Section 4.1: Properties of Binary Relations

A “binary relation” R over some set A is a subset of A×A. If (x,y) ∈ R we sometimes write x R y.

**Example:** Let R be the binary relation “less” (“<”) over \( \mathbb{N} \).
\[
\{(0,1), (0,2), \ldots (1,2), (1,3), \ldots \}
\]
(4,7) ∈ R
Normally, we write: 4 < 7

**Additional Examples:** Here are some binary relations over A={0,1,2}
- Ø \( (nothing \ is \ related \ to \ anything) \)
- A×A \( (everything \ is \ related \ to \ everything) \)
- eq = \{(0,0), (1,1),(2,2)\}
- less = \{(0,1),(0,2),(1,2)\}
Representing Relations with Digraphs (directed graphs)

Let $R = \{(a,b), (b,a), (b,c)\}$ over $A=\{a,b,c\}$

We can represent $R$ with this graph:
Properties of Binary Relations:

R is **reflexive**
- \( x \, R \, x \) for all \( x \in A \)
  - Every element is related to itself.

R is **symmetric**
- \( x \, R \, y \) implies \( y \, R \, x \), for all \( x, y \in A \)
  - The relation is reversible.

R is **transitive**
- \( x \, R \, y \) and \( y \, R \, z \) implies \( x \, R \, z \), for all \( x, y, z \in A \)
  - Example:
    - \( i < 7 \) and \( 7 < j \) implies \( i < j \).

R is **irreflexive**
- \((x, x) \notin R\), for all \( x \in A \)
  - Elements aren’t related to themselves.

R is **antisymmetric**
- \( x \, R \, y \) and \( y \, R \, x \) implies that \( x = y \), for all \( x, y, z \in A \)
  - Example: \( i \leq 7 \) and \( 7 \leq i \) implies \( i = 7 \).
Properties of Binary Relations:

R is **reflexive**
\[ x \in R \ x \text{ for all } x \in A \]
Every element is related to itself.

R is **symmetric**
\[ x \in R \ y \text{ implies } y \in R \ x, \text{ for all } x, y \in A \]
The relation is reversible.

R is **transitive**
\[ x \in R \ y \text{ and } y \in R \ z \text{ implies } x \in R \ z, \text{ for all } x, y, z \in A \]
Example:
\[ i < 7 \text{ and } 7 < j \text{ implies } i < j. \]

R is **irreflexive**
\[ (x, x) \notin R, \text{ for all } x \in A \]
Elements aren’t related to themselves.

R is **antisymmetric**
\[ x \in R \ y \text{ and } y \in R \ x \text{ implies that } x = y, \text{ for all } x, y, z \in A \]
Example:
\[ i \leq 7 \text{ and } 7 \leq i \text{ implies } i = 7. \]
Properties of Binary Relations:

R is **reflexive**
\[ x \, R \, x \text{ for all } x \in A \]
Every element is related to itself.

R is **symmetric**
\[ x \, R \, y \implies y \, R \, x \text{ for all } x,y \in A \]
The relation is reversible.

R is **transitive**
\[ x \, R \, y \text{ and } y \, R \, z \implies x \, R \, z \text{ for all } x,y,z \in A \]
Example:
\[ i < 7 \text{ and } 7 < j \implies i < j. \]

R is **irreflexive**
\[ (x,x) \not\in R, \text{ for all } x \in A \]
Elements aren’t related to themselves.

R is **antisymmetric**
\[ x \, R \, y \text{ and } y \, R \, x \text{ implies that } x = y, \text{ for all } x,y,z \in A \]
Example: \[ i \leq 7 \text{ and } 7 \leq i \implies i = 7. \]

**Symmetric:**
All edges are 2-way:
Might as well use undirected edges!
Properties of Binary Relations:

R is **reflexive**
- \( x \mathrel{R} x \) for all \( x \in A \)
- Every element is related to itself.

R is **symmetric**
- \( x \mathrel{R} y \) implies \( y \mathrel{R} x \), for all \( x, y \in A \)
- The relation is reversible.

