Path and tree problems

Lecture 12
ICOM 4075
Finite binary relations, this is, relations whose set of pairs is finite, can be interpreted as directed graphs through the rule:

“(a, b) in the relation if and only if (a, b) are connected and a precedes b in the graph”

Reciprocally, the set E of the edges (connections) of a graph $G = (V, E)$ can interpreted as a relation over V. This relation is simply described by either of the rules:

a) “(a, b) is a pair in the relation if and only if (a, b) is a pair in E and G is a directed graph”

b) “Both, (a, b) and (b, a) are pairs in the relation if and only if \{a, b\} is in E”
What is a path problem?

In general, a path problem is the problem of finding a path that satisfies some specific properties between two or more vertices in a given graph.

A large variety of problems can be viewed as instances of path problems. Among them are communications across a network, distribution or recollection of goods, net-centric computation, routing, production and service scheduling, etc.

Most of these problems cannot be solved without the help of carefully crafted computing methods.
First problem

The one – step connection problem (OCP): Given a directed graph D find the smaller graph T such that
1. D is a sub – graph of T, and
2. Every path form a to b in D can be executed in one step in T (this is, (a, b) is an edge in T)

In order to model this problem, we introduce the concept of transitive closure
Graphs and relations

As remarked before, the set of edges in a directed graph can be thought as a relation and reciprocally, any binary relation can be thought as a directed graph. Here is an application of this principle:

**Example:** Let \( G = (\{1, 2, 3, 4\}, \{(1, 2), (2, 3), (3, 4), (4, 3)\}) \)

The associated relation is relation

\[
R = \{(1, 2), (2, 3), (3, 4), (4, 3)\}
\]
Introducing adjacency matrices

**Definition**: The adjacency matrix of a graph with n vertices is an array $A[i, j]$, $0 \leq i, j \leq n – 1$, whose entries are defined by the rule:

$$A[i, j] = \begin{cases} 
1, & \text{if nodes } i \text{ and } j \text{ are connected} \\
0, & \text{otherwise}
\end{cases}$$
Example

The adjacency matrix for the graph $G = (V, E)$, $V = \{1, 2, 3, 4\}$ and $E = \{(1, 2), (2, 3), (3, 4), (4, 3)\}$ in the previous example is:

$$
\begin{bmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
\end{bmatrix}
$$
Introducing transitive relations

**Definition**: A relation R is said to be transitive if and only if

\[(a, b) \text{ and } (b, c) \in R \text{ implies that } (a, c) \in R\]

**Illustration**: Relation \(\{(x, y): x < y\}\) is transitive but the relation \(E = \{(1, 2), (2, 3), (3, 4), (4, 3)\}\) (edges of the previous digraph) is not.

**Definition**: The transitive closure of a relation R, denoted \(t(R)\) is the smallest transitive relation that contains R
Finding the transitive closure of a relation

Consider again the relation:

\[ E = \{(1, 2), (2, 3), (3, 4), (4, 3)\} \]. Let’s compute \( t(E) \)

Start with \( t(E) = E \). Now, generate new pairs by composing pairs in \( t(E) \):

- \((1, 2)\) and \((2, 3)\) in \( t(E) \) imply \((1, 3)\) in \( t(E) \)
- \((2, 3)\) and \((3, 4)\) in \( t(E) \) imply \((2, 4)\) in \( t(E) \)
- \((1, 3)\) and \((3, 4)\) in \( t(E) \) imply \((1, 4)\) in \( t(E) \)

So, \( t(E) = \{(1, 2), (1, 3), (2, 3), (2, 4), (3, 4), (4, 3)\} \)

Compositions in \( t(E) \) render no new pairs

\textbf{Definition}: A relation is transitively closed if composition between pairs render no new pairs
Graph of relation $t(E)$ solves OCP for the original graph!!!

The graph of $t(E)$ is:

There is a one-step path for any path in the original graph $(V, E)!!!!$
So, we’ve got a method for solving OCP

