Lecture 20: Minimum Spanning Trees
CSCI 700 - Algorithms I

Andrew Rosenberg

November 24, 2009
Today

- Minimum Spanning Trees: Kruskal’s Algorithm
- Path Lengths using Adjacency Matrices
Recall, Prim’s Algorithm to construct a Minimum Spanning Tree

- Pick a starting node
- Extend the MST along the shortest edge connecting an included node to an excluded node.
- Continue until every node is in the MST.
Kruskal’s algorithm also identifies Minimum Spanning Trees.

- Identify the lowest weight edge \((u, v)\).
- Add it to the MST if \((u, v)\) does not create a cycle.
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example

The image shows a graph with labeled vertices (a, b, c, d, e, f, g, h, i) and weighted edges. The edges between the vertices are shown with weights as follows:

- a to b: 1
- b to c: 5
- c to d: 3
- c to e: 3
- e to d: 2
- e to f: 3
- f to g: 7
- g to h: 1
- f to i: 1

The graph is a representation of the edges used in the Kruskal’s Algorithm example.
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example

The image shows a graph with vertices labeled a, b, c, d, e, f, g, h, and i. The edges are labeled with their weights as follows:

- Edge ab: weight 1
- Edge bc: weight 2
- Edge cd: weight 3
- Edge ce: weight 3
- Edge de: weight 3
- Edge ef: weight 3
- Edge fg: weight 1
- Edge gh: weight 1
- Edge hi: weight 2

The graph is colored with blue edges indicating the selected edges in the Kruskal’s Algorithm.
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example
Kruskal’s Algorithm Example

The diagram illustrates a graph with nodes labeled a, b, c, d, e, f, g, h, and i, and edges connecting them with weights. The edges are as follows:

- Edge from a to b with weight 1
- Edge from b to c with weight 2
- Edge from c to d with weight 5
- Edge from c to e with weight 3
- Edge from e to f with weight 3
- Edge from f to g with weight 3
- Edge from g to h with weight 1
- Edge from h to i with weight 2

The algorithm proceeds by selecting the edge with the smallest weight that does not form a cycle. In this case, the edges selected are:

- a to b
- c to e
- f to g

These edges form a minimum spanning tree of the graph.
Kruskal’s Algorithm

MST-Kruskal(G,w)

\[ A \leftarrow \emptyset \]
\[ \text{for } v \in V \text{ do} \]
\[ \quad \text{Make-Set}(v) \]
\[ \text{end for} \]
\[ \text{sort}(E,w) \quad // \text{Sort the edges in } E \text{ by weight } w \]
\[ \text{for } (u, v) \in E \text{ do} \]
\[ \quad \text{if } \text{Find-Set}(u) = \text{Find-Set}(v) \text{ then} \]
\[ \quad \quad A \leftarrow A \cup (u, v) \]
\[ \quad \quad \text{Union}(u,v) \]
\[ \quad \text{end if} \]
\[ \text{end for} \]
\[ \text{return } A \]
Using Adjacency Matrices to calculate path length

<table>
<thead>
<tr>
<th>Source</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Diagram:

- D → A → C
- D → A → E
- B → A → C
- B → A → E
- E → A → C
- E → A → D
Adjacency Matrix cell $M_{uv}$ is 1 if $u$ and $v$ are adjacent.

Therefore the adjacency matrix, $M$, contains a count of the number of paths of length 1 between $u$ and $v$.

Claim: $M^d$ contains the number of paths of length $d$ between $u$ and $v$. 
How do we multiply matrices?

\[
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix} \times \begin{pmatrix}
a & b \\
c & d
\end{pmatrix}
= \begin{pmatrix}
a^2 + cb & ba + db \\
ac + cd & cb + d^2
\end{pmatrix}
\]
What does this mean when multiply an adjacency matrix to itself?

\[
\begin{pmatrix}
  a & b \\
  c & d
\end{pmatrix}
= 
\begin{pmatrix}
  N(A \rightarrow A) & N(A \rightarrow B) \\
  N(B \rightarrow A) & N(B \rightarrow B)
\end{pmatrix}
\]
What does this mean when multiply an adjacency matrix to itself?

\[
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}
= \begin{pmatrix}
N(A \rightarrow A) & N(A \rightarrow B) \\
N(B \rightarrow A) & N(B \rightarrow B)
\end{pmatrix}
\]

\[
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}
\ast
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}
= \begin{pmatrix}
a^2 + cb & ba + db \\
ac + cd & cb + d^2
\end{pmatrix}
\]
Path length with Matrix Multiplication

What does this mean when Multiply an adjacency matrix to itself?

\[
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}
= \begin{pmatrix}
N(A \rightarrow A) & N(A \rightarrow B) \\
N(B \rightarrow A) & N(B \rightarrow B)
\end{pmatrix}
\]

\[
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}
\ast
\begin{pmatrix}
a & b \\
c & d
\end{pmatrix}
= \begin{pmatrix}
a^2 + cb & ba + db \\
ac + cd & cb + d^2
\end{pmatrix}
\]

\[
\begin{pmatrix}
N(A, A)^2 + N(A, B)N(B, A) & N(A, B)N(A, A) + N(A, B)N(B, B) \\
N(B, A)N(A, A) + N(B, B)N(B, A) & N(A, B)N(B, A) + N(B, B)^2
\end{pmatrix}
\]
Using Adjacency Matrices to calculate path length

\[ M = \begin{array}{c|ccccc}
\text{SOURCE} & \text{A} & \text{B} & \text{C} & \text{D} & \text{E} \\
\hline
\text{A} & 0 & 1 & 0 & 1 & 0 \\
\text{B} & 0 & 0 & 1 & 0 & 0 \\
\text{C} & 1 & 0 & 0 & 0 & 0 \\
\text{D} & 0 & 0 & 0 & 0 & 1 \\
\text{E} & 1 & 0 & 0 & 0 & 0 \\
\end{array} \]
Using Adjacency Matrices to calculate path length

\[ M^2 = \]

<table>
<thead>
<tr>
<th>Source</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
\text{Source} & \quad \text{Destination} \\
A & \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \\
B & \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \\
C & \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \\
D & \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \\
E & \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \\
\end{align*}
\]
Using Adjacency Matrices to calculate path length

\[ M^3 = \begin{array}{cccccc}
A & B & C & D & E \\
\hline
A & 2 & 0 & 0 & 0 & 0 \\
B & 0 & 1 & 0 & 1 & 0 \\
C & 0 & 0 & 1 & 0 & 1 \\
D & 0 & 1 & 0 & 1 & 0 \\
E & 0 & 0 & 1 & 0 & 1 \\
\end{array} \]
Bye

- Next time (11/24)
  - Hashing