R is **transitive**
- \( x \mathrel{R} y \) and \( y \mathrel{R} z \) implies \( x \mathrel{R} z \), for all \( x, y, z \in A \)
- Example:
  - \( i < 7 \) and \( 7 < j \) implies \( i < j \).

R is **irreflexive**
- \((x,x) \notin R\), for all \( x \in A \)
- Elements aren’t related to themselves.

R is **antisymmetric**
- \( x \mathrel{R} y \) and \( y \mathrel{R} x \) implies that \( x = y \), for all \( x, y, z \in A \)
- Example: \( i \leq 7 \) and \( 7 \leq i \) implies \( i = 7 \).

**Symmetric:**
All edges are 2-way: Might as well use undirected edges!
Properties of Binary Relations:

R is **reflexive**
\[ x \mathbin{R} x \text{ for all } x \in A \]
Every element is related to itself.

R is **symmetric**
\[ x \mathbin{R} y \text{ implies } y \mathbin{R} x, \text{ for all } x,y \in A \]
The relation is reversible.

R is **transitive**
\[ x \mathbin{R} y \text{ and } y \mathbin{R} z \text{ implies } x \mathbin{R} z, \text{ for all } x,y,z \in A \]
Example:
\[ i<7 \text{ and } 7<j \text{ implies } i<j. \]

R is **irreflexive**
\[ (x,x) \notin R, \text{ for all } x \in A \]
Elements aren’t related to themselves.

R is **antisymmetric**
\[ x \mathbin{R} y \text{ and } y \mathbin{R} x \text{ implies that } x=y, \text{ for all } x,y,z \in A \]
Example: \[ i\leq 7 \text{ and } 7\leq i \text{ implies } i=7. \]

**Symmetric:**
All edges are 2-way:
Might as well use undirected edges!
Properties of Binary Relations:

R is **reflexive**
- \( x \mathrel{R} x \) for all \( x \in A \)
  - Every element is related to itself.

R is **symmetric**
- \( x \mathrel{R} y \) implies \( y \mathrel{R} x \), for all \( x, y \in A \)
  - The relation is reversible.

R is **transitive**
- \( x \mathrel{R} y \) and \( y \mathrel{R} z \) implies \( x \mathrel{R} z \), for all \( x, y, z \in A \)
  - Example:
    - \( i < 7 \) and \( 7 < j \) implies \( i < j \).

R is **irreflexive**
- \((x, x) \notin R\), for all \( x \in A\)
  - Elements aren’t related to themselves.

R is **antisymmetric**
- \( x \mathrel{R} y \) and \( y \mathrel{R} x \) implies that \( x = y \), for all \( x, y, z \in A \)
  - Example: \( i \leq 7 \) and \( 7 \leq i \) implies \( i = 7 \).

**Transitive:**
- If you can get from \( x \) to \( y \), then there is an edge directly from \( x \) to \( y \)!
Properties of Binary Relations:

R is **reflexive**
\[ x \ R \ x \text{ for all } x \in A \]
Every element is related to itself.

R is **symmetric**
\[ x \ R \ y \text{ implies } y \ R \ x, \text{ for all } x,y \in A \]
The relation is reversible.

R is **transitive**
\[ x \ R \ y \text{ and } y \ R \ z \text{ implies } x \ R \ z, \text{ for all } x,y,z \in A \]
Example:
\[ i < 7 \text{ and } 7 < j \text{ implies } i < j. \]

R is **irreflexive**
\[ (x,x) \not\in R, \text{ for all } x \in A \]
Elements aren’t related to themselves.

R is **antisymmetric**
\[ x \ R \ y \text{ and } y \ R \ x \text{ implies that } x = y, \text{ for all } x,y,z \in A \]
Example: \[ i \leq 7 \text{ and } 7 \leq i \text{ implies } i = 7. \]

**Transitive:**
If you can get from x to y, then there is an edge directly from x to y!
Properties of Binary Relations:

R is **reflexive**
\[ x \ R \ x \text{ for all } x \in A \]
Every element is related to itself.

