Introduction to Trees

CMPT 125
Mo Chen
SFU Computing Science
16/3/2020

Lecture 26

Today:

- Graphs and Rooted Trees
- Tree Anatomy
- Expression Trees

Graphs

Also: *nodes*

Also: *arcs*

A *graph* depicts the relationships among a collection

- the collection is known as the *vertices* (depicted by dots or junctions)
- the relations are known as the *edges* (depicted by lines between dots)

What are the vertices and edges in these common graphs?

Graphs

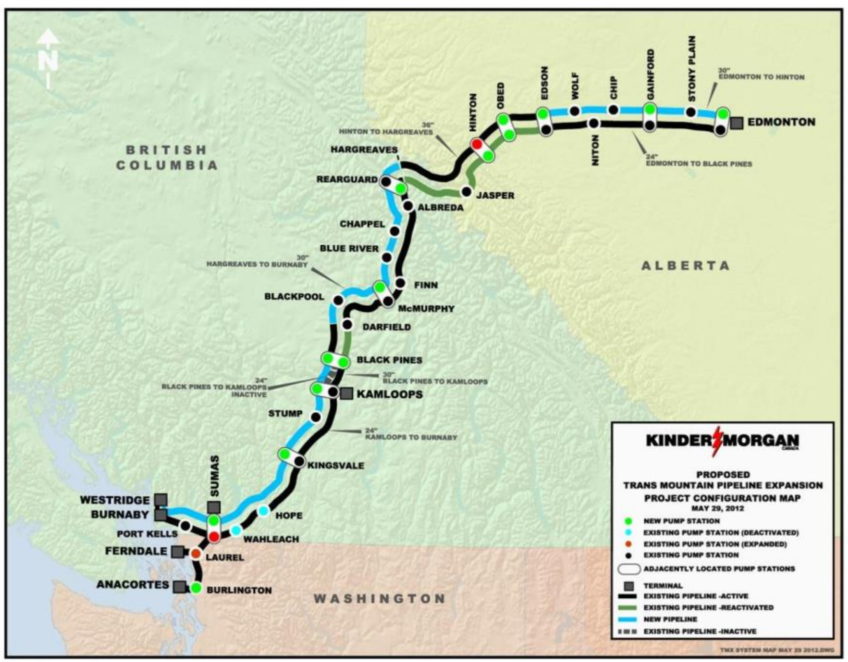
A *graph* depicts the relationships among a collection

- the collection is known as the *vertices* (depicted by dots or junctions)
- the relations are known as the *edges* (depicted by lines between dots)

Also: *nodes*

Also: *arcs*

What are the vertices and edges in these common graphs?



Gas Pipe System

Vertices?

- Stations

Edges?

- Pipes

Graphs

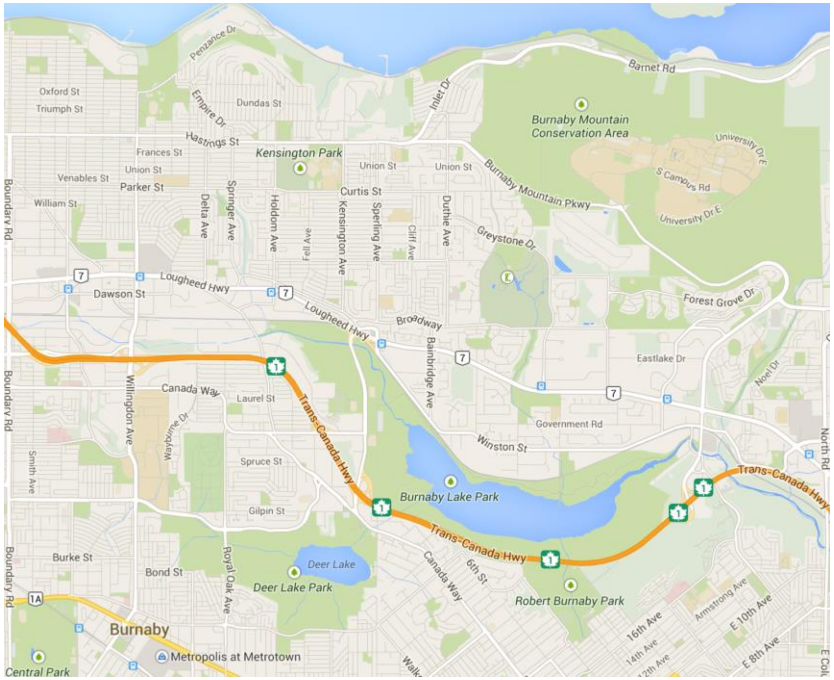
Also: *nodes*

Also: *arcs*

A *graph* depicts the relationships among a collection

- the collection is known as the *vertices* (depicted by dots or junctions)
- the relations are known as the *edges* (depicted by lines between dots)

What are the vertices and edges in these common graphs?



Road Map

Vertices?

- Intersections

Edges?

