Lecture 28: Minimum Spanning Trees

- Introduction
- Kruskal’s Algorithm
- Prim’s Algorithm
Spanning Trees

- Given an edge weighted graph $G$ (not digraph!), a spanning tree of $G$ is a subgraph $T$ that is:
  - A tree: connected and acyclic.
  - Spanning: includes all of the vertices of $G$. 
INTRODUCTION

Properties

- A connected graph $G$ can have more than one spanning tree.
- All possible spanning trees of $G$ have the same number of vertices and edges.
- A spanning tree has $|V| - 1$ edges.
- A spanning tree by definition cannot have any cycle.
- Adding one edge to the spanning tree would create a cycle (i.e. spanning trees are maximally acyclic).
- Removing one edge from the spanning tree would make the graph disconnected (i.e. spanning trees are minimally connected).
Minimum spanning tree problem

- Given a connected edge-weighted undirected graph find a spanning tree of minimum weight.
Minimum spanning applications

- Network design
- Cluster analysis
- Cancer imaging
- Cosmology
- Weather data interpretation
- Many others

  - https://personal.utdallas.edu/~besp/teaching/mst-applications.pdf
Lecture 28: Minimum Spanning Trees

- Introduction
- Kruskal’s Algorithm
- Prim’s Algorithm
Kruskal’s algorithm

- Sort edges in ascending order of weight.
- Starting from the one with the smallest weight, add it to the MST $T$ unless doing so would create a cycle.

- Uses a data structure called Union-Find (Chapter 1.5 in book).
- Running time of $|E| \log |V|$ in worst case.
Kruskal's Algorithm Demo
Kruskal's algorithm demo

Consider edges in ascending order of weight.
- Add next edge to tree $T$ unless doing so would create a cycle.
Kruskal's algorithm demo

Consider edges in ascending order of weight.
- Add next edge to tree $T$ unless doing so would create a cycle.

in MST $\rightarrow$ 0-7 0.16

does not create a cycle
Kruskal's algorithm demo

Consider edges in ascending order of weight.

- Add next edge to tree $T$ unless doing so would create a cycle.

```
0-7  0.16
2-3  0.17
```

in MST

```
0-7  0.16
2-3  0.17
```
Kruskal's algorithm demo

Consider edges in ascending order of weight.

- Add next edge to tree $T$ unless doing so would create a cycle.

```
0-7  0.16
2-3  0.17
1-7  0.19
```
Kruskal's algorithm demo

Consider edges in ascending order of weight.

- Add next edge to tree $T$ unless doing so would create a cycle.

<table>
<thead>
<tr>
<th>Edge</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0–7</td>
<td>0.16</td>
</tr>
<tr>
<td>2–3</td>
<td>0.17</td>
</tr>
<tr>
<td>1–7</td>
<td>0.19</td>
</tr>
<tr>
<td>0–2</td>
<td>0.26</td>
</tr>
</tbody>
</table>

in MST does not create a cycle
Kruskal's algorithm demo

Consider edges in ascending order of weight.

- Add next edge to tree $T$ unless doing so would create a cycle.

\[
\begin{array}{l}
0-7 & 0.16 \\
2-3 & 0.17 \\
1-7 & 0.19 \\
0-2 & 0.26 \\
5-7 & 0.28 \\
\end{array}
\]

$in\text{ MST}$ $5-7$ $0.28$

\[\text{does not create a cycle}\]
Kruskal's algorithm demo

Consider edges in ascending order of weight.
  - Add next edge to tree $T$ unless doing so would create a cycle.
Kruskal's algorithm demo

Consider edges in ascending order of weight.

- Add next edge to tree $T$ unless doing so would create a cycle.
Kruskal's algorithm demo

Consider edges in ascending order of weight.

- Add next edge to tree $T$ unless doing so would create a cycle.
Consider edges in ascending order of weight.

- Add next edge to tree $T$ unless doing so would create a cycle.
Kruskal's algorithm demo

Consider edges in ascending order of weight.
- Add next edge to tree $T$ unless doing so would create a cycle.
Kruskal's algorithm demo

Consider edges in ascending order of weight.

- Add next edge to tree $T$ unless doing so would create a cycle.

```
<table>
<thead>
<tr>
<th>Edge</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-7</td>
<td>0.16</td>
</tr>
<tr>
<td>2-3</td>
<td>0.17</td>
</tr>
<tr>
<td>1-7</td>
<td>0.19</td>
</tr>
<tr>
<td>0-2</td>
<td>0.26</td>
</tr>
<tr>
<td>5-7</td>
<td>0.28</td>
</tr>
<tr>
<td>1-3</td>
<td>0.29</td>
</tr>
<tr>
<td>1-5</td>
<td>0.32</td>
</tr>
<tr>
<td>2-7</td>
<td>0.34</td>
</tr>
<tr>
<td>4-5</td>
<td>0.35</td>
</tr>
<tr>
<td>1-2</td>
<td>0.36</td>
</tr>
<tr>
<td>4-7</td>
<td>0.37</td>
</tr>
</tbody>
</table>
```
Consider edges in ascending order of weight.
• Add next edge to tree \( T \) unless doing so would create a cycle.
Kruskal's algorithm demo

Consider edges in ascending order of weight.