R is **symmetric**
\[ x \ R \ y \text{ implies } y \ R \ x, \text{ for all } x, y \in A \]
The relation is reversible.

R is **transitive**
\[ x \ R \ y \text{ and } y \ R \ z \text{ implies } x \ R \ z, \text{ for all } x, y, z \in A \]
Example:
\[ i < 7 \text{ and } 7 < j \text{ implies } i < j. \]

R is **irreflexive**
\[ (x,x) \notin R, \text{ for all } x \in A \]
Elements aren’t related to themselves.

R is **antisymmetric**
\[ x \ R \ y \text{ and } y \ R \ x \text{ implies } x = y, \text{ for all } x, y, z \in A \]
Example: \[ i \leq 7 \text{ and } 7 \leq i \text{ implies } i = 7. \]
Properties of Binary Relations:

R is **reflexive**
- \( x \mathbin{R} x \) for all \( x \in \mathcal{A} \)
  - Every element is related to itself.

R is **symmetric**
- \( x \mathbin{R} y \) implies \( y \mathbin{R} x \), for all \( x, y \in \mathcal{A} \)
  - The relation is reversible.

R is **transitive**
- \( x \mathbin{R} y \) and \( y \mathbin{R} z \) implies \( x \mathbin{R} z \), for all \( x, y, z \in \mathcal{A} \)
  - Example: \( i < 7 \) and \( 7 < j \) implies \( i < j \).

R is **irreflexive**
- \((x, x) \notin R\), for all \( x \in \mathcal{A} \)
  - Elements aren’t related to themselves.

R is **antisymmetric**
- \( x \mathbin{R} y \) and \( y \mathbin{R} x \) implies that \( x = y \), for all \( x, y, z \in \mathcal{A} \)
  - Example: \( i \leq 7 \) and \( 7 \leq i \) implies \( i = 7 \).

**Antisymmetric:**
You won’t see any edges like these!
(although \( x \mathbin{R} x \) is okay:)
Properties of Binary Relations:

R is reflexive \( x \ R \ x \) for all \( x \in A \)
R is symmetric \( x \ R \ y \) implies \( y \ R \ x \), for all \( x,y \in A \)
R is transitive \( x \ R \ y \) and \( y \ R \ z \) implies \( x \ R \ z \), for all \( x,y,z \in A \)
R is irreflexive \( (x,x) \not\in R \), for all \( x \in A \)
R is antisymmetric \( x \ R \ y \) and \( y \ R \ x \) implies that \( x=y \), for all \( x,y,z \in A \)

Examples: Here are some binary relations over \( A=\{0,1\} \).
Which of the properties hold?

Answers:

\[ \emptyset \]
\[ A \times A \]
\[ eq = \{(0,0), (1,1)\} \]
\[ less = \{(0,1)\} \]
Properties of Binary Relations:

R is reflexive \( x R x \) for all \( x \in A \)

R is symmetric \( x R y \) implies \( y R x \), for all \( x, y \in A \)

R is transitive \( x R y \) and \( y R z \) implies \( x R z \), for all \( x, y, z \in A \)

R is irreflexive \( (x, x) \notin R \), for all \( x \in A \)

R is antisymmetric \( x R y \) and \( y R x \) implies that \( x = y \), for all \( x, y, z \in A \)

Examples: Here are some binary relations over \( A = \{0, 1\} \).

Which of the properties hold?