- Roads

Graphs

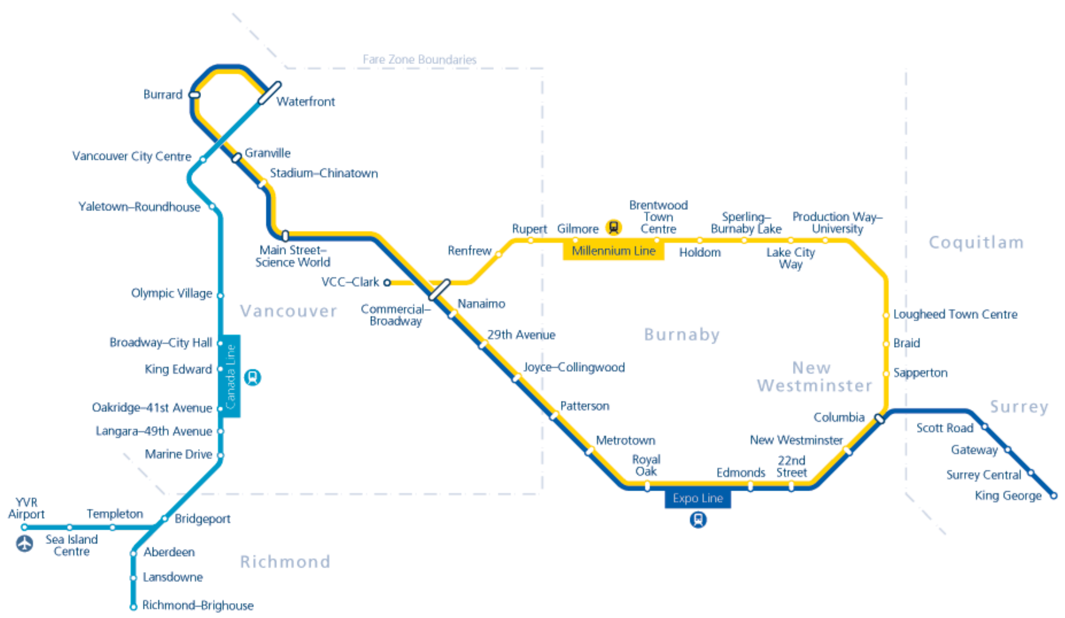
Also: *nodes*

Also: *arcs*

A *graph* depicts the relationships among a collection

- the collection is known as the *vertices* (depicted by dots or junctions)
- the relations are known as the *edges* (depicted by lines between dots)

What are the vertices and edges in these common graphs?



Skytrain Map

Vertices?

- Stations

Edges?

- Tracks

Graphs

Also: *nodes*

Also: *arcs*

A *graph* depicts the relationships among a collection

- the collection is known as the *vertices* (depicted by dots or junctions)
- the relations are known as the *edges* (depicted by lines between dots)

What are the vertices and edges in these common graphs?

Social Media

Vertices?
- People

Edges?
- Friends

Graphs

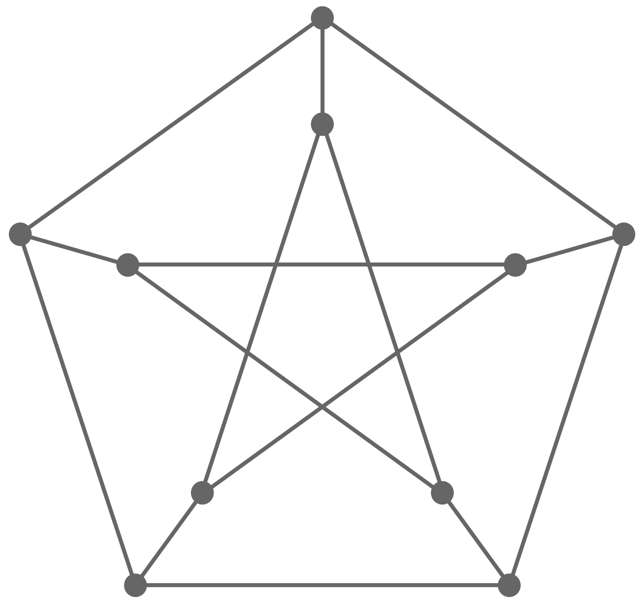
Also: *nodes*

Also: *arcs*

A *graph* depicts the relationships among a collection

- the collection is known as the *vertices* (depicted by dots or junctions)
- the relations are known as the *edges* (depicted by lines between dots)

What are the vertices and edges in these common graphs?

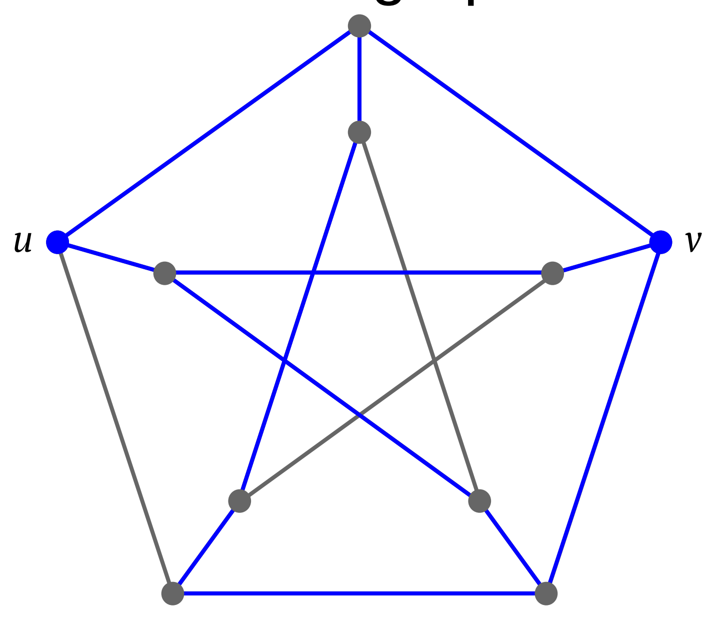


Paths and Connectivity

A *path* from vertex u to vertex v is sequence of edges which connect a sequence of vertices between them.

