

## Representing Relations Using Matrices

A relation  $R$  on  $A = \{a_1, a_2, \dots, a_n\}$  can be represented by a matrix  $\mathbf{M}$  by:

$$m_{ij} = \begin{cases} 1 & \text{if } (a_i, a_j) \in R \\ 0 & \text{if } (a_i, a_j) \notin R \end{cases}$$

E.g.,  $R = \{(1, 3), (1, 4), (2, 3), (2, 4), (3, 1), (3, 4)\}$  can be represented by the matrix  $\mathbf{M}$ :

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$R^2$  can be obtained by Boolean product  $\mathbf{M} \odot \mathbf{M}$ .

## Transitive Closure Algorithms

This works by repeated composite operations.

**procedure** *transitive\_closure*

(**R**:  $n \times n$  zero-one matrix)

{The **R** matrix represents the relation}

{The **C** matrix will represent the closure}

**C** := **R**

**for**  $i := 2$  **to**  $n$

**C** := **C**  $\vee$  (**C**  $\odot$  **R**)

**return** **C**

This works by “compressing paths.”

**procedure** *warshall*

(**R**:  $n \times n$  zero-one matrix)

{The **R** matrix represents the relation}

{The **C** matrix will represent the closure}

**C** := **R**

**for**  $k := 1$  **to**  $n$

**for**  $i := 1$  **to**  $n$

**for**  $j := 1$  **to**  $n$

$c_{ij} := c_{ij} \vee (c_{ik} \wedge c_{kj})$

**return** **C**