Answers:

\[ \emptyset \text{ symmetric, transitive, irreflexive, antisymmetric} \]

\[ A \times A \text{ reflexive, symmetric, transitive} \]

\[ \text{eq} = \{(0,0), (1,1)\} \text{ reflexive, symmetric, transitive, antisymmetric} \]

\[ \text{less} = \{(0,1)\} \text{ transitive, irreflexive, antisymmetric} \]
Composition of Relations

If R and S are binary relations, then the composition of R and S is

\[ R \circ S = \{(x,z) \mid x \in R y \text{ and } y \in S z \text{ for some } y \} \]
Composition of Relations

If R and S are binary relations, then the composition of R and S is

\[ R \circ S = \{(x,z) \mid x \in R \, y \text{ and } y \in S \, z \text{ for some } y \} \]
Composition of Relations

If R and S are binary relations, then the composition of R and S is

\[ R \circ S = \{(x,z) | x R y \text{ and } y S z \text{ for some } y \} \]

Examples:

\[ \text{eq} \circ \text{less} = ? \]

\[ R \circ \emptyset = ? \]

\[ \text{isMotherOf} \circ \text{isFatherOf} = ? \]

\[ \text{isSonOf} \circ \text{isSiblingOf} = ? \]
Composition of Relations

If R and S are binary relations, then the composition of R and S is
\[ R \circ S = \{(x,z) \mid x R y \text{ and } y S z \text{ for some } y\} \]

**Examples:**

\[ \text{eq} \circ \text{less} = \text{less} \]
\[ \{ (x,z) \mid x=y \text{ and } y<x, \text{ for some } y\} \]

\[ R \circ \emptyset = \emptyset \]

\[ \text{isMotherOf} \circ \text{isFatherOf} = \text{isPaternalGrandmotherOf} \]
\[ \{ (x,z) \mid x \text{ isMotherOf } y \text{ and } y \text{ isFatherOf } x, \text{ for some } y\} \]

\[ \text{isSonOf} \circ \text{isSiblingOf} = \text{isNephewOf} \]
\[ \{ (x,z) \mid x \text{ isSonOf } y \text{ and } y \text{ isSiblingOf } x, \text{ for some } y\} \]
Representing Relations with Digraphs (directed graphs)

Let $R = \{(a,b), (b,a),(b,c)\}$ over $A=\{a,b,c\}$

Let $R^2 = R \circ R = ?$

We can represent $R$ graphically:
Representing Relations with Digraphs (directed graphs)

Let $R = \{(a,b), (b,a),(b,c)\}$ over $A=\{a,b,c\}$

Let $R^2 = R \circ R$
Let $R^3 = R \circ R \circ R = ?$

We can represent $R$ graphically:
Representing Relations with Digraphs (directed graphs)

Let $R = \{ (a,b), (b,a), (b,c) \}$ over $A = \{a,b,c\}$

Let $R^2 = R \circ R$
Let $R^3 = R \circ R \circ R = R^2 \circ R$.

We can represent $R$ graphically:

In this example, $R^3$ happens to be the same relation as $R$.
$R^3 = R$

Note: By definition, $R^0 = \text{Eq}$, where $x \text{ Eq } y$ iff $x = y$. 
Reflexive Closure

Given a relation R, we want to add to it just enough “edges” to make the resulting relation satisfy the reflexive property.

**Reflexive Closure of R is** \( r(R) = R \cup Eq \), where Eq is the equality relation.

**Example:**

\[ r(R) = R \cup Eq = \{(a,b),(b,a),(b,c),(a,a),(b,b),(c,c)\} \]
Symmetric Closure

Given a relation R, we want to add to it just enough “edges” to make the resulting relation satisfy the symmetric property.

**Symmetric Closure of R** is $s(R) = R \cup R^c$, where $R^c$ is the converse relation. $R^c = \{(b,a) \mid a R b\}$

**Example:**

$s(R) = R \cup R^c = \{(a,b),(b,a),(b,c),(c,b)\}$
Transitive Closure

Given a relation $R$, we want to add to it just enough “edges” to make the resulting relation satisfy the transitivity property.