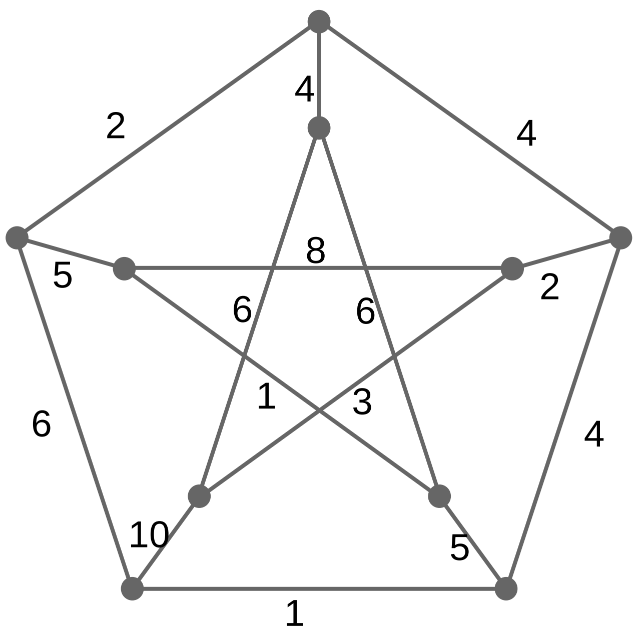
A graph is *connected* if each pair of vertices has a path

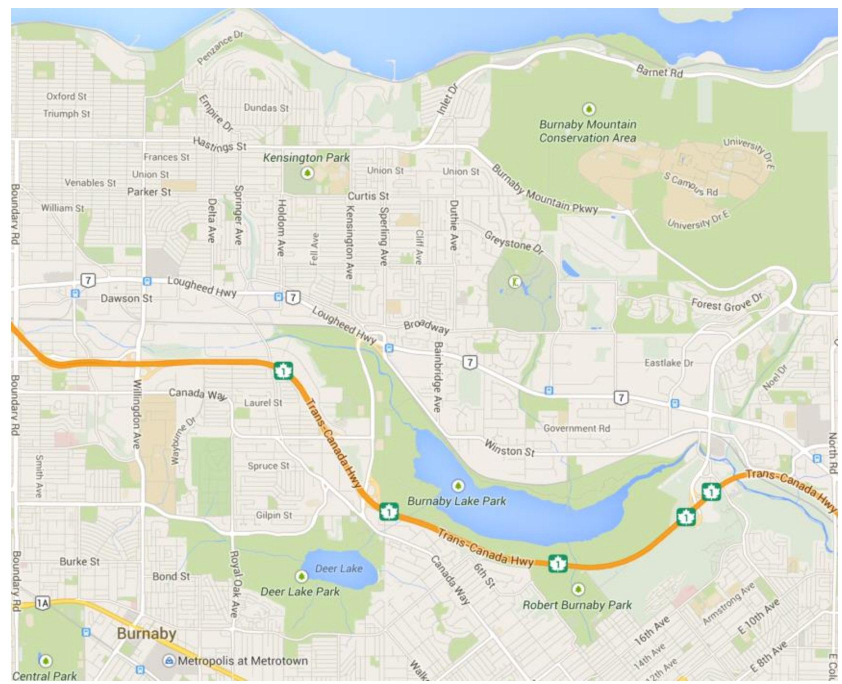
Q. When might disconnected graphs arise?



Weighted Graphs

- Each edge has an associated weight
- Weight can represent distance or cost





Representing Graphs

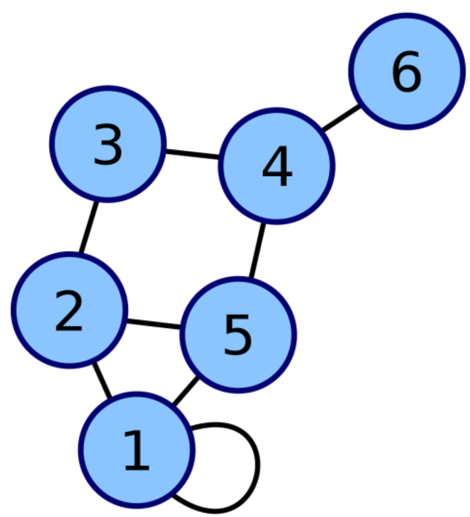
- Graphs as a collection of node objects

```
template <class T>
class node {
    private:
        // a node holds data of a generic type
        T data;

        // list of nodes that this node is connected to
        LL<node> * connections;

     public:
        ...
        ...
}
```

