Searching trees, weighted graphs, and shortest paths

Steph van Willigenburg
Math 442-201 2019WT2

24 March 2020

Spanning trees of the complete graph

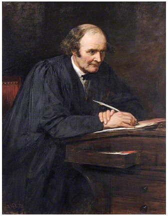
Last time:

Theorem

There are

$$n^{(n-2)}$$

non-isomorphic labelled trees with $n \geq 2$ vertices.



Wikimedia

This time:

Corollary

$$\tau(K_n) = n^{n-2}$$

Proof of $\tau(K_n) = n^{n-2}$

Label vertices of K_n with $1, 2, \ldots, n$, and $\mathcal{T}$ be set of all non-iso labelled trees, n vertices.

1. Each spanning tree of K_n is now a labelled tree so

$$\tau(K_n) \leq |\mathcal{T}|.$$

2. Each $T \in \mathcal{T}$ is iso to a spanning tree of K_n, $\hat{T}$, by the iso

$$\text{edge } ij \in E(T) \mapsto \text{edge } ij \in E(\hat{T})$$

so

$$\tau(K_n) \geq |\mathcal{T}|.$$

3. By Cayley's theorem

$$\tau(K_n) = |\mathcal{T}| = n^{n-2}. \quad \square$$

Rooted trees

Definition

Let T be a tree. We say it is rooted if it has a distinguished vertex v. We call v the root.

Note: We draw root at top, leaves at bottom.

Example

Breadth first search (BFS)

Given a rooted tree T with root v.

1. From v visit all vertices path length 1 away from v.
2. $i := i + 1$.
3. From v visit all vertices path length i away from v.
4. Repeat 2 and 3 until all vertices visited.

Note: For consistency $\rightarrow$ go left.

Example

Depth first search (DFS)

Given a rooted tree T with root v.

1. From v visit v' path length 1 away from v.
2. Visit v'' not already visited, path length 1 away from v'.
3. $v' := v''$.
4. Repeat 2 and 3 until can go no further. Back-track to last vertex choice v''' and $v' := v'''$.
5. Repeat 2, 3 and 4 until all vertices visited.

Note: For consistency $\rightarrow$ go left.

Example

Weighted graphs

Definition

Let G be a graph. We say G is weighted if each $e \in E(G)$ has a weight $w(e)$. The sum of all weights is the weight of G $W(G)$.

Note: Normally positive integers.

Example

$G =$

A is distance from B and $W(G) =$

Shortest path problem - applied BFS

Definition

Let G be a weighted graph, and $u, v \in V(G)$. A path from u to v with smallest sum of edge weights is called a shortest path from u to v.

Example

The shortest path from A to D is

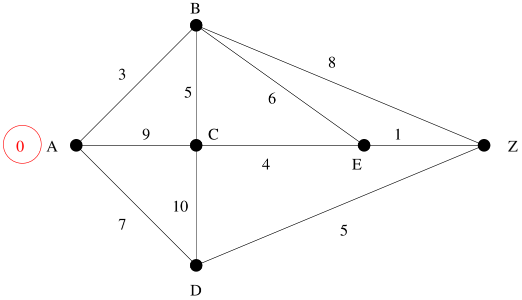
Dijkstra's algorithm for shortest path

Let G be a connected weighted graph and $\ell(v)$ denote label of v. Compute shortest distance from vertices A to Z.

- Let $\ell(A) = 0$. Make it permanent. Assign temporary labels $\ell(A) + d$ to all adjacent to A distance d away. Make smallest temporary into permanent.

Note: Permanent labels cannot be changed.

Example



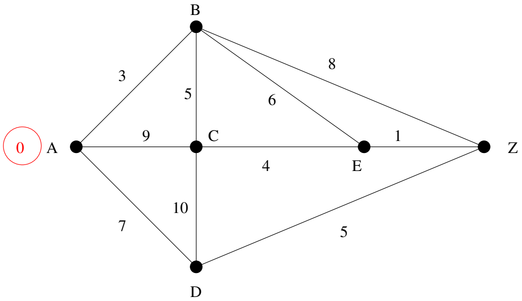
Dijkstra's algorithm for shortest path

Let G be a connected weighted graph and $\ell(v)$ denote label of v. Compute shortest distance from vertices A to Z.

- If v just permanent. Assign temporary labels $\ell(v) + d$ to all adjacent to v distance d away if smaller label than present or no label. Make smallest temporary into permanent.
- Repeat until all vertices have labels that are permanent.

Note: Permanent labels cannot be changed.

Example



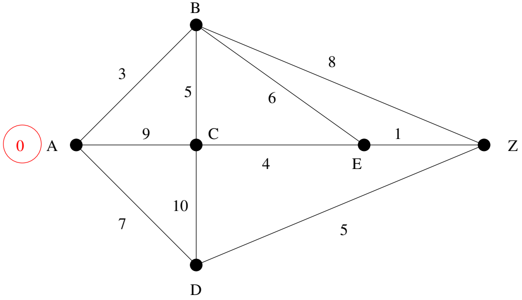
Dijkstra's algorithm for shortest path

Let G be a connected weighted graph and $\ell(v)$ denote label of v. Compute shortest distance from vertices A to Z.

- The shortest distance from A to Z is label at Z. The shortest path found by starting at Z, include edge weight d between v, w if

$$\ell(w) - \ell(v) = d.$$

Example Shortest distance = Shortest path =



In summary

- Spanning trees of the complete graph
- Breadth first search
- Depth first search
- Weighted graphs
- Dijkstra's algorithm for shortest path

Thanks! See you next time!