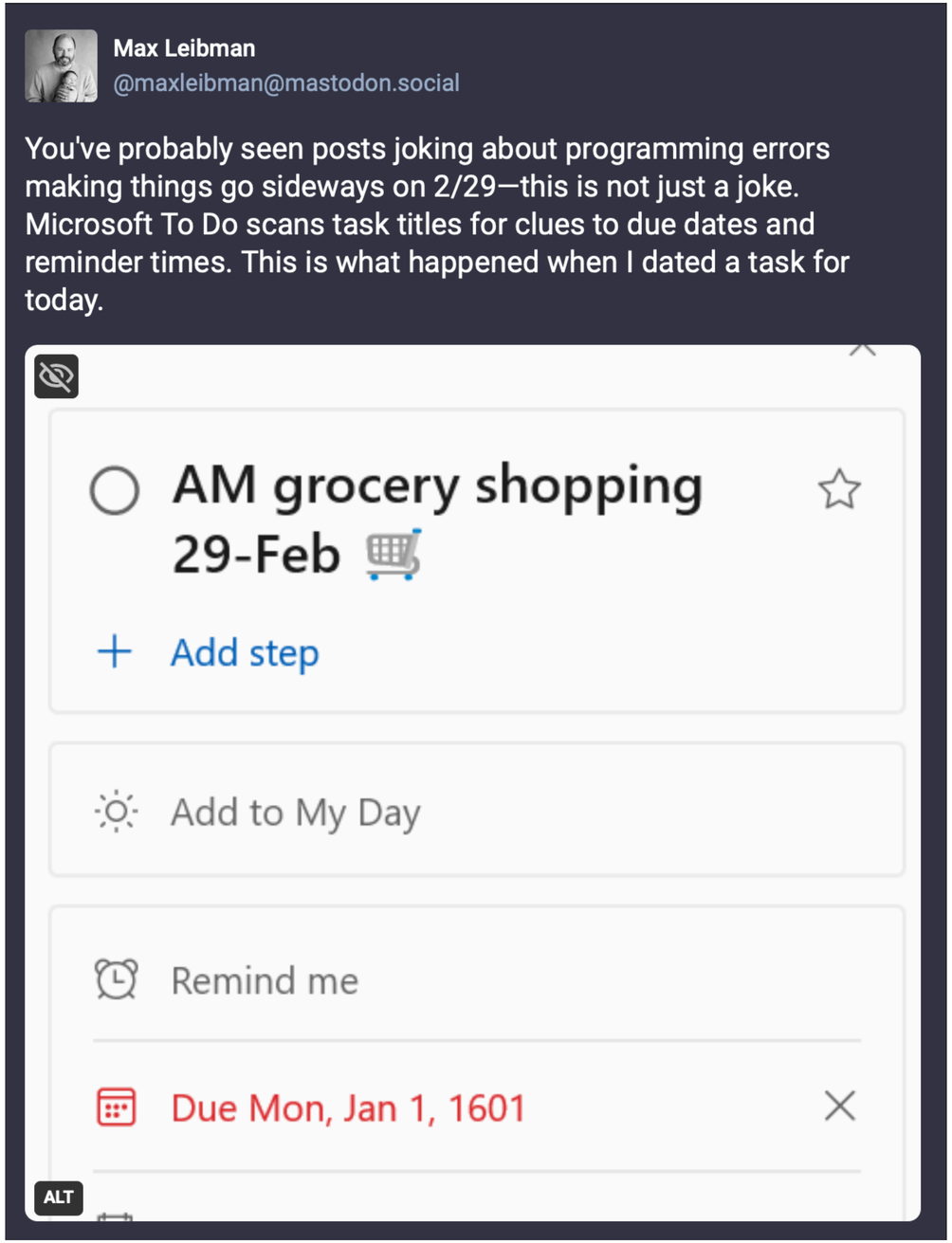
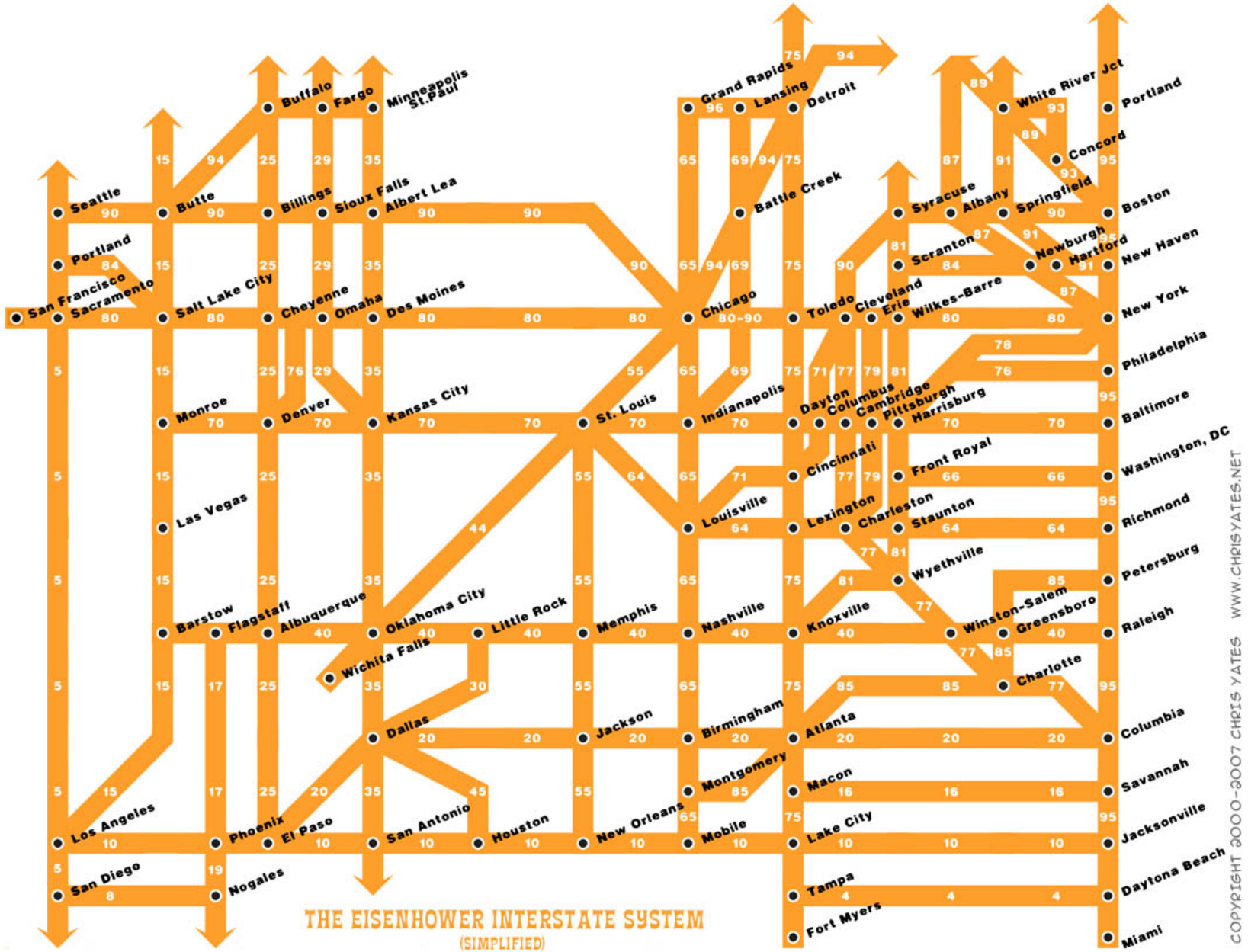
CMPU 101 § 53 · Computer Science I