- Add next edge to tree $T$ unless doing so would create a cycle.

![Graph diagram showing a minimum spanning tree (MST)]
Kruskal's algorithm demo

Consider edges in ascending order of weight.
- Add next edge to tree $T$ unless doing so would create a cycle.
Kruskal’s algorithm demo

Consider edges in ascending order of weight.

- Add next edge to tree $T$ unless doing so would create a cycle.
Consider edges in ascending order of weight.

- Add next edge to tree $T$ unless doing so would create a cycle.
Kruskal's algorithm demo

Consider edges in ascending order of weight.
- Add next edge to tree $T$ unless doing so would create a cycle.

![Minimum spanning tree diagram]

0–7 0.16
2–3 0.17
1–7 0.19
0–2 0.26
5–7 0.28
1–3 0.29
1–5 0.32
2–7 0.34
4–5 0.35
1–2 0.36
4–7 0.37
0–4 0.38
6–2 0.40
3–6 0.52
6–0 0.58
6–4 0.93

*a minimum spanning tree*
Practice Time
Answer
Lecture 28: Minimum Spanning Trees

- Introduction
- Kruskal’s Algorithm
- Prim’s Algorithm
Prim’s algorithm

- Start with a random vertex (here, 0) and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $|V| - 1$ edges.

- Two versions, lazy and eager. We will see lazy, here...
- Uses min-priority queue.
- Running time of $|E| \log |V|$ in worst case, as well.
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V-1$ edges.

an edge-weighted graph
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

MST edges

0–7
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

![Graph with min weight edges and MST edges highlighted]
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

MST edges

0-7   1-7
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

**MST edges**

- 0–7
- 1–7

```
<table>
<thead>
<tr>
<th>Edges</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>0–2</td>
<td>0.26</td>
</tr>
<tr>
<td>5–7</td>
<td>0.28</td>
</tr>
<tr>
<td>1–3</td>
<td>0.29</td>
</tr>
<tr>
<td>1–5</td>
<td>0.32</td>
</tr>
<tr>
<td>2–7</td>
<td>0.34</td>
</tr>
<tr>
<td>1–2</td>
<td>0.36</td>
</tr>
<tr>
<td>4–7</td>
<td>0.37</td>
</tr>
<tr>
<td>0–4</td>
<td>0.38</td>
</tr>
<tr>
<td>6–0</td>
<td>0.58</td>
</tr>
</tbody>
</table>
```
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

MST edges

0–7  1–7  0–2
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

MST edges

0–7  1–7  0–2

Edges with exactly one endpoint in $T$ (sorted by weight)

2–3  0.17
5–7  0.28
1–3  0.29
1–5  0.32
4–7  0.37
0–4  0.38
6–2  0.40
6–0  0.58
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

**MST edges**
- 0-7
- 1-7
- 0-2
- 2-3
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

Min weight edge with exactly one endpoint in $T$

Edges with exactly one endpoint in $T$ (sorted by weight)

MST edges

0–7  1–7  0–2  2–3
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

**MST edges**

0–7  1–7  0–2  2–3  5–7
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

**MST edges**

0–7  1–7  0–2  2–3  5–7
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

MST edges

0–7  1–7  0–2  2–3  5–7  4–5
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

MST edges

0–7 1–7 0–2 2–3 5–7 4–5
Prim's algorithm demo

- Start with vertex 0 and greedily grow tree $T$.
- Add to $T$ the min weight edge with exactly one endpoint in $T$.
- Repeat until $V - 1$ edges.

MST edges

0–7 1–7 0–2 2–3 5–7 4–5 6–2
Practice Time
Answer...
Lecture 28: Minimum Spanning Trees

- Introduction
- Kruskal’s Algorithm
- Prim’s Algorithm
Readings:

- Textbook: Chapter 4.3 (Pages 604-629)
- Website: https://algs4.cs.princeton.edu/43mst/

Practice Problems:

https://visualgo.net/en/mst
Readings:

- Textbook: Chapter 4.3 (Pages 604-629)
- Website:
  - [https://algs4.cs.princeton.edu/43mst/](https://algs4.cs.princeton.edu/43mst/)

Practice Problems:

[https://visualgo.net/en/mst](https://visualgo.net/en/mst)