- Adjacency matrix

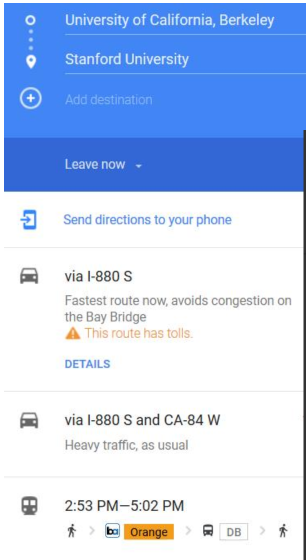


$$\begin{pmatrix} 2 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

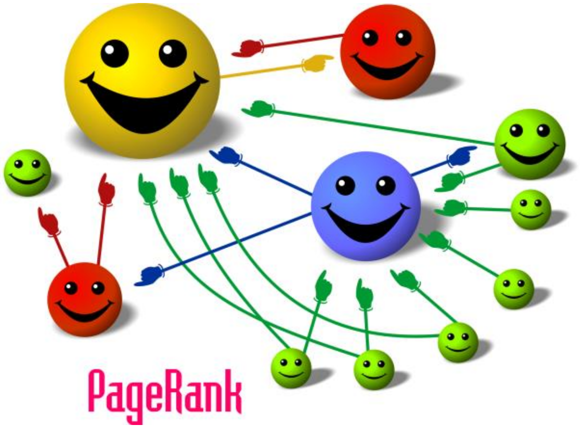
Two (Extremely) Useful Classes of Algorithms

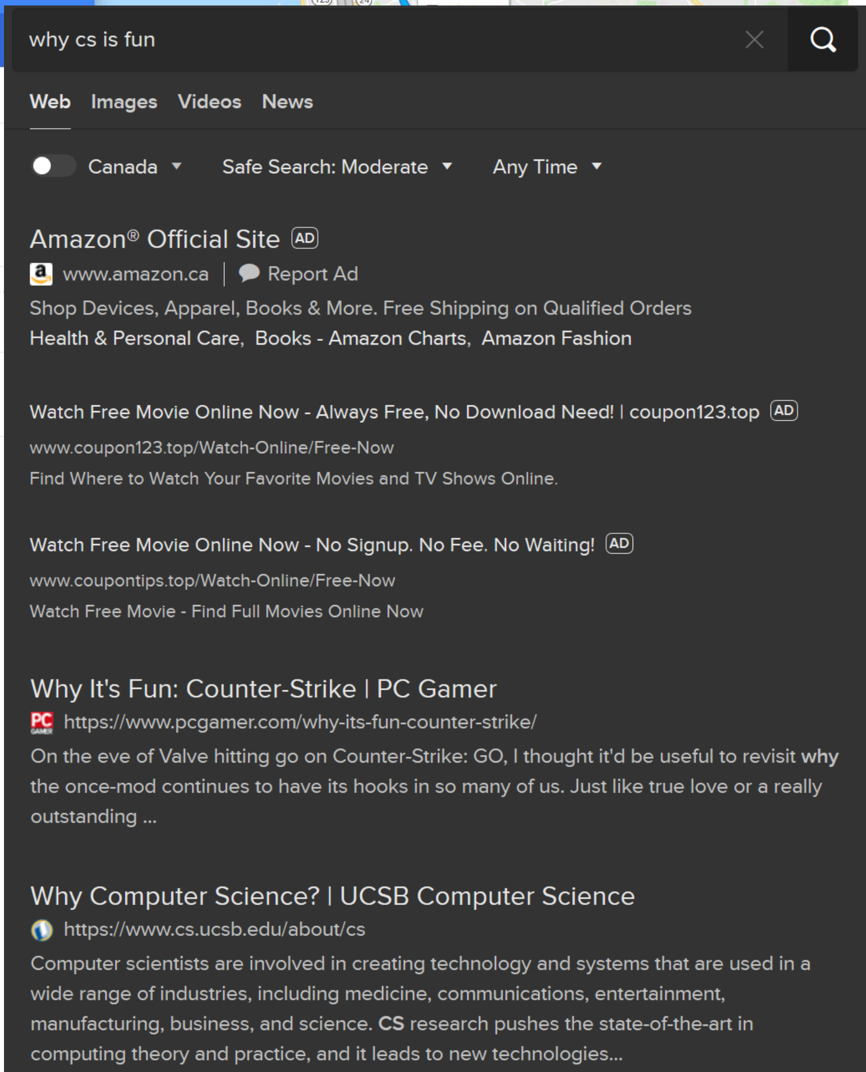
- Shortest Path Algorithms (A*, Dijkstras, etc.)

50	51	54	57	65	69
48	52	51	58	64	64
47	53	52	54	60	63
45	48	49	56	64	61
44	45	51	57	58	60
42	46	50	52	58	59



- Search engines (e.g. PageRank)