Graphs and Simulation

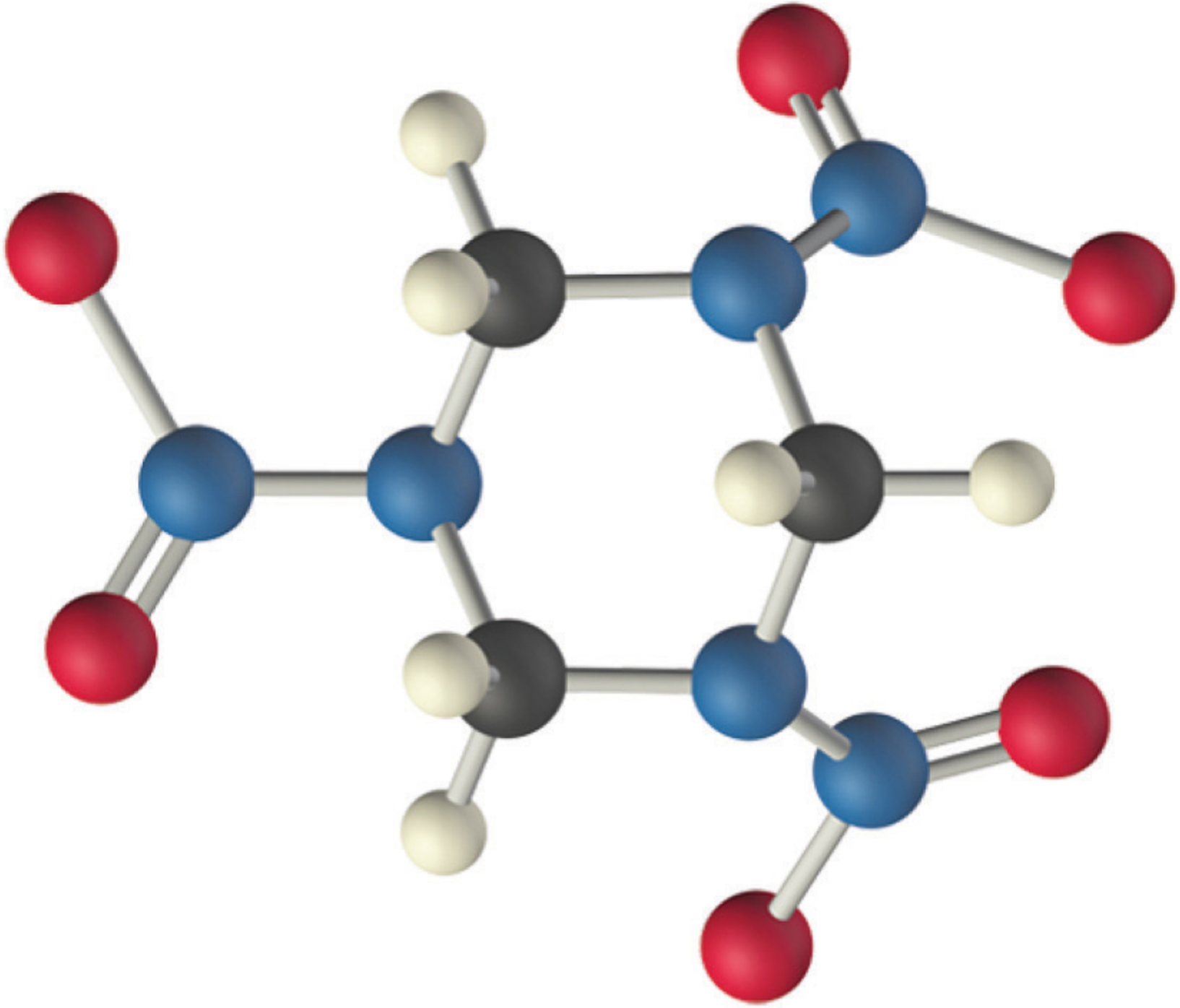
29 February 2024



This is why we write tests.