Transitive Closure of $R$ is $t(R) = R \cup R^2 \cup R^3 \cup ...$

Note: If the number of nodes is finite...
If $|A| = n$ then $t(R) = R \cup R^2 \cup R^3 \cup ... \cup R^n$

Example:

$t(R) = R \cup R^2 \cup R^3 = \{(a,b),(b,a),(b,c),(a,a),(b,b),(a,c)\}$

If there is a path from $x$ to $y$, then add an edge directly from $x$ to $y$. 
In-Class Quiz:

Let $R = \{(x, x+1) \mid x \in \mathbb{Z}\}$

What is $t(R)$?
What is $rt(R)$?
What is $st(R)$?
In-Class Quiz:

Let \( R = \{(x, x+1) \mid x \in \mathbb{Z} \} \)

What is \( t(R) \)?
What is \( rt(R) \)?
What is \( st(R) \)?
In-Class Quiz:

Let \( R = \{(x, x+1) \mid x \in \mathbb{Z} \} \)

What is \( t(R) \)?  \( < \)
What is \( rt(R) \)?  \( \leq \)
What is \( st(R) \)?
In-Class Quiz:

Let \( R = \{(x, x+1) \mid x \in \mathbb{Z} \} \)

What is \( t(R) \)?  
\( < \)
What is \( rt(R) \)?  
\( \leq \)
What is \( st(R) \)?  
\( \neq \)
Adjacency Matrix

Idea: Use a matrix to represent a directed graph (or a relation).

Let \( R = \{(a,b),(b,c),(c,d)\} \)

Number the elements in the set: 1, 2, 3, ...
Adjacency Matrix

Idea: Use a matrix to represent a directed graph (or a relation).

Let $R = \{(a,b),(b,c),(c,d)\}$

Number the elements in the set: 1, 2, 3, ...

\[ \begin{array}{cccc}
  & a & b & c & d \\
 1 & & & & \\
 2 & & & & \\
 3 & & & & \\
 4 & & & & \\
\end{array} \]
Adjacency Matrix

Idea: Use a matrix to represent a directed graph (or a relation).

Let $R = \{(a,b),(b,c),(c,d)\}$

Number the elements in the set: 1, 2, 3, ...

Now we can write $R = \{(1,2),(2,3),(3,4)\}$

The matrix $M$ will have a “1” in position $M_{ij}$ if $i R j$, and “0” otherwise.

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

1 → 2 → 3 → 4
Computing the Transitive Closure: Warshall’s Algorithm

We can use this matrix to compute the \( t(R) \), the transitive closure of \( R \).

Idea: Every time we find this pattern:

\[ i \rightarrow k \rightarrow j \]

...add this edge:

```
Warshall’s Algorithm

for k := 1 to n
    for i := 1 to n
        for j := 1 to n
            if \( M_{ik} = M_{kj} = 1 \) then
                \( M_{ij} := 1 \)
            endIf
        endFor
    endFor
endFor
```
Computing the Transitive Closure: Warshall’s Algorithm

We can use this matrix to compute the t(R), the transitive closure of R.

Idea: Every time we find this pattern:

...add this edge:

Consider all ways to bypass node 1. Then forget about node 1. Consider all ways to bypass node 2. and so on...

Warshall’s Algorithm

\[
\begin{align*}
&\text{for } k := 1 \text{ to } n \\
&\quad \text{for } i := 1 \text{ to } n \\
&\quad \quad \text{for } j := 1 \text{ to } n \\
&\quad \quad \quad \text{if } M_{ik} = M_{kj} = 1 \text{ then} \\
&\quad \quad \quad \quad M_{ij} := 1 \\
&\quad \quad \text{endIf} \\
&\quad \text{endFor} \\
&\text{endFor} \\
&\text{endFor}
\end{align*}
\]
Computing the Transitive Closure: Warshall’s Algorithm

We can use this matrix to compute the $t(R)$, the transitive closure of $R$.

Idea: Every time we find this pattern:

$$\text{...add this edge:}$$

Consider all ways to bypass node 1. Then forget about node 1. Consider all ways to bypass node 2. and so on...

Warshall’s Algorithm

```plaintext
for k := 1 to n
  for i := 1 to n
    for j := 1 to n
      if $M_{ik} = M_{kj} = 1$ then
        $M_{ij} := 1$
      endif
    endFor
  endFor
endFor
```
Computing the Transitive Closure: Warshall’s Algorithm

We can use this matrix to compute the $t(R)$, the transitive closure of $R$.