The following method solves OCP for any given directed graph $G = (V, E)$

**Method:**

1. Model the edges in the given graph as a relation $E$

2. Compute the transitive closure of the relation $E$, this is $t(E)$

3. Build the graph $C = (V, t(E))$
Warshall algorithm for computing the transitive closure

**Input**: an n x n array M representing the adjacency matrix of a directed graph

For k ← 0 to n − 1

For i ← 0 to n − 1

For j ← 0 to n − 1

If M[i, k] = M[k, j] = 1

M[i, j] ← 1

Return M
Warshall applied to previous example

<table>
<thead>
<tr>
<th>Input:</th>
<th>k = 0</th>
<th>k = 1</th>
<th>k = 2</th>
<th>k = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>k = 0</td>
<td>0 1 0 0</td>
<td>0 1 1 0</td>
<td>0 1 1 1</td>
<td>0 1 1 1</td>
</tr>
<tr>
<td></td>
<td>0 0 1 0</td>
<td>0 0 1 0</td>
<td>0 0 1 1</td>
<td>0 0 1 1</td>
</tr>
<tr>
<td></td>
<td>0 0 0 1</td>
<td>0 0 0 1</td>
<td>0 0 0 1</td>
<td>0 0 1 1</td>
</tr>
<tr>
<td></td>
<td>0 0 1 0</td>
<td>0 0 1 0</td>
<td>0 0 1 1</td>
<td>0 0 1 1</td>
</tr>
</tbody>
</table>
Second problem

The minimal weighted path problem (MWP):
Given a weighted directed graph and two nodes i and j, find a path between i and j such that the sum of the weights of its edges is minimal.

The minimal weighted path problem can be modeled with a variant of the adjacency matrix, the so-called weighted adjacency matrix.
Weighted adjacency matrix

**Definition**: The weighted adjacency matrix of a weighted graph with \( n \) vertices is an array \( A[i, j], 0 \leq i, j \leq n - 1 \), whose entries are defined by the rule:

\[
A[i, j] = \begin{cases} 
\text{weight of (i, j), if nodes i and j are connected} \\
\text{Inf, otherwise}
\end{cases}
\]
Example: a directed weighted graph

Let G be the graph:
Its representations as pair of sets and weighted matrix

The graph is:
\[ G = (\{1, 2, 3, 4, 5, 6\}, \{(1, 2), 10\}, ((1, 3), 10\}, ((1, 5), 20\}, ((1, 6), 10\}, ((2, 4), 30\}, ((3, 4), 30\}, ((5, 4), 40\}, ((6, 5), 5)\} \]

And the weighted adjacency matrix of G is:

\[
\begin{array}{cccccccc}
0 & 10 & 10 & \text{Inf} & 20 & 10 & \\
\text{Inf} & 0 & \text{Inf} & 30 & \text{Inf} & \text{Inf} & \\
\text{Inf} & \text{Inf} & 0 & 30 & \text{Inf} & \text{Inf} & \\
\text{Inf} & \text{Inf} & \text{Inf} & 0 & \text{Inf} & \text{Inf} & \\
\text{Inf} & \text{Inf} & \text{Inf} & 40 & 0 & \text{Inf} & \\
\text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & 5 & 0 & \\
\end{array}
\]
Observation

There are 4 different paths between vertex 1 and vertex 4

These are:

i. 1, 2, 4
ii. 1, 3, 4
iii. 1, 5, 4
iv. 1, 6, 5, 4

Question is: Which one(s) is (are) minimal weighted path?
The answer

By exhaustive calculation

<table>
<thead>
<tr>
<th>Path</th>
<th>Path Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 2, 4</td>
<td>10 + 30 = 40</td>
</tr>
<tr>
<td>1, 3, 4</td>
<td>10 + 30 = 40</td>
</tr>
<tr>
<td>1, 5, 4</td>
<td>20 + 40 = 60</td>
</tr>
<tr>
<td>1, 6, 5, 4</td>
<td>10 + 5 + 40 = 55</td>
</tr>
</tbody>
</table>

**Conclusion:** 1, 2, 3 and 1, 3, 4 are solutions of MWP for the pair of vertices i = 1 and j = 4
All solutions: Floyd’s algorithm

Floyd’s algorithm replaces the weighted adjacency matrix with a matrix that represents the minimal weights for paths between vertices

**Input:** The weighted adjacency matrix $M$

For $k \leftarrow 0, n - 1$

For $i \leftarrow 0, n - 1$

For $j \leftarrow 0, n - 1$

$$M[i, j] \leftarrow \min\{M[i, j], M[i, k] + M[k, j]\}$$

Return $M$
The output of Floyd’s algorithm for the graph $G = (\{1, 2, 3, 4, 5, 6\}, \{((1,2), 10), ((1, 3), 10), ((1, 5), 20), ((1, 6), 10), ((2, 4), 30), ((3, 4), 30), ((5, 4), 40), ((6, 5), 5)\} )$ is the matrix:

$$
\begin{array}{cccccccc}
0 & 10 & 10 & 40 & 15 & 10 \\
\text{Inf} & 0 & \text{Inf} & 30 & \text{Inf} & \text{Inf} \\
\text{Inf} & \text{Inf} & 0 & 30 & \text{Inf} & \text{Inf} \\
\text{Inf} & \text{Inf} & \text{Inf} & 0 & \text{Inf} & \text{Inf} \\
\text{Inf} & \text{Inf} & \text{Inf} & 40 & 0 & \text{Inf} \\
\text{Inf} & \text{Inf} & \text{Inf} & 45 & 5 & 0 \\
\end{array}
$$

