots, a_{m-p}, c_1, c_2, \ldots, c_p)$ belongs to R and the n-tuple $(c_1, c_2, \ldots, c_p, b_1, b_2, \ldots, b_{n-p})$ belongs to S.

Databases and Relations

- **Example:** What is $J_1(Y, R)$, where Y contains the fields **Student Name** and **Year of Birth**,
- Y = {(1978, Ackermann),
  (1972, Adams),
  (1917, Chou),
  (1984, Goodfriend),
  (1982, Rao),
  (1970, Stevens)},
- and R contains the student records as defined before ?

Databases and Relations

- **Solution:** The resulting relation is:
- {(1978, Ackermann, 231455, CS, 3.88),
(1972, Adams, 888323, Physics, 3.45),
(1917, Chou, 102147, CS, 3.79),
(1984, Goodfriend, 453876, Math, 3.45),
(1982, Rao, 678543, Math, 3.90),
(1970, Stevens, 786576, Psych, 2.99)}

- Since Y has two fields and R has four, the relation J_1(Y, R) has 2 + 4 – 1 = 5 fields.

Test and Final Project

Test

- This is a OPEN BOOK & OPEN NOTE exam. Also, you cannot access the Internet or use your laptop computer. Do the exam independently.
- There are a total of 100 points in the exam. Plan your work accordingly.
- Write out the steps for all problems to receive the full credit. Use additional pages if necessary.

Problem	Points	Scores
Problem 1: True or False	10	
Problem 2: Logic and Proofs	8	
Problem 3: Functions, Sequences and Sums	15	
Problem 4: Algorithm Complexity	7	
Problem 5: Integer Representations and Modular Arithmetic	15	
Problem 6: Induction and Recursion	10	
Problem 7: Counting and Discrete Probability	15	
Problem 8: Relations	20	

About Final Project

- **Research investigations and determine a topic**
  Identify publications and research that uses some discrete math terminologies.
- Make interesting analogies and application of investigation to your current research.
- e.g., Graph theory, numerical computation, proofs by induction.

About Final Project

- **Final Project**
  Do some research on your project, related work, preliminary results. Related work.
- Conclude project with new ideas, formalizations, out of the box thinking.
- Implement the solutions.
- Submit complete project in IEEE conference proceeding format.
- Present your work in class during the last week of class (May 2). Presentation should be about 10 minutes/person. Hence a group of 2-3 people should give a 20-25 minute presentation.

1. Cryptography

- Encryption and decryption are part of **cryptography**, which is part of discrete mathematics. For example, secure internet shopping uses public-key cryptography.

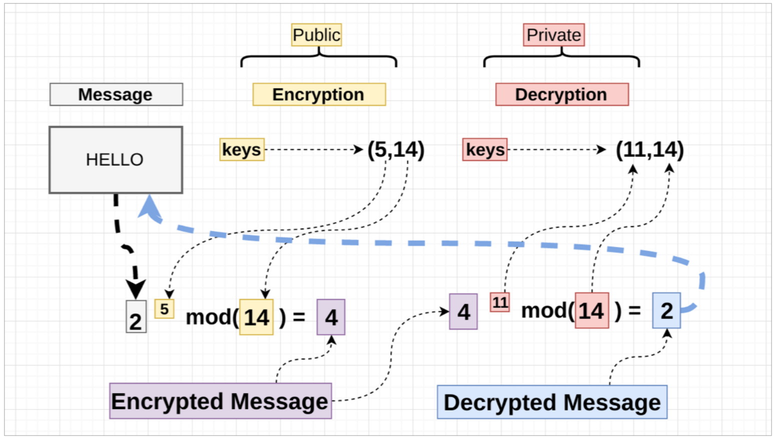
- Related reading:
- ***Internet shopping: stopping the scammers***
- Link: https://ima.org.uk/779/internet-shopping-stopping-the-scammers/

2.RSA Cryptosystem

- The goal of this project is to familiarize you with the RSA encryption algorithm through programming. You may work individually or in pairs on this project.

- Write a program to implement the RSA algorithm for cryptography.

