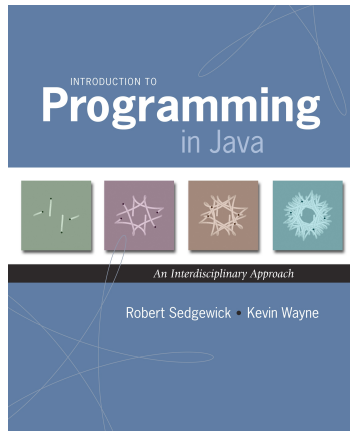


4.5 Small World Phenomenon



Introduction to Programming in Java: An Interdisciplinary Approach · Robert Sedgewick and Kevin Wayne · Copyright © 2002–2010 · 04/05/12 10:08:45 PM

Small World Phenomenon

Small world phenomenon. Six handshakes away from anyone.

An experiment to quantify effect. [Stanley Milgram, 1960s]

- You are given personal info of another person. e.g., occupation and age
- Goal: deliver message.
- Restriction: can only forward to someone you know by first name.
- Outcome: message delivered with average of 5 intermediaries.



Stanley Milgram



Kevin Bacon

2

Applications of Small World Phenomenon

Sociology applications.

- Looking for a job.
- Marketing products or ideas.
- Formation and spread of fame and fads.
- Train of thought followed in a conversation.
- Defining representative-ness of political bodies.
- **Kevin Bacon** game (movies, rock groups, facebook, etc.).

Other applications.

- Electronic circuits.
- Synchronization of neurons.
- Analysis of World Wide Web.
- Design of electrical power grids.
- Modeling of protein interaction networks.
- Phase transitions in coupled Kuramoto oscillators.
- Spread of infectious diseases and computer viruses.
- Evolution of cooperation in multi-player iterated Prisoner's Dilemma.

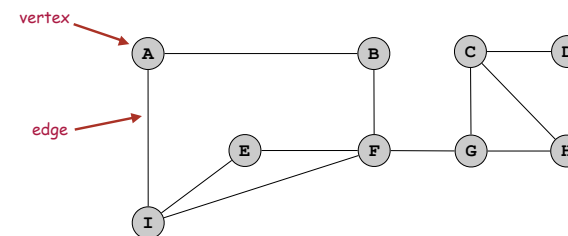
Reference. Duncan J. Watts, Small Worlds: The Dynamics of Networks between Order and Randomness, Princeton University Press, 1999.

5

Graph Data Type

Application demands a new data type.

- **Graph** = data type that represents pairwise connections.
- **Vertex** = element.
- **Edge** = connection between two vertices.



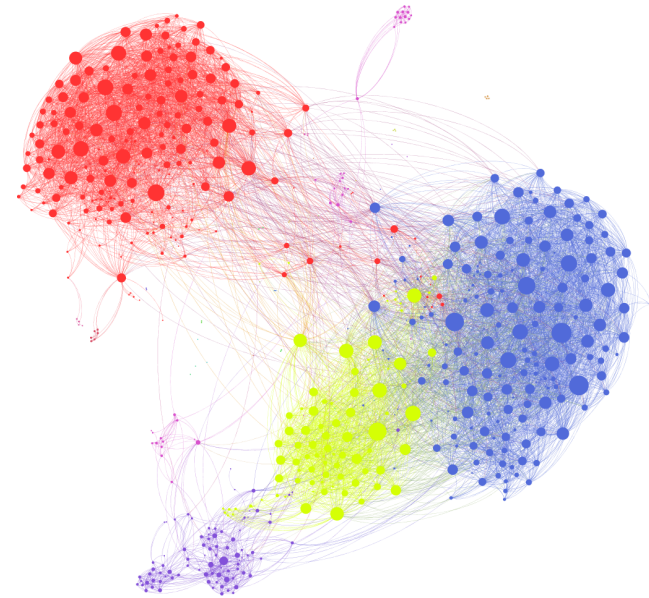
6

Graph Applications

graph	vertices	edges
communication	telephones, computers	fiber optic cables
circuits	gates, registers, processors	wires
mechanical	joints	rods, beams, springs
hydraulic	reservoirs, pumping stations	pipelines
financial	stocks, currency	transactions
transportation	street intersections, airports	highways, airway routes
scheduling	tasks	precedence constraints
software systems	functions	function calls
internet	web pages	hyperlinks
games	board positions	legal moves
social relationship	people, actors	friendships, movie casts
neural networks	neurons	synapses
protein networks	proteins	protein-protein interactions
chemical compounds	molecules	bonds