Idea: Every time we find this pattern:

...add this edge:

Consider all ways to bypass node 1. Then forget about node 1. Consider all ways to bypass node 2. and so on...

Warshall’s Algorithm

```
for k := 1 to n
    for i := 1 to n
        for j := 1 to n
            if $M_{ik} = M_{kj} = 1$ then
                $M_{ij} := 1$
            endIf
        endFor
    endFor
endFor
```
Computing the Transitive Closure: Warshall’s Algorithm

We can use this matrix to compute the t(R), the transitive closure of R.

Idea: Every time we find this pattern:

...add this edge:

Consider all ways to bypass node 1. Then forget about node 1. Consider all ways to bypass node 2. and so on...

Warshall’s Algorithm

```plaintext
for k := 1 to n
    for i := 1 to n
        for j := 1 to n
            if M_{ik} = M_{kj} = 1 then
                M_{ij} := 1
            endIf
        endFor
    endFor
endFor
```
Computing the Transitive Closure: Warshall’s Algorithm

We can use this matrix to compute the $t(R)$, the transitive closure of $R$.

Idea: Every time we find this pattern:

...add this edge:

Consider all ways to bypass node 1. Then forget about node 1. Consider all ways to bypass node 2. and so on...

Warshall’s Algorithm

```
for k := 1 to n
    for i := 1 to n
        for j := 1 to n
            if $M_{ik} = M_{kj} = 1$ then
                $M_{ij} := 1$
            endif
        endFor
    endFor
endFor
```
Example:

k=1

\[ M = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \]

Warshall’s Algorithm

```plaintext
for k := 1 to n
    for i := 1 to n
        for j := 1 to n
            if M_{ik} = M_{kj} = 1 then
                M_{ij} := 1
            endif
        endfor
    endfor
endfor
```
Example:

\( k = 2 \)

\[
M = \begin{bmatrix}
0 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

Warshall’s Algorithm

\[
\text{for } k := 1 \text{ to } n \\
\quad \text{for } i := 1 \text{ to } n \\
\quad \quad \text{for } j := 1 \text{ to } n \\
\quad \quad \quad \text{if } M_{ik} = M_{kj} = 1 \text{ then} \\
\quad \quad \quad \quad M_{ij} := 1 \\
\quad \quad \quad \text{endIf} \\
\quad \quad \text{endFor} \\
\quad \text{endFor} \\
\text{endFor}
\]
Example:

\[ k = 3 \]

\[
M = \begin{bmatrix}
0 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0
\end{bmatrix}
\]

Warshall’s Algorithm

\[
\text{for } k := 1 \text{ to } n \\
\quad \text{for } i := 1 \text{ to } n \\
\quad \quad \text{for } j := 1 \text{ to } n \\
\quad \quad \quad \text{if } M_{ik} = M_{kj} = 1 \text{ then} \\
\quad \quad \quad \quad M_{ij} := 1 \\
\quad \quad \quad \text{endIf} \\
\quad \quad \text{endFor} \\
\quad \text{endFor} \\
\text{endFor}
\]
Example:

\[ \text{k} = 3 \]

\[
M = \begin{bmatrix}
0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

Warshall’s Algorithm

\[
\text{for } k := 1 \text{ to } n \\
\quad \text{for } i := 1 \text{ to } n \\
\quad \quad \text{for } j := 1 \text{ to } n \\
\quad \quad \quad \text{if } M_{ik} = M_{kj} = 1 \text{ then} \\
\quad \quad \quad \quad M_{ij} := 1 \\
\quad \quad \end{If} \\
\quad \endFor \\
\endFor \\
\endFor
\]
**Example:**

$k=4$ ...Done;
no more changes.

\[ M = \begin{bmatrix}
0 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 \\
\end{bmatrix} \]

**Warshall’s Algorithm**

```
for k := 1 to n
    for i := 1 to n
        for j := 1 to n
            if \( M_{ik} = M_{kj} = 1 \)
                then
                    \( M_{ij} := 1 \)
            endIf
        endFor
    endFor
endFor
```
Path Problems: Floyd’s Algorithm

Consider a directed graph with weights on the edges.

