Databases and Relations

Let us take a look at a type of database representation that is based on relations, namely the **relational data model.**

A database consists of n-tuples called **records**, which are made up of **fields**.

These fields are the **entries** of the n-tuples.

The relational data model represents a database as an n-ary relation, that is, a set of records.

Databases and Relations

Example: Consider a 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

In a database, a primary key should remain one even if new records are added.

Therefore, we should use a primary key of the **intension** of the database, which contains all the n-tuples that can ever be included in our database.

Combinations of domains can also uniquely identify n-tuples in an n-ary relation.

When the values of a **set of domains** determine an n-tuple in a relation, the **Cartesian product** of these domains is called a **composite 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

In some cases, applying a projection to an entire table may not only result in fewer columns, but also in **fewer rows**.

Why is that?

Some records may only have differed in those fields that were deleted, so they become **identical**, and there is no need to list identical records more than once.

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

In other words, to generate $J_p(R, S)$, we have to find all the elements in R whose p last components match the p first components of an element in S.

The new relation contains exactly these matches, which are combined to tuples that contain each matching field only once.

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.

Representing Relations

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

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

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

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

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

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 $R \cup S$ and $R \cap S$?

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 $A \odot B$, is the $m \times n$ matrix with (i, j)th entry $[c_{ij}]$, where

$c_{ij} = (a_{i1} \wedge b_{1j}) \vee (a_{i2} \wedge b_{2j}) \vee \ldots \vee (a_{ik} \wedge b_{kj})$.

$c_{ij} = 1$ 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:

$M_{B \circ A} = M_A \odot M_B$

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

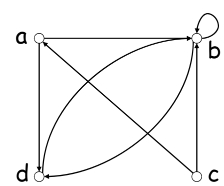
Definition: A **directed graph**, or **digraph**, consists of a set V of **vertices** (or **nodes**) together with a set E of ordered pairs of elements of V called **edges** (or **arcs**).

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)∈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.

Closures of Relations (section 9.4)

What is the **closure** of a relation? (sec. 9.4)

Definition: Let R be a relation on a set A. R may or may not have some **property P**, such as reflexivity, symmetry, or transitivity.

If there is a relation S with property P containing R such that S is a subset of every relation with property P containing R, then S is called the closure of R with respect to P.

Note that the closure of a relation with respect to a property may not exist.

Closures of Relations

If the closure of a relation R under a property P exists then this closure is the intersection of all relations with property P containing R.

The proof of this important fact is exercise 14, p. 607 (exercise 14, p. 554, 6th ed.).

Closures of Relations

An example where the closure might not exist is the property "the relation is infinite".

Any finite relation R on Z x Z, e.g. R = {(0,1), (0, 10)} does not have a closure under this infinite property.

No infinite relation containing R is contained in every infinite relation containing R.

Closures of Relations

Example I: Find the **reflexive closure** of relation R = {(1, 1), (1, 2), (2, 1), (3, 2)} on the set A = {1, 2, 3}.

Solution: We know that any reflexive relation on A must contain the elements (1, 1), (2, 2), and (3, 3).

By adding (2, 2) and (3, 3) to R, we obtain the reflexive relation S, which is given by
S = {(1, 1), (1, 2), (2, 1), (2, 2), (3, 2), (3, 3)}.

S is reflexive, contains R, and is contained within every reflexive relation that contains R.

Therefore, S is the **reflexive closure** of R.

Closures of Relations

Example II: Find the **symmetric closure** of the relation R = {(a, b) | a > b} on the set of positive integers.

Solution: The symmetric closure of R is given by

$R \cup R^{-1} = \{(a, b) \mid a > b\} \cup \{(b, a) \mid a > b\}$

$= \{(a, b) \mid a \neq b\}$

Closures of Relations

Example III: Find the **transitive closure** of the relation R = {(1, 3), (1, 4), (2, 1), (3, 2)} on the set A = {1, 2, 3, 4}.

Solution: R would be transitive, if for all pairs (a, b) and (b, c) in R there were also a pair (a, c) in R.

If we add the missing pairs (1, 2), (2, 3), (2, 4), and (3, 1), will R be transitive?

No, because the extended relation R contains (3, 1) and (1, 4), but does not contain (3, 4).

By adding new elements to R, we also add **new requirements** for its transitivity. We need to look at **paths in digraphs** to solve this problem.

Closures of Relations

Imagine that we have a relation R that represents all **train connections** in the US.

For example, if (Boston, Philadelphia) is in R, then there is a **direct** train connection from Boston to Philadelphia.

If R contains (Boston, Philadelphia) and (Philadelphia, Washington), there is an **indirect** connection from Boston to Washington.

Because there are indirect connections, it is not possible by just looking at R to determine which cities are connected by trains.

The transitive closure of R contains exactly those pairs of cities that are connected, **either directly or indirectly**.

Graphs and Relations

Definition: A **path** from a to b in the directed graph G is a sequence of one or more edges (x_0, x_1), (x_1, x_2), (x_2, x_3), …, (x_{n-1}, x_n) in G, where x_0 = a and x_n = b.

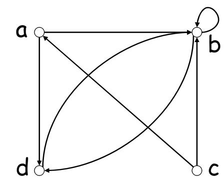
In other words, a path is a **sequence of edges** where the terminal vertex of an edge is the same as the initial vertex of the next edge in the path.