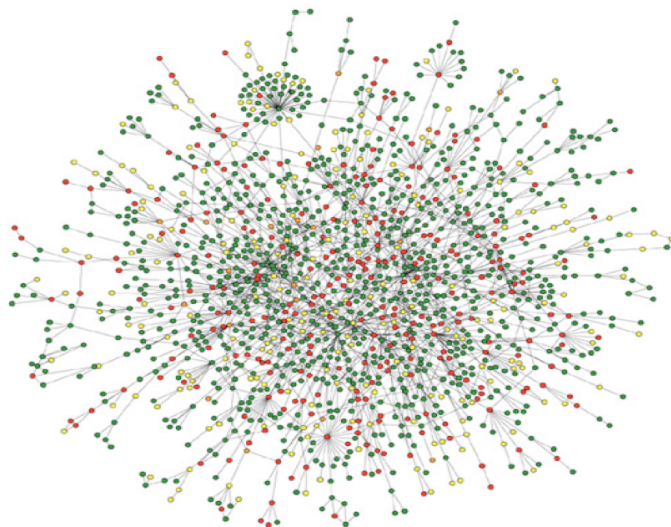
7

Facebook Friends



9

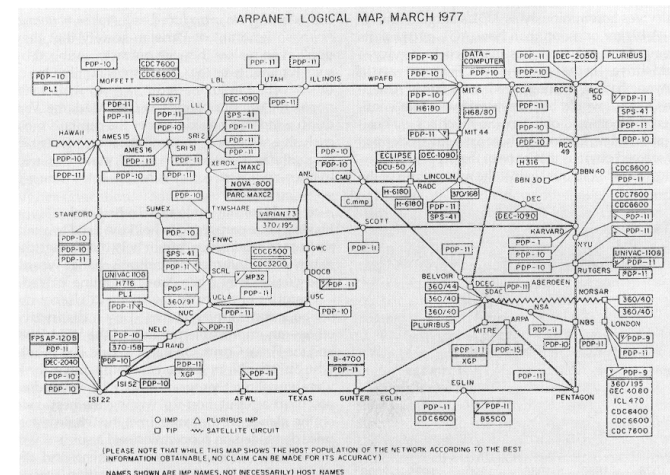
Protein Interaction Network



Reference: Jeong et al, Nature Review | Genetics

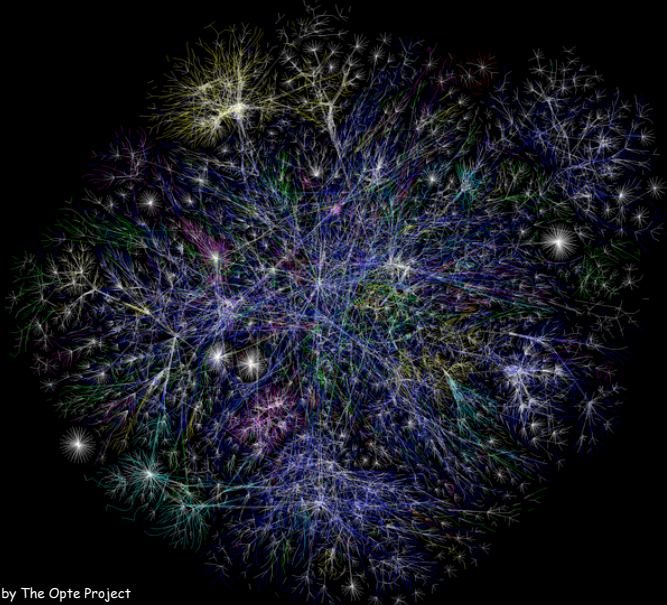
10

ARPANET



11

The Internet



The Internet as mapped by The Opte Project
<http://www.opte.org>

Internet Movie Database

Input format. Movie followed by list of performers, separated by slashes.

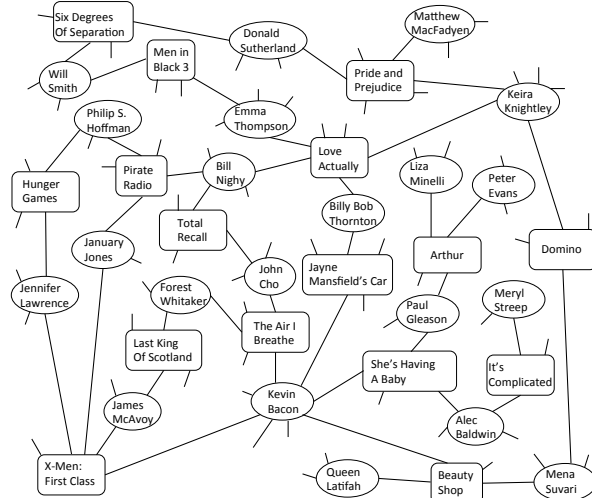
```
% more movies.txt
...
Tin Men (1987)/DeBoy, David/Blumenfeld, Alan/... /Geppi, Cindy/Hershey, Barbara
Tirez sur le pianiste (1960)/Heymann, Claude/.../Berger, Nicole (I)
Titanic (1997)Paxton, Bill/DiCaprio, Leonardo/.../Winslet, Kate
Titus (1999)/Weisskopf, Hermann/Rhys, Matthew/.../McEwan, Geraldine
To All a Good Night (1980)/George, Michael (II)/.../Gentile, Linda
To Be or Not to Be (1942)/Verebes, Ernő (I)/.../Lombard, Carole (I)
To Be or Not to Be (1983)/Brooks, Mel (I)/.../Bancroft, Anne
To Catch a Thief (1955)/Paris, Manuel/Grant, Cary/.../Kelly, Grace
To Die For (1989)/Bond, Steve (I)/Jones, Duane (I)/.../Maddalena, Julie
To Die For (1995)/Smith, Kurtwood/Kidman, Nicole/.../Tucci, Maria
To Die Standing (1990)/Sacha, Orlando/Anthony, Gerald/.../Rose, Jamie
To End All Wars (2001)/Kimura, Sakae/Ellis, Greg (II)/.../Sutherland, Kiefer
To Kill a Clown (1972)/Alda, Alan/Clavering, Eric/Lamberts, Heath/Danner, Blythe
To Live and Die in L.A. (1985)/McGroarty, Pat/Williams, Donnie/.../Dafoe, Willem
...
```