Seattle
Portland
San Francisco
Sacramento
San Diego
Los Angeles
Las Vegas
Barstow
Flagstaff
Phoenix
Nogales
El Paso
Albuquerque
Monroe
Butte
Salt Lake City
Denver
Cheyenne
Billings
Buffalo
Fargo
Sioux Falls
Minneapolis St.Paul
Albert Lea
Omaha
Des Moines
Kansas City
Oklahoma City
Wichita Falls
Dallas
San Antonio
Houston
Little Rock
Memphis
Jackson
New Orleans
St. Louis
Chicago
Indianapolis
Louisville
Nashville
Birmingham
Montgomery
Mobile
Grand Rapids
Lansing
Detroit
Battle Creek
Toledo
Cleveland
Erie
Dayton
Columbus
Cincinnati
Lexington
Knoxville
Atlanta
Macon
Lake City
Tampa
Fort Myers
Syracuse
Albany
Scranton
Wilkes-Barre
Pittsburgh
Cambridge
Harrisburg
Wyethville
Front Royal
Charleston
Staunton
Winston-Salem
Greensboro
Charlotte
White River Jct
Springfield
Concord
Newburgh
Hartford
Portland
Boston
New Haven
New York
Philadelphia
Baltimore
Washington, DC
Richmond
Petersburg
Raleigh
Columbia
Savannah
Jacksonville
Daytona Beach
Miami
THE EISENHOWER INTERSTATE SYSTEM
(SIMPLIFIED)
COPYRIGHT 2000-2007 CHRIS YATES WWW.CHRISYATES.NET

Highways



Chemical bonds

Dystopian proxies for friendship

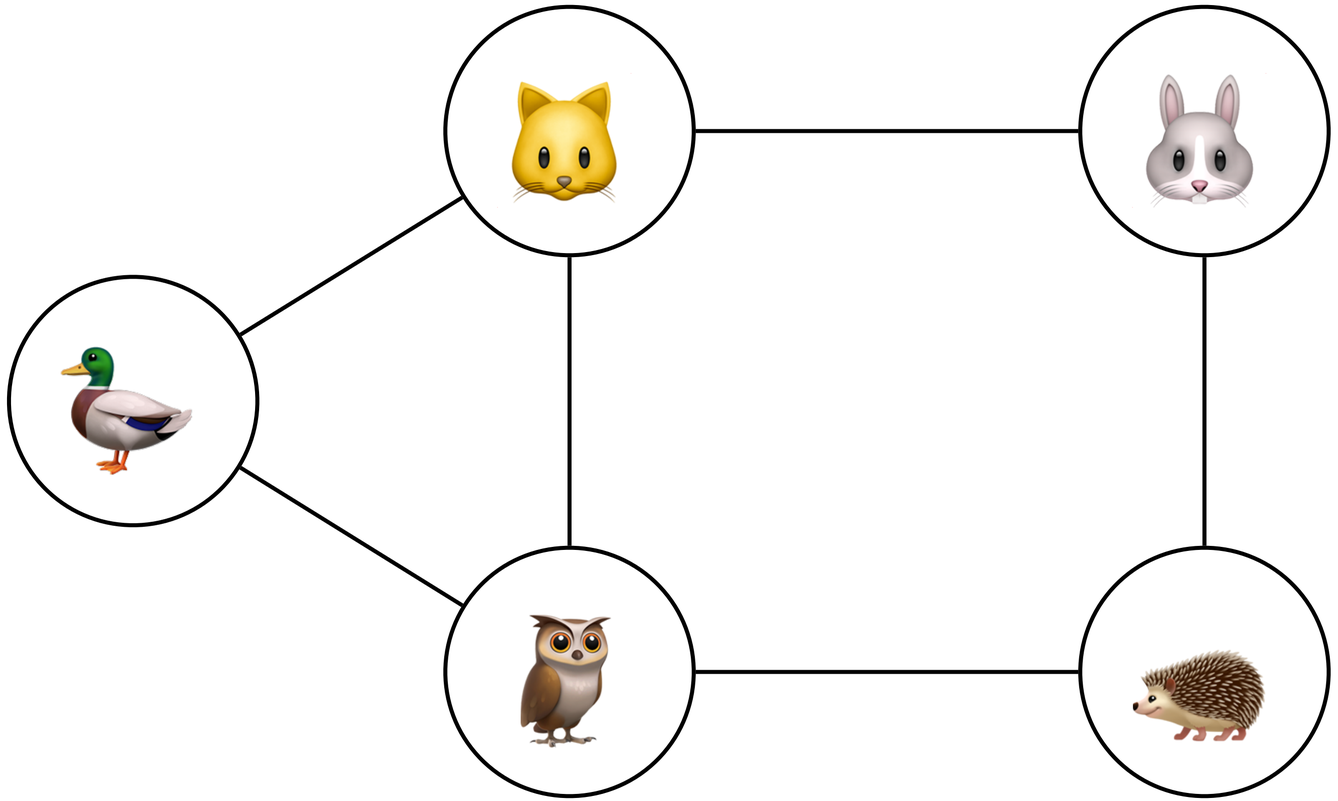
Each of these structures consists of

a collection of objects and

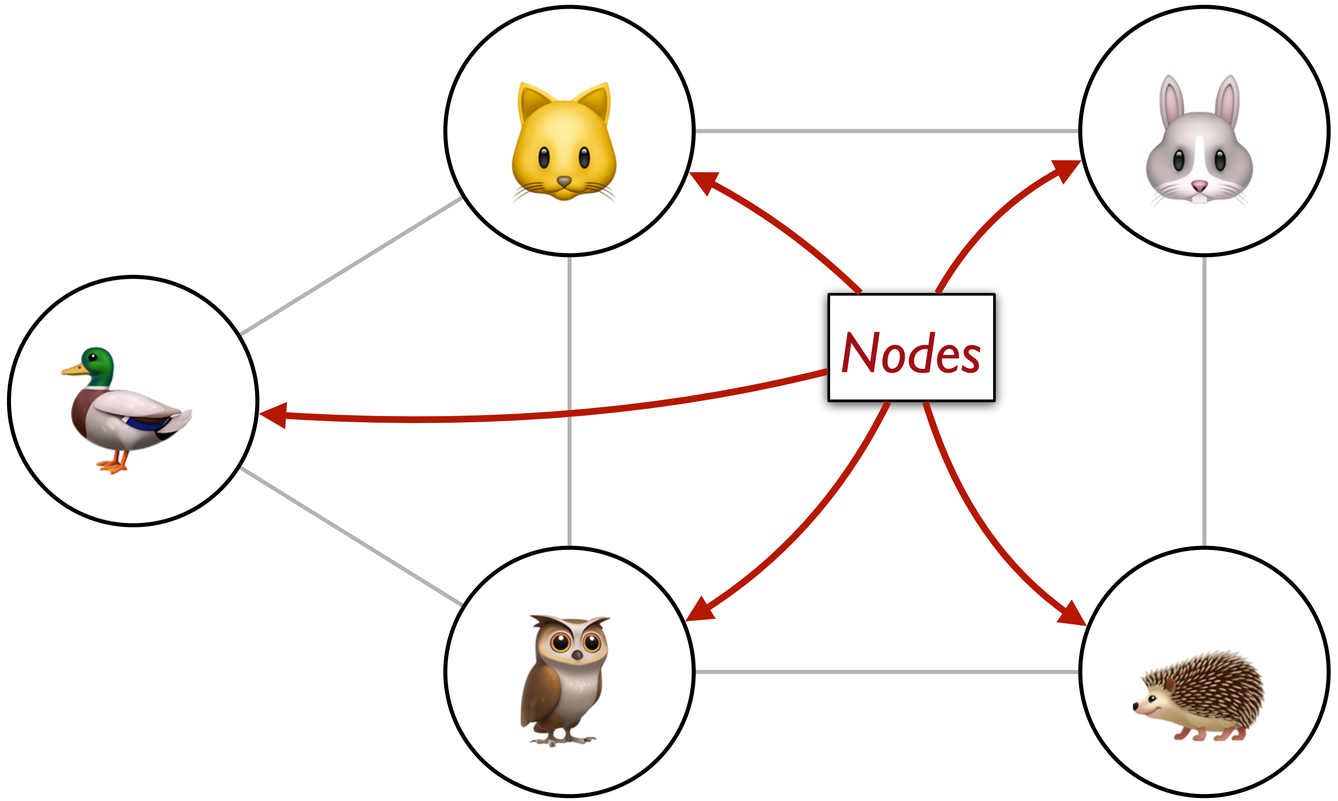
links between those objects.