Problems:
Find the cheapest path from x to y.
Find the shortest path from x to y.

(Just make all weights = 1!)
Path Problems: Floyd’s Algorithm
Consider a directed graph with weights on the edges.

Idea:
Represent the graph with a matrix.
Store the weights.
For non-existent edges, use a weight of \( \infty \).
Then modify Warshall’s Algorithm.

\[
M = \begin{bmatrix}
0 & 10 & 10 & \infty & 20 & 10 \\
\infty & 0 & \infty & 30 & \infty & \infty \\
\infty & \infty & 0 & 30 & \infty & \infty \\
\infty & \infty & \infty & 0 & \infty & \infty \\
\infty & \infty & \infty & 40 & 0 & \infty \\
\infty & \infty & \infty & \infty & 5 & 0
\end{bmatrix}
\]
Path Problems: Floyd’s Algorithm
Consider a directed graph with weights on the edges.

Floyd’s Algorithm:

\[
\begin{align*}
\text{for } &k := 1 \text{ to } n \\
\text{for } &i := 1 \text{ to } n \\
\text{for } &j := 1 \text{ to } n \\
\text{if } &M_{ik} + M_{kj} < M_{ij} \\
&M_{ij} := M_{ik} + M_{kj}
\end{align*}
\]

\[
M = \begin{bmatrix}
0 & 10 & 10 & \infty & 20 & 10 \\
\infty & 0 & \infty & 30 & \infty & \infty \\
\infty & \infty & 0 & 30 & \infty & \infty \\
\infty & \infty & \infty & 0 & \infty & \infty \\
\infty & \infty & \infty & 40 & 0 & \infty \\
\infty & \infty & \infty & \infty & 5 & 0
\end{bmatrix}
\]
Path Problems: Floyd’s Algorithm

Consider a directed graph with weights on the edges.

Floyd’s Algorithm:

for \( k := 1 \) to \( n \)
  for \( i := 1 \) to \( n \)
    for \( j := 1 \) to \( n \)
      if \( M_{ik} + M_{kj} < M_{ij} \)
        \( M_{ij} := M_{ik} + M_{kj} \)
      endif
    endfor
  endfor
endfor

\[
M = \begin{bmatrix}
0 & 10 & 10 & 40 & 15 & 10 \\
\infty & 0 & \infty & 30 & \infty & \infty \\
\infty & \infty & 0 & 30 & \infty & \infty \\
\infty & \infty & \infty & 0 & \infty & \infty \\
\infty & \infty & \infty & 40 & 0 & \infty \\
\infty & \infty & \infty & 45 & 5 & 0
\end{bmatrix}
\]
Path Problems: Floyd’s Algorithm

How can we remember the best path? P=next node in best path!

Floyd’s Algorithm:

\[
\begin{align*}
\text{for } & k := 1 \text{ to } n \\
\text{for } & i := 1 \text{ to } n \\
& \text{for } j := 1 \text{ to } n \\
& \text{if } M_{ik} + M_{kj} < M_{ij} \\
& \quad M_{ij} := M_{ik} + M_{kj} \\
& \quad P_{ij} := k \\
\end{align*}
\]

\[
M = \begin{bmatrix}
0 & 10 & 10 & 40 & 15 & 10 \\
\infty & 0 & \infty & 30 & \infty & \infty \\
\infty & \infty & 0 & 30 & \infty & \infty \\
\infty & \infty & \infty & 0 & \infty & \infty \\
\infty & \infty & \infty & 40 & 0 & \infty \\
\infty & \infty & \infty & 45 & 5 & 0 \\
\end{bmatrix}
\quad P = \begin{bmatrix}
0 & 0 & 0 & 2 & 6 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 5 & 0 & 0 \\
\end{bmatrix}
\]