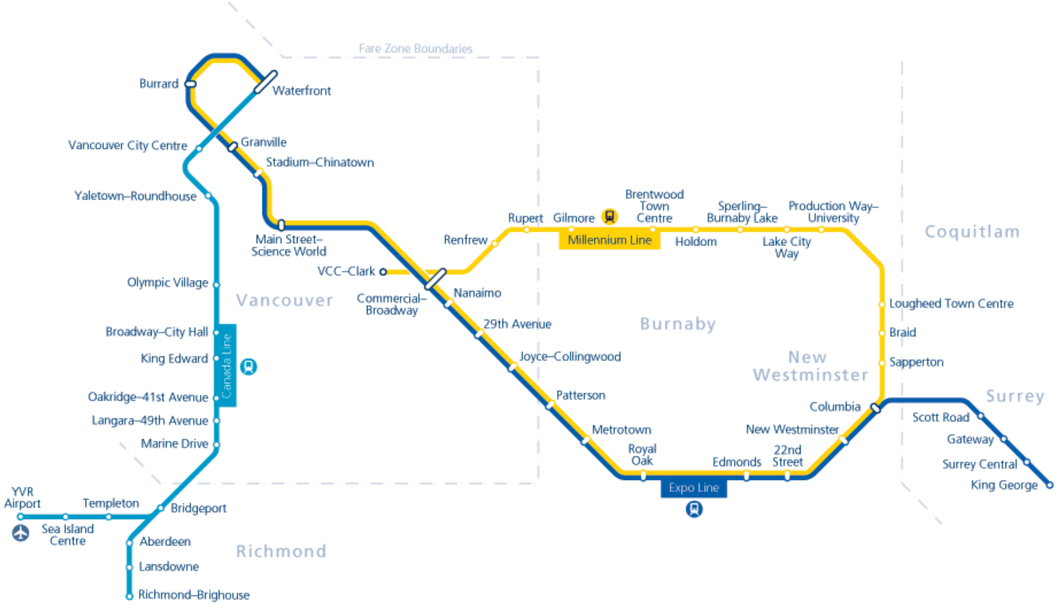
Trees

A *tree* is a minimally connected graph

- all vertices connected
- no cycles

E.g., Skytrain is almost a tree.

- Q. How can we make it into a tree?



Example: Building a Network

The Problem: Join a community with the least amount of infrastructure.

- roads
- train tracks
- ethernet trunk

Strategy:

- Use the smallest tree, a *minimum spanning tree*

Algorithm:

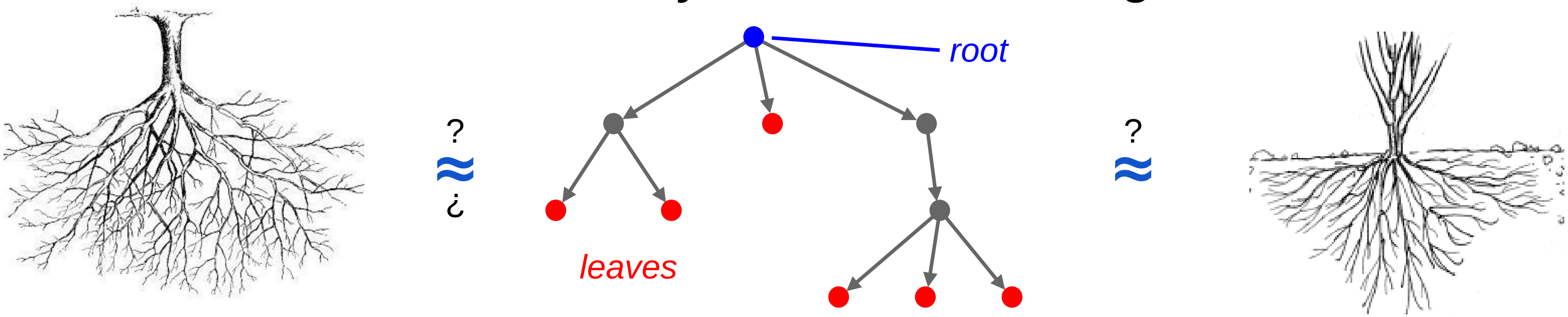
- Repeatedly select the smallest connection that doesn't form a cycle.
- Kruskal's Algorithm

Directed Graphs and Rooted Trees

Sometimes denote a *direction* on an edge:

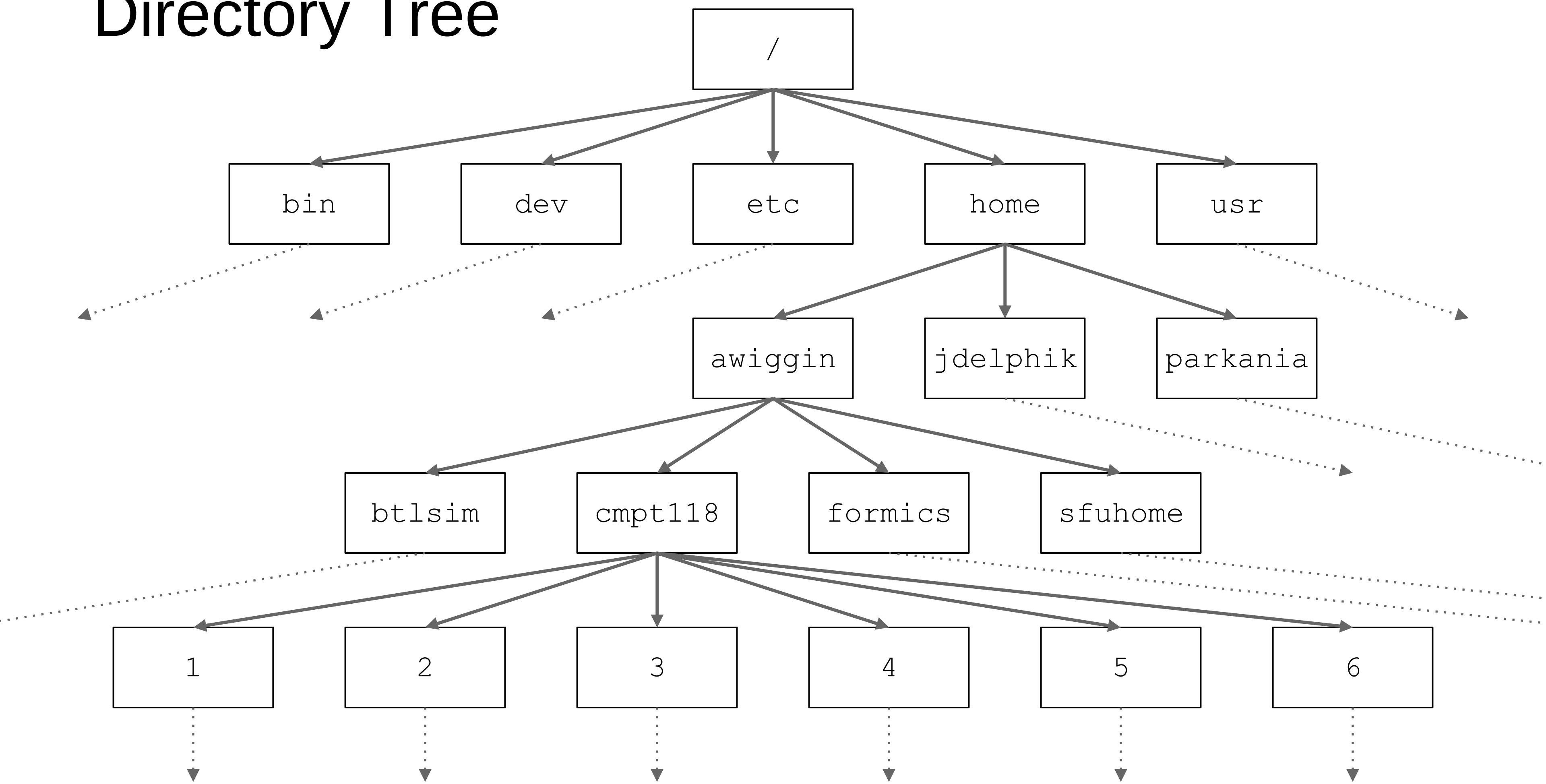
- one way traffic
- pipeline flow
- web links
- dependency
- A “likes” B

A *rooted tree* is a directed version of the tree where exactly one vertex, called the *root*, has no inbound edges, but all other vertices have exactly one inbound edge.