We'd like to find a general framework for describing these objects and their properties.

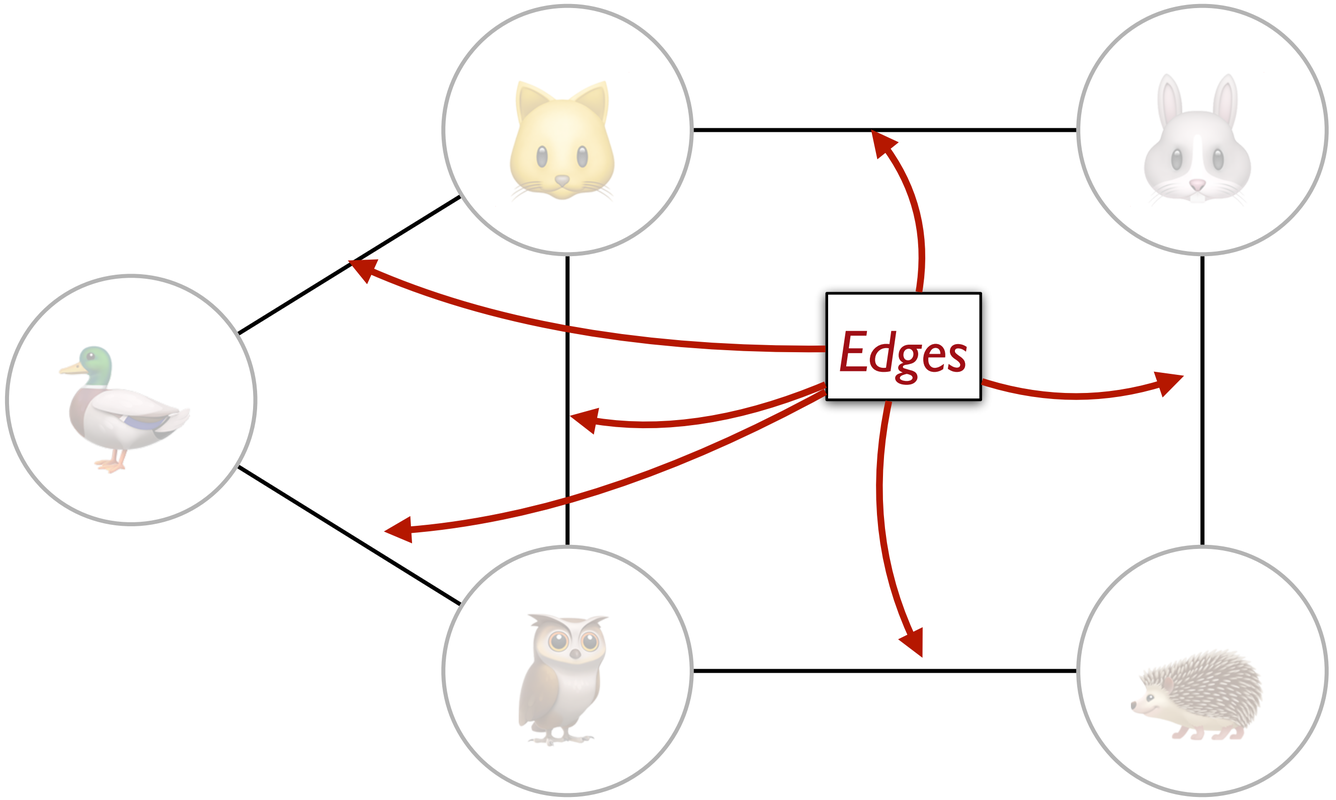
A *graph* is a mathematical structure for representing relationships between entities.