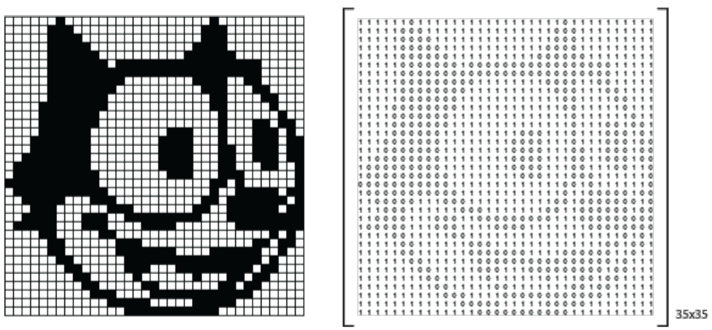
- Related reading:
- **RSA Cryptography**
- Link: http://www.cs.kzoo.edu/math250/rsa_project.html



3.Digital image processing

- **Digital image processing** uses discrete mathematics to merge images or apply filters.

- Related reading:
- ***Matrices and Digital Images***
- Link: http://blog.kleinproject.org/?p=588



The matrix corresponding to Felix The Cat

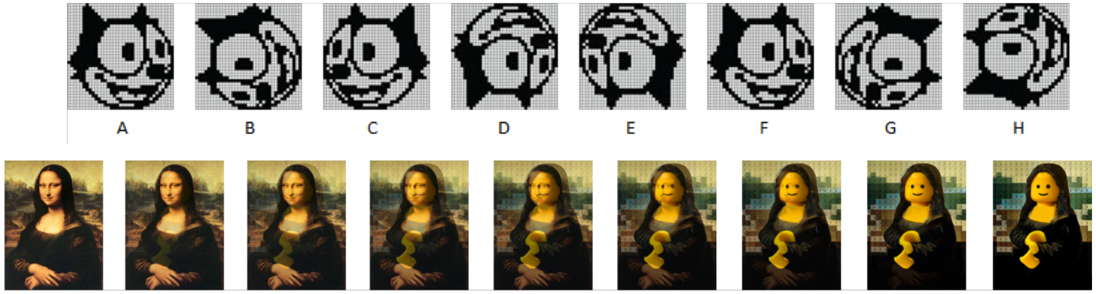
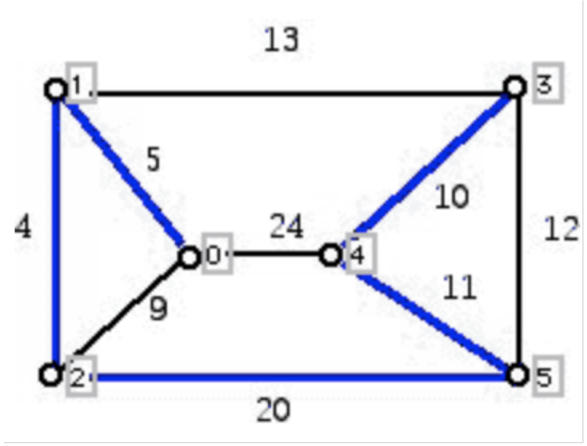


Image processing

4.Minimum cost spanning trees

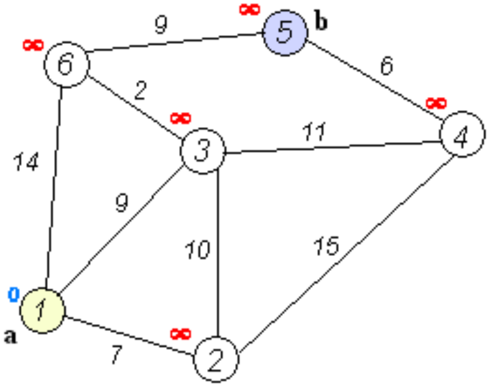
- **Wiring a computer network** using the least amount of cable is a minimum-weight spanning tree problem.

- Related reading:
- Trees: A Mathematical Tool for All Seasons
- Link: http://www.ams.org/publicoutreach/feature-column/fcarc-trees



5.Google Maps

- **Google Maps** uses discrete mathematics to determine fastest driving routes and times. There is a simpler version that works with small maps and technicalities involved in adapting to large maps.
- Related reading
- ***The Simple, Elegant Algorithm That Makes Google Maps Possible***
- Link: https://motherboard.vice.com/en_us/article/4x3pp9/the-simple-elegant-algorithm-that-makes-google-maps-possible



6. Bayesian Network Classifiers

- This project will explore going beyond the Gaussian Naïve Bayes classifier by training a Bayes network.

- Related reading
- ***Bayesian Network Classifiers***
- Link: http://www.cs.technion.ac.il/~dang/journal_papers/friedman1997Bayesian.pdf

Next class

- Topic: Graphs
- Pre-class reading: Chap 10