The minimal weight for a path from 1 to 4 is 40.
A third problem

In the previous lecture we defined the minimal spanning tree problem (MST) as:

Given a connected weighted graph $G = (V, E)$ and a vertex $v$ in $V$, find a minimal spanning tree that has $v$ as root.

An instance of the minimal spanning tree problem is the problem of designing a routing algorithm for broadcasting a message from a node to the rest of the computers in a network in such a way that delays are minimal.
Consider the graph:

and the vertex A. What is the minimal spanning tree?
The answer is computed with Prim’s algorithm
Prim’s algorithm

**Input**: An undirected weighted graph \( G = (V, E) \) and a vertex \( v \)

Set \( S = \emptyset \) and \( W = \{v\} \)

While \( W \neq V \)

Find edge \( \{a, b\} \) = minimal weight among all \( x \) in \( W \) and \( y \) in \( V - W \)

\( S \leftarrow S \cup \{\{a, b\}\} \)

\( W \leftarrow W \cup \{b\} \)

Return \((V, S)\)
Run of Prim’s algorithm on the previous graph with input vertex A

<table>
<thead>
<tr>
<th>S</th>
<th>W</th>
<th>V – W</th>
</tr>
</thead>
<tbody>
<tr>
<td>φ</td>
<td>{ A }</td>
<td>{B, C, D, E, F, G}</td>
</tr>
<tr>
<td>{A, C}</td>
<td>{A, C}</td>
<td>{B, D, E, F, G}</td>
</tr>
<tr>
<td>{A, C}, {C, D}</td>
<td>{A, C, D}</td>
<td>{B, E, F, G}</td>
</tr>
<tr>
<td>{A, C}, {C, D}, {D, B}</td>
<td>{A, C, D, B}</td>
<td>{E, F, G}</td>
</tr>
<tr>
<td>{A, C}, {C, D}, {D, B}, {D, F}</td>
<td>{A, C, D, B, F}</td>
<td>{E, G}</td>
</tr>
<tr>
<td>{A, C}, {C, D}, {D, B}, {D, F}, {F, G}</td>
<td>{A, C, D, B, F, G}</td>
<td>{E}</td>
</tr>
<tr>
<td>{A, C}, {C, D}, {D, B}, {D, F}, {F, G}, {G, E}}</td>
<td>{A, C, D, B, F, G, E}</td>
<td>φ</td>
</tr>
</tbody>
</table>

The output is the tree:
\[ T = (\{A, B, C, D, E, F, G\}, \{{A, C}, \{C, D\}, \{D, B\}, \{D, F\}, \{F, G\}, \{G, E\}\}) \]
Spanning tree returned by Prim
Summary

• Path problems
• One – step connection problem
• Graphs and relations
• Adjacency matrix
• Transitive relations and closure
• Warshall’s algorithm
• The minimal weighted path problem
• Weighted adjacency matrix
• All solutions to minimal weighted path: Floyd’s algorithm
• Minimal spanning tree
• Prim’s algorithm
Exercises

1. Page 71: Exercises 11, 12
2. Decide whether or not the following relations are transitive
   a) \( R = \{(a, b), (a, d), (a, c), (b, c), (c, d)\} \)
   b) \( R(x, y) : \text{“x integer, y integer and } x + y \text{ even”} \)
3. Given the relation \( E = \{(a, b), (c, b), (c, d), (c, e)\} \)
   a) Write the adjacency matrix
   b) Write each one of the transformations of the adjacency matrix under Warshall’s algorithm
   c) Draw pictures of the graphs \((\{a, b, c, d, e\}, E)\) and \((\{a, b, c, d, e\}, t(E))\)
Exercises

4. Given the weighted graph: ({A, B, C, D, E, F}, {{A,B},2),
   ({F,E},10), ({D,E},9), ({D,C},8)})
   a) Write the weighted adjacency matrix
   b) Apply Floyd’s algorithm and interpret the results
   c) Draw a picture of the graph and identify all minimal weight paths
   d) Find a minimal weighted spanning tree that starting at node A using Prim’s algorithm (write down all intermediate results produced by the method in a table, just as the one in the class)