Rooted Tree Examples

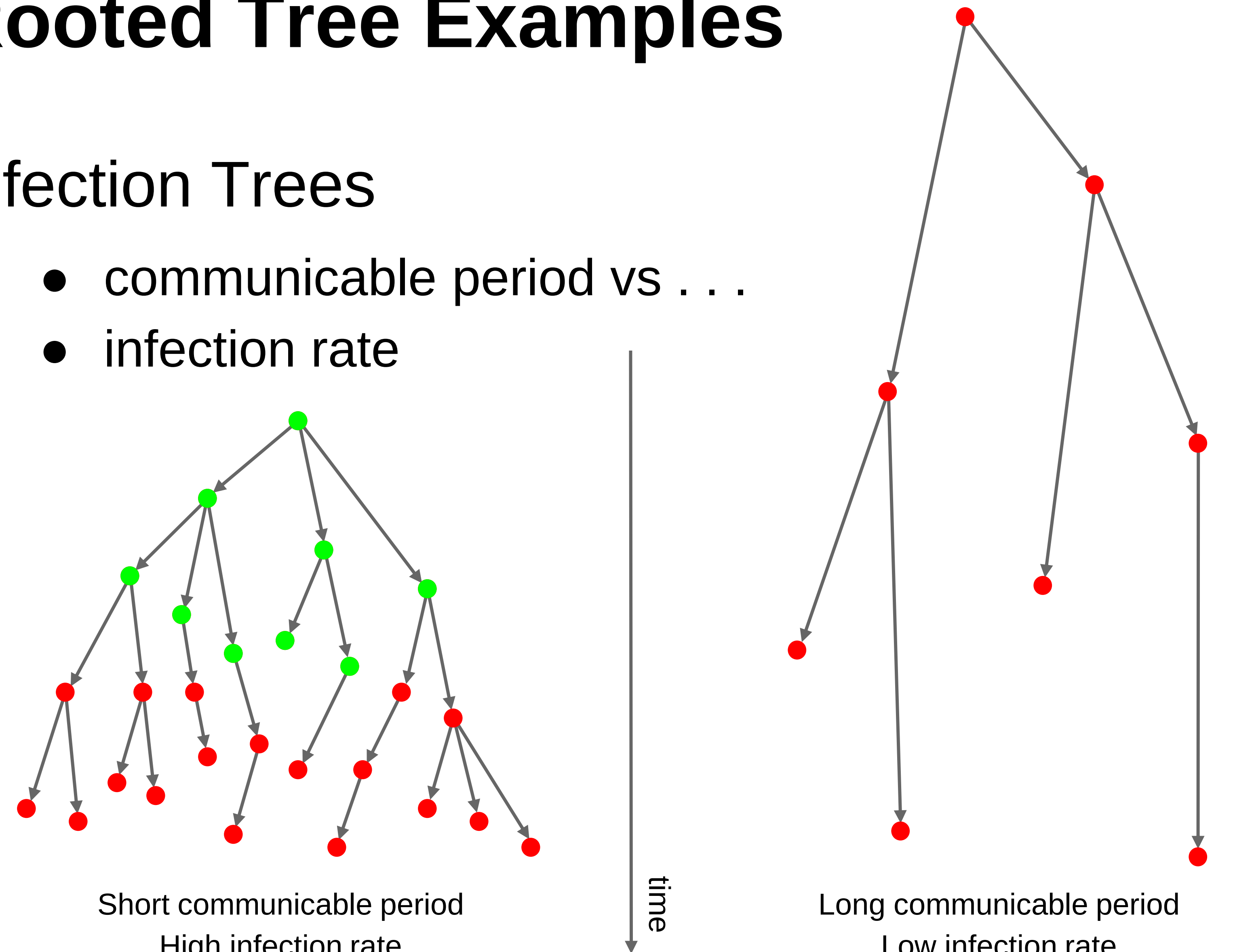
Directory Tree



Rooted Tree Examples

Infection Trees

- communicable period vs . . .
- infection rate



Short communicable period
High infection rate

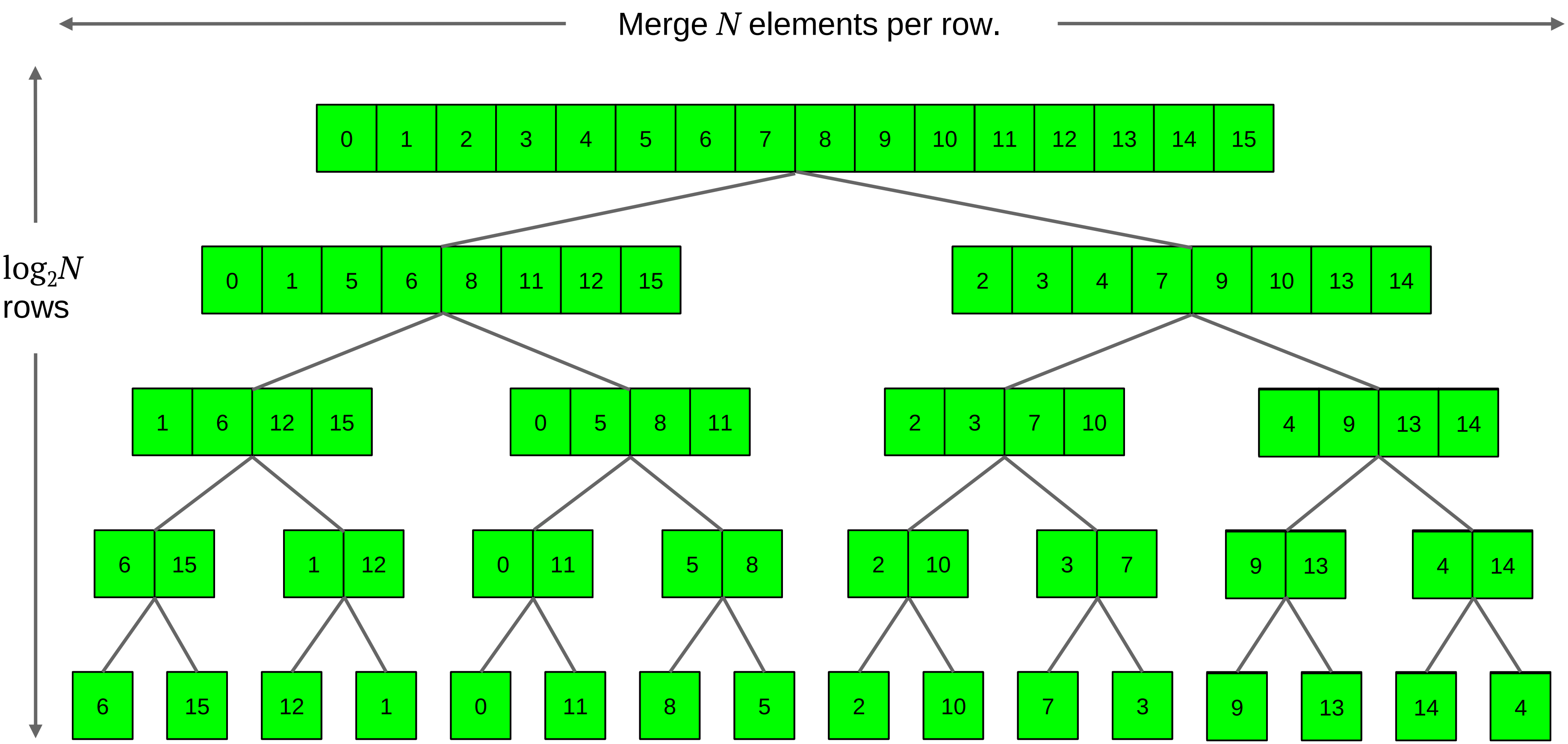
Long communicable period
Low infection rate

Mergesort

Recursion tree:

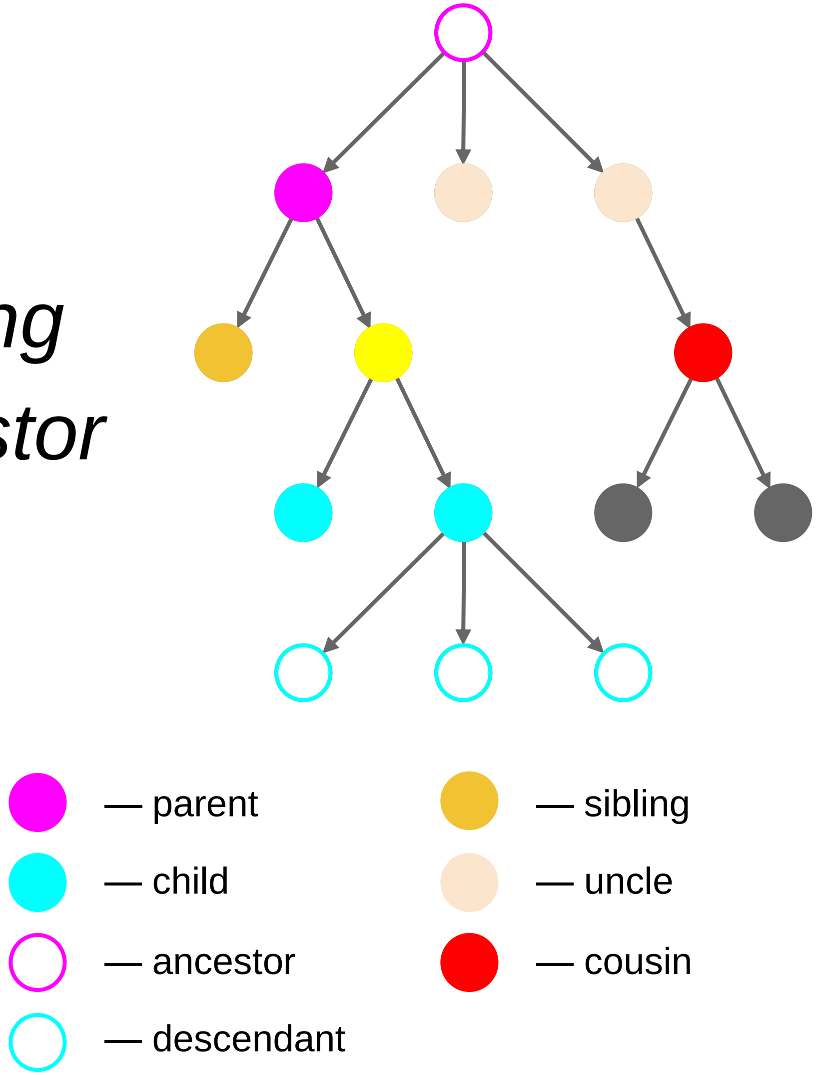
- $O(N)$ work per row
- $O(\log N)$ rows

$\Rightarrow$ $O(N \log N)$ running time



Tree Terminology

- Inbound edge: *parent*
- Outbound edge: *child*
- Common parent: *sibling*
- On path to root: *ancestor*
- Path away from root: *descendant*
- No child: *leaf*
- At least one child: *internal node*

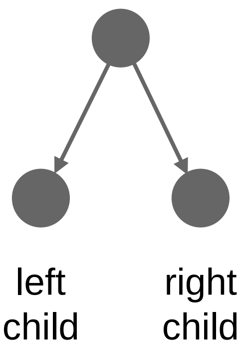


Binary Trees

An *m-ary tree* is a tree in which each vertex has at most *m* children.

The common name for a 2-ary tree is *binary tree*.

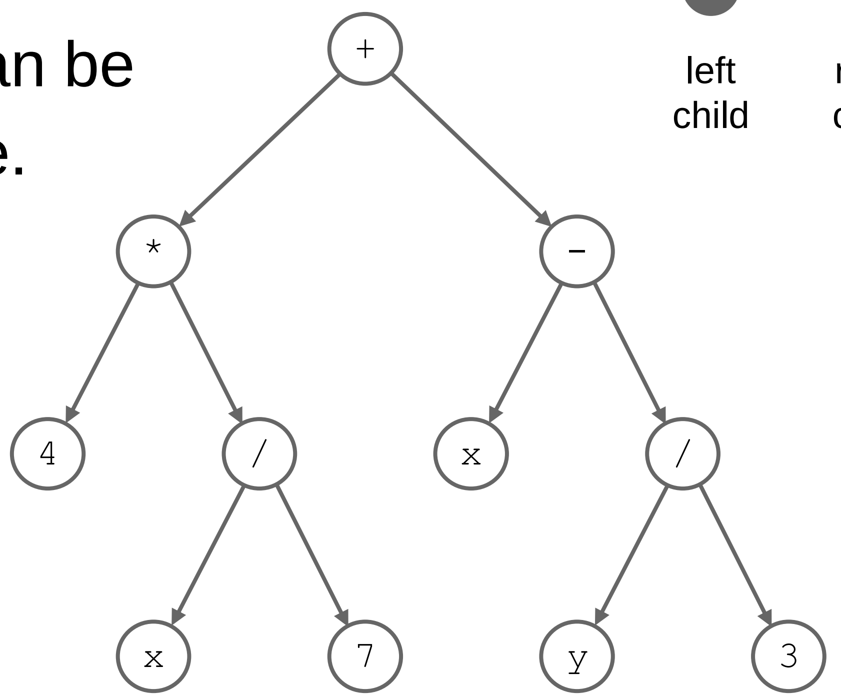
- denote its children as the *left child* and *right child*



Any algebraic expression can be represented as a binary tree.

Represents the expression:

`(4*(x/7))+(x-(y/3))`



Q. How do you evaluate an expression tree?