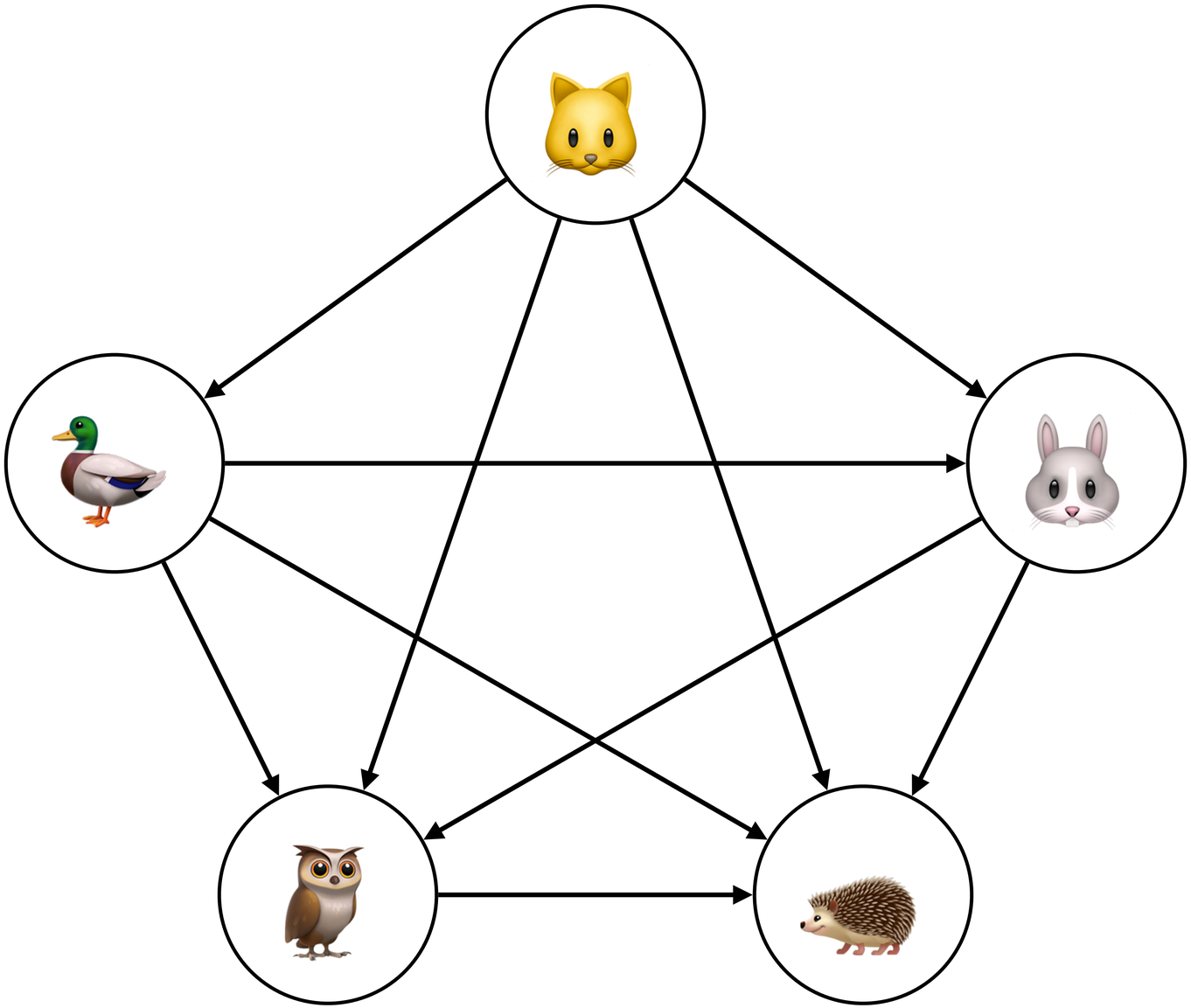
A graph consists of a set of *nodes* (or *vertices*) connected by *edges* (or *arcs*).



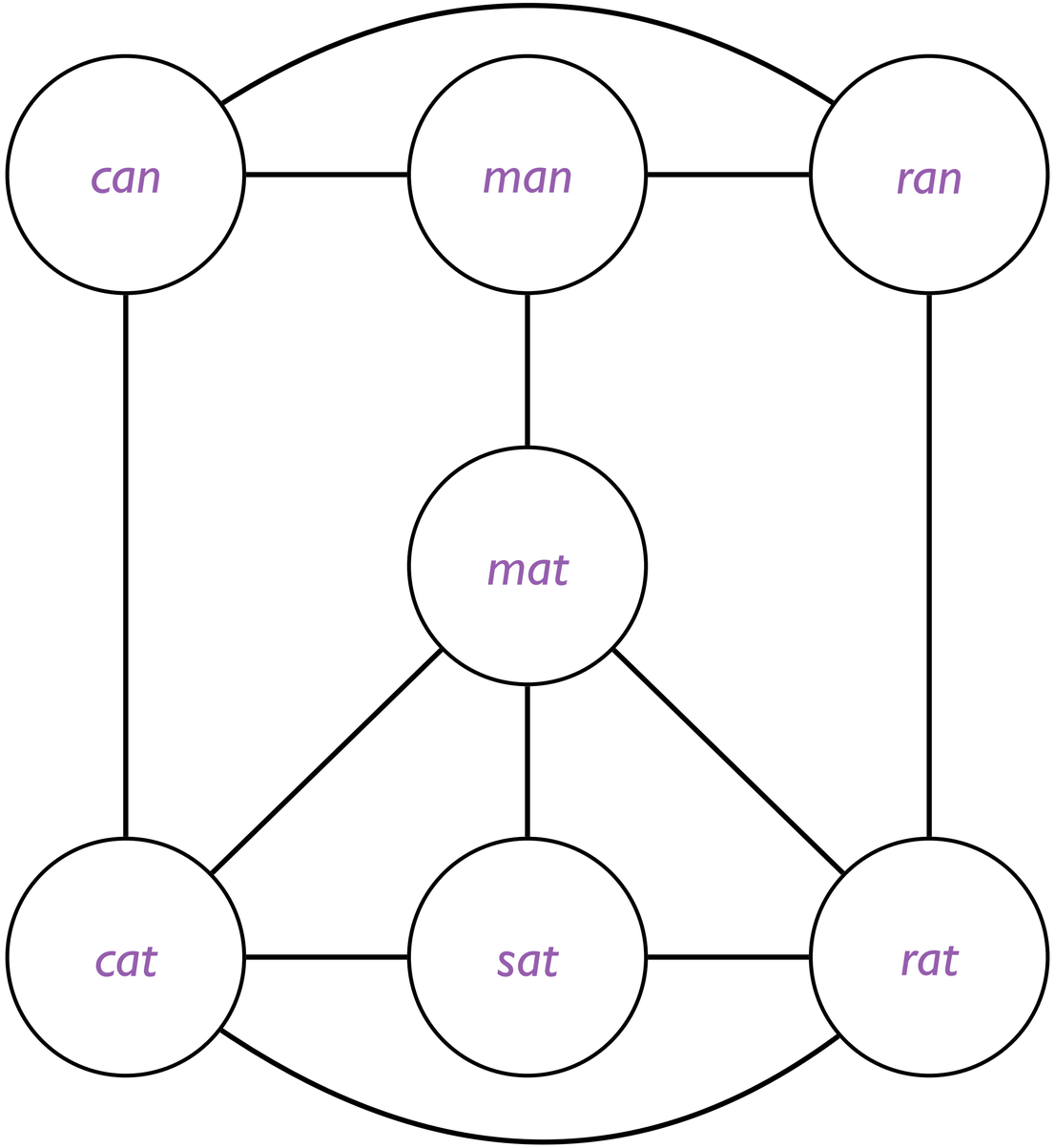
A graph consists of a set of *nodes* (or *vertices*) connected by *edges* (or *arcs*).