This path is denoted by $x_0, x_1, x_2, \ldots, x_n$ and has **length** n.

A path that begins and ends at the same vertex is called a **circuit** or **cycle**.

Graphs and Relations

Example: Let us take a look at the following graph:



Is **c,a,b,d,b** a path in this graph? Yes.

Is **d,b,b,b,d,b,d** a circuit in this graph? Yes.

Is there any circuit including **c** in this graph? No.

Graphs and Relations

Due to the one-to-one correspondence between graphs and relations, we can transfer the definition of path from graphs to relations:

Definition: There is a path from a to b in a relation R, if there is a sequence of elements a, x_1, x_2, …, x_{n-1}, b with $(a, x_1) \in R$, $(x_1, x_2) \in R$, …, and $(x_{n-1}, b) \in R$.

Theorem: Let R be a relation on a set A. There is a path from a to b of length n if and only if $(a, b) \in R^n$.

Closures of Relations

According to the train example, the transitive closure of a relation consists of the pairs of vertices in the associated directed graph that are connected by a path.

Definition: Let R be a relation on a set A. The connectivity relation R* consists of the pairs (a, b) such that there is a path between a and b in R.

We know that R^n consists of the pairs (a, b) such that a and b are connected by a path of length n.

Therefore, R* is the union of R^n over all positive integers n:

$$R^* = \bigcup_{n=1}^{\infty} R^n = R^1 \cup R^2 \cup R^3 \cup \ldots$$

Closures of Relations

Theorem: The transitive closure of a relation R equals the connectivity relation R*.

But how can we compute R* ?

Lemma: Let A be a set with n elements, and let R be a relation on A. If there is a path in R from a to b, then there is such a path with length not exceeding n.

Moreover, if $a \neq b$ and there is a path in R from a to b, then there is such a path with length not exceeding (n – 1).

Closures of Relations

This lemma is based on the observation that if a path from a to b visits any vertex more than once, it must include at least one **circuit**.

These circuits can be **eliminated** from the path, and the reduced path will still connect a and b.

Theorem: For a relation R on a set A with n elements, the transitive closure R* is given by:

$R^* = R \cup R^2 \cup R^3 \cup \ldots \cup R^n$

For matrices representing relations we have:

$M_{R^*} = M_R \vee M_R^{[2]} \vee M_R^{[3]} \vee \ldots \vee M_R^{[n]}$

Closures of Relations

Let us finally solve **Example III** by finding the **transitive closure** of the relation R = {(1, 3), (1, 4), (2, 1), (3, 2)} on the set A = {1, 2, 3, 4}.

R can be represented by the following matrix M_R:

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

Closures of Relations

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

$$M_R^{[3]} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad M_R^{[4]} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$M_{R^*} = M_R \vee M_R^{[2]} \vee M_R^{[3]} \vee M_R^{[4]} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Closures of Relations

Solution: The transitive closure of the relation R = {(1, 3), (1, 4), (2, 1), (3, 2)} on the set A = {1, 2, 3, 4} is given by the relation
{(1, 1), (1, 2), (1, 3), (1, 4), (2, 1), (2, 2), (2, 3), (2, 4), (3, 1), (3, 2), (3, 3), (3, 4)}

Warshall's Algorithm

A more efficient way of computing the transitive closure of a relation with digraph on vertices $\{v_1, v_2, \ldots, v_n\}$:

Theorem (p. 606). Let $W_k = (w_{ij}^{[k]})$ be the 0,1 matrix $w_{ij}^{[k]} = 1$ iff there is a path from v_i to v_j with any interior vertices in the set $\{v_1, v_2, \ldots v_k\}$. Then
$w_{ij}^{[k]} = w_{ij}^{[k-1]} \vee (w_{ik}^{[k-1]} \wedge w_{kj}^{[k-1]})$
$W_0 = W_R$, $W_n = W_{R^*}$.

Proof: We'll use induction.

Base case: k=0. $W_0 = W_R$ because there can be no interior vertices, so just a single edge.

Induction step: If true for k-1, show
$w_{ij}^{[k]} = w_{ij}^{[k-1]} \vee (w_{ik}^{[k-1]} \wedge w_{kj}^{[k-1]})$
because there is a path from v_i to v_j using interior vertices from $\{v_1, v_2, \ldots, v_k\}$ iff

- There is a path without v_k as an interior vertex (so $w_{ij}^{[k-1]} = 1$) or
- There is path with v_k as an interior vertex, in which case both $w_{ik}^{[k-1]}$ and $w_{kj}^{[k-1]}$ are 1. (there must be a k-1path from v_i to v_k and from v_k to v_j)

Using Warshall's Algorithm

As shown in the book, the formula giving Warshall's Algorithm easily translates to computer code.

If you do it by hand, just note that in $w_{ij}^{[k]} = w_{ij}^{[k-1]} \vee (w_{ik}^{[k-1]} \wedge w_{kj}^{[k-1]})$ you go from W_{k-1} to W_k by looking at the matrix for W_{k-1}. If you can go from v_i to v_k in W_{k-1} then in W_k you can add an entry ij if v_k goes to v_j in W_{k-1}. (this is easier than it sounds)

Transitive Closure via Warshall's Algorithm

$$W_0 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad W_1 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$W_2 = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \qquad W_3 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$M_{R^*} = W_3 = W_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

27 Oct 2015 CS 320 43