A graph consists of a set of *nodes* (or *vertices*) connected by *edges* (or *arcs*).

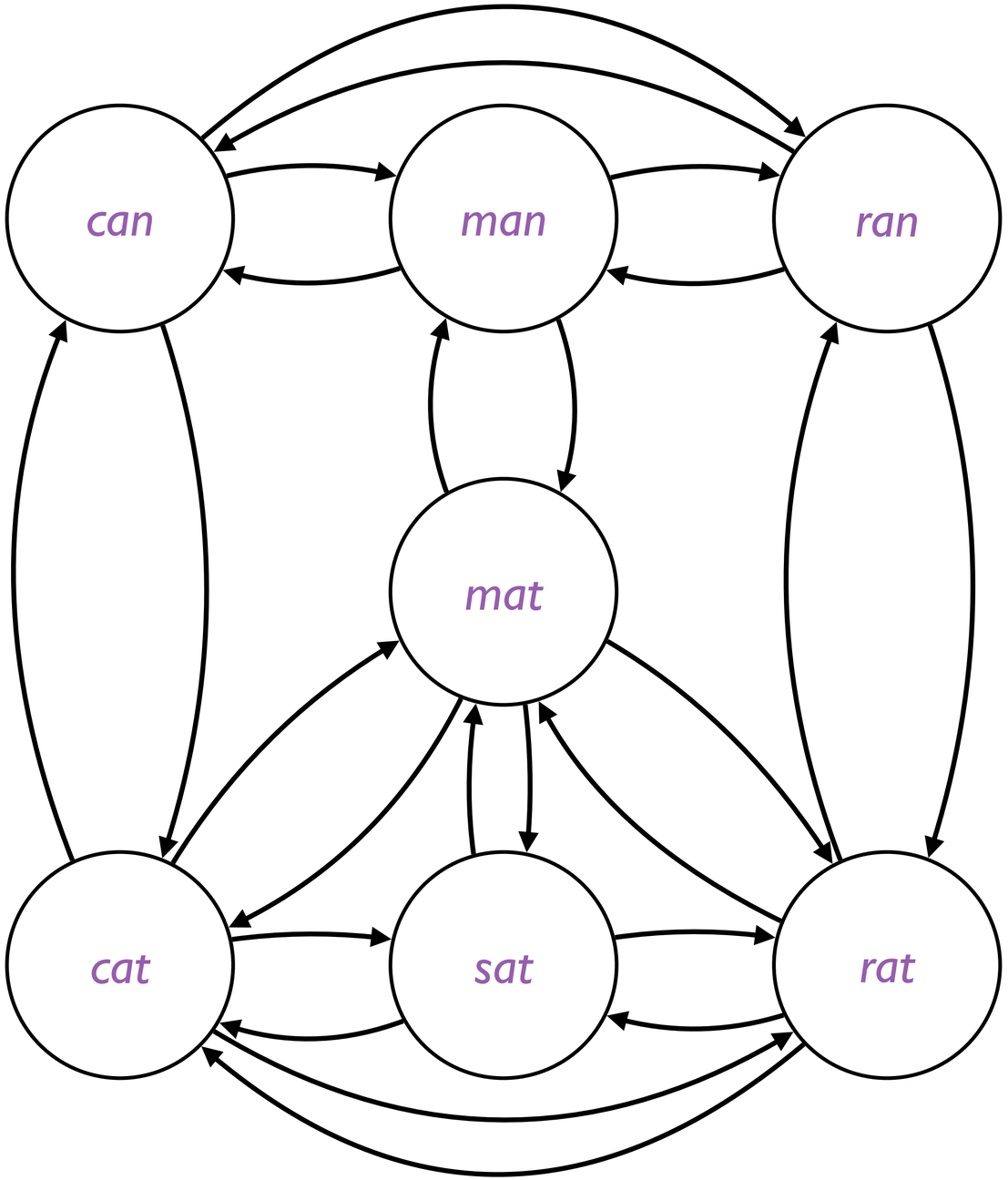


Some graphs are *directed*.



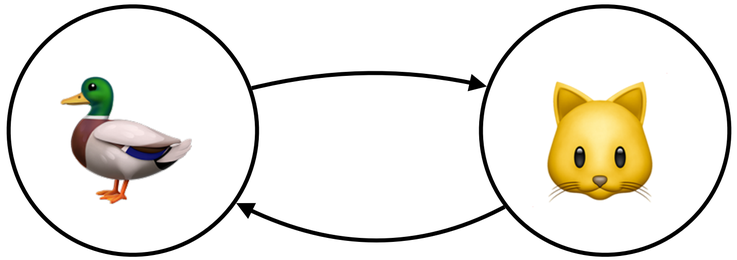
Some graphs are *undirected*.

Every undirected graph can also be represented as a directed graph, albeit with twice the edges.

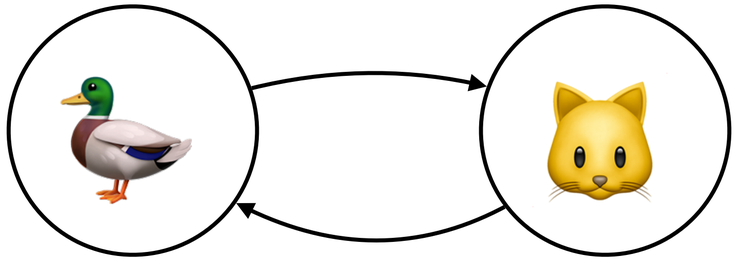

can
man
ran
mat
cat
sat
rat

How can we represent a graph?

```
data Graph:
  | vertex(name :: String, neighbors :: List<Graph>)
end
```



```
data Graph:
  | vertex(name :: String, neighbors :: List<Graph>)
end
```



```
vertex("🦆",
  [list:
    vertex("🐱",
      [list:
        vertex("🦆",
          ...
```

This breaks horribly if there are cycles in the graph.

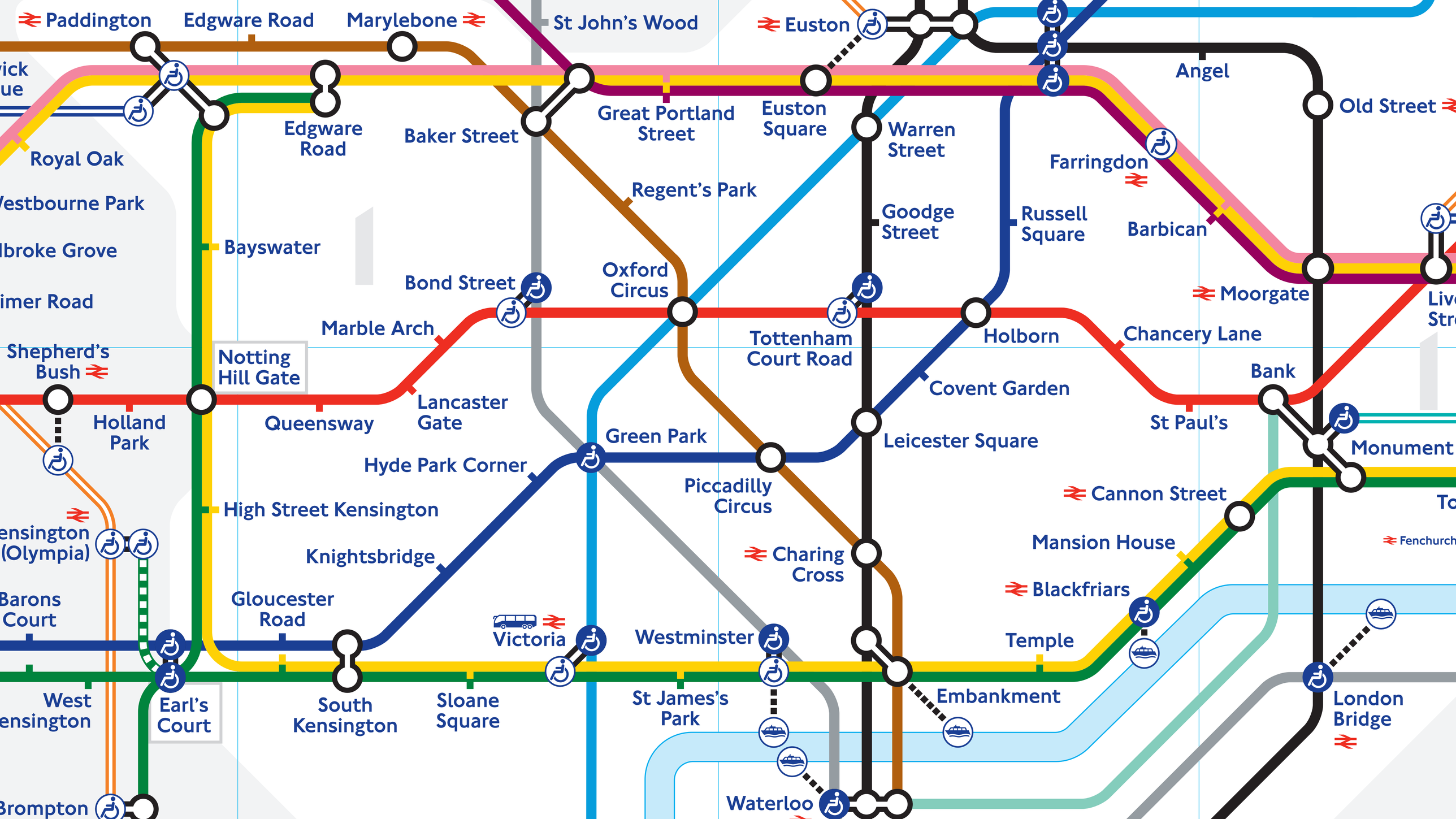
Better plan:

```
data Vertex:
  | vertex(name :: String, neighbors :: List<String>)
end
```

Now a graph is a `List<Vertex>`.

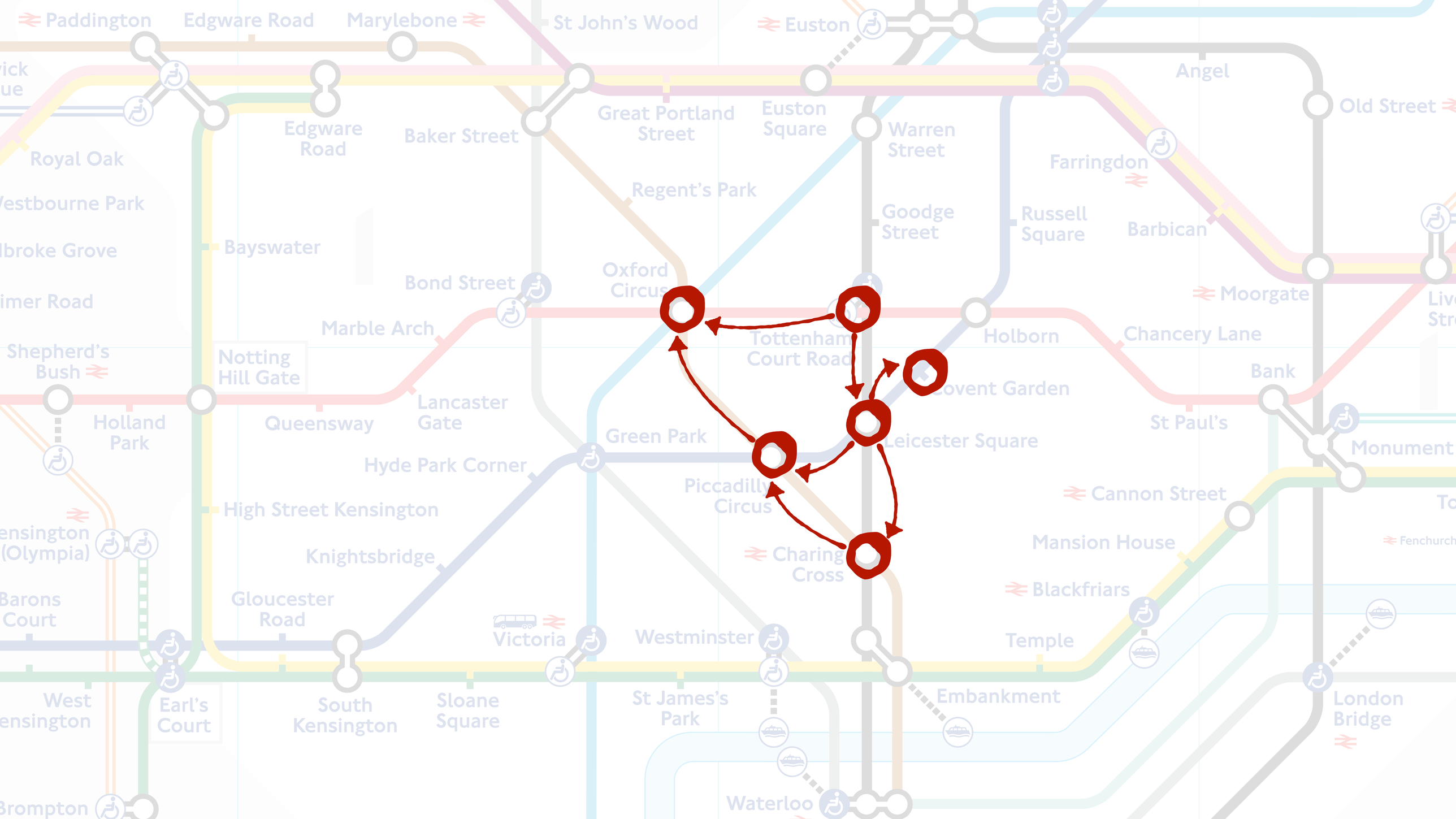
This is called an “adjacency list” representation.

Example: London Underground


Paddington
Edgware Road
Marylebone
St John's Wood
Euston
Angel
Old Street
Royal Oak
Edgware Road
Baker Street
Great Portland Street
Euston Square
Warren Street
Farringdon
Regent's Park
Goodge Street
Russell Square
Barbican
Bayswater
Bond Street
Oxford Circus
Moorgate
Marble Arch
Tottenham Court Road
Holborn
Chancery Lane
Shepherd's Bush
Notting Hill Gate
Bank
Holland Park
Queensway
Lancaster Gate
Covent Garden
St Paul's
Green Park
Leicester Square
Monument
Hyde Park Corner
Piccadilly Circus
Cannon Street
High Street Kensington
Mansion House
Knightsbridge
Charing Cross
Blackfriars
Barons Court
Gloucester Road
Victoria
Westminster
Temple
West Kensington
Earl's Court
South Kensington
Sloane Square
St James's Park
Embankment
London Bridge
Waterloo


Paddington
Edgware Road
Marylebone
St John's Wood
Euston
Angel
Old Street
Royal Oak
Edgware Road
Baker Street
Great Portland Street
Euston Square
Warren Street
Farringdon
Regent's Park
Goodge Street
Russell Square
Barbican
Bayswater
Bond Street
Oxford Circus
Moorgate
Marble Arch
Tottenham Court Road
Holborn
Chancery Lane
Shepherd's Bush
Notting Hill Gate
Bank
Covent Garden
Holland Park
Queensway
Lancaster Gate
Green Park
Leicester Square
St Paul's
Monument
Hyde Park Corner
Piccadilly Circus
Cannon Street
High Street Kensington
Mansion House
Knightsbridge
Charing Cross
Blackfriars
Gloucester Road
Victoria
Westminster
Temple
Barons Court
West Kensington
Earl's Court
South Kensington
Sloane Square
St James's Park
Embankment
London Bridge
Waterloo
Brompton


Paddington
Edgware Road
Marylebone
St John's Wood
Euston
Angel
Old Street
Great Portland Street
Euston Square
Warren Street
Edgware Road
Baker Street
Royal Oak
Farringdon
Regent's Park
Goodge Street
Russell Square
Barbican
Bayswater
Oxford Circus
Bond Street
Moorgate
Marble Arch
Tottenham Court Road
Holborn
Chancery Lane
Shepherd's Bush
Notting Hill Gate
Bank
Lancaster Gate
Holland Park
Queensway
Green Park
Leicester Square
St Paul's
Hyde Park Corner
Piccadilly Circus
Monument
High Street Kensington
Cannon Street
Knightsbridge
Mansion House
Charing Cross
Gloucester Road
Blackfriars
Victoria
Westminster
Temple
Earl's Court
South Kensington
Sloane Square
St James's Park
Embankment
London Bridge
Waterloo


Paddington
Edgware Road
Marylebone
St John's Wood
Euston
Angel
Old Street
Great Portland Street
Euston Square
Warren Street
Edgware Road
Baker Street
Royal Oak
Farringdon
Regent's Park
Goodge Street
Russell Square
Barbican
Bayswater
Oxford Circus
Bond Street
Moorgate
Marble Arch
Tottenham Court Road
Holborn
Chancery Lane
Shepherd's Bush
Notting Hill Gate
Bank
Lancaster Gate
Queensway
Holland Park
Green Park
Leicester Square
St Paul's
Hyde Park Corner
Monument
Piccadilly Circus
Cannon Street
High Street Kensington
Mansion House
Knightsbridge
Charing Cross
Blackfriars
Gloucester Road
Victoria
Westminster
Temple
Earl's Court
South Kensington
Sloane Square
St James's Park
Embankment
London Bridge
Waterloo

This isn't a complete model. We could keep adding directional edges – and, in fact, all of the stations that are connected would eventually have edges going both ways.

Exercise

Let's use a reactor to simulate a traveller riding the subway, following the connections in the graph.

Exercise

Let's use a reactor to simulate a traveller riding the subway, following the connections in the graph.

Starter code:

`tinyurl.com/101-2024-02-29-starter`

Code:

tinyurl.com/101-2024-02-29

Challenge exercise

Update the simulation to allow any number of travellers at the same time.

Code:

tinyurl.com/101-2024-02-29-rush-hour

Acknowledgments

This lecture incorporates material from:

Keith Schwarz, Stanford University

Laney Strange, Northeastern University