<http://www.imdb.com/interfaces>

Internet Movie Database

Q. How to represent the movie-performer relationships?
A. Use a graph.

- Vertex: performer or movie.
- Edge: connect performer to movie.

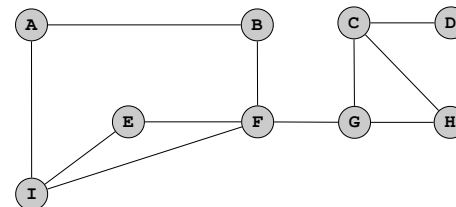


Graph API

Graph data type.

```
public class Graph (graph with String vertices)
    Graph() create an empty graph
    Graph(In in) read graph from input stream
    void addEdge(String v, String w) add edge v-w
    Iterable<String> adjacentTo(String v) neighbors of v
```

to support use with foreach



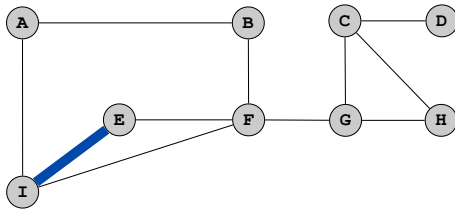
% more tiny.txt

```
A/B/I
B/A/F
C/D/G/H
D/C
E/F/I
F/B/E/G/I
G/C/F/H
H/C/G
I/A/E/F
```

Graph Representation

Graph representation: use a symbol table.

- Key = name of vertex.
- Value = set of neighbors.



String	SET<String>
key	value
A	B I
B	A F
C	D G H
D	C
E	F I
F	B E G I
G	C F H
H	C G
I	A E F

symbol table

16

Set Data Type

Set data type. Unordered collection of distinct keys.

```
public class SET<Key extends Comparable<Key>>
```

```
    SET() create a set
```

```
    boolean isEmpty() is the set empty?
```

```
    void add(Key key) add key to the set
```

```
    boolean contains(Key key) is key in the set?
```

Note: Implementations should also implement the Iterable<Key> interface to enable clients to access keys in sorted order with foreach loops

Q. How to implement?

A. Identical to symbol table, but ignore values.

17

Graph Implementation

```
public class Graph {
    private ST<String, SET<String>> st;

    public Graph() {
        st = new ST<String, SET<String>>();
    }

    public void addEdge(String v, String w) {
        if (!st.contains(v)) addVertex(v);
        if (!st.contains(w)) addVertex(w);
        st.get(v).add(w); ← add w to v's set of neighbors
        st.get(w).add(v); ← add v to w's set of neighbors
    }

    private void addVertex(String v) {
        st.put(v, new SET<String>()); ← add new vertex v
        with no neighbors
    }

    public Iterable<String> adjacentTo(String v) {
        return st.get(v);
    }
}
```

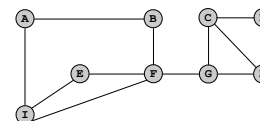
18

Graph Implementation (continued)

Second constructor. To read graph from input stream.

```
public Graph(In in) {
    st = new ST<String, SET<String>>();
    while (!in.isEmpty()) {
        String line = in.readLine();
        String[] names = line.split("/");
        for (int i = 1; i < names.length; i++)
            addEdge(names[0], names[i]);
    }
}
```

```
In in = new In("tiny.txt");
Graph G = new Graph(G, in);
```



```
% more tiny.txt
```

```
A/B/I
B/A/F
C/D/G/H
D/C
E/F/I
F/B/E/G/I
G/C/F/H
H/C/G
I/A/E/F
```

19

Graph Client: Movie Finder

Performer and movie queries.

- Given a performer, find all movies in which they appeared.
- Given a movie, find all performers.

```
public class MovieFinder {
    public static void main(String[] args) {

        In in = new In(args[0]); ← read in graph from a file
        Graph G = new Graph(in);

        while (!StdIn.isEmpty()) { ← process queries
            String v = StdIn.readLine();
            for (String w : G.adjacentTo(v))
                StdOut.println(w);
        }
    }
}
```

20

Graph Client: Movie Finder

```
% java MovieFinder action.txt
Bacon, Kevin
Death Sentence (2007)
River Wild, The (1994)
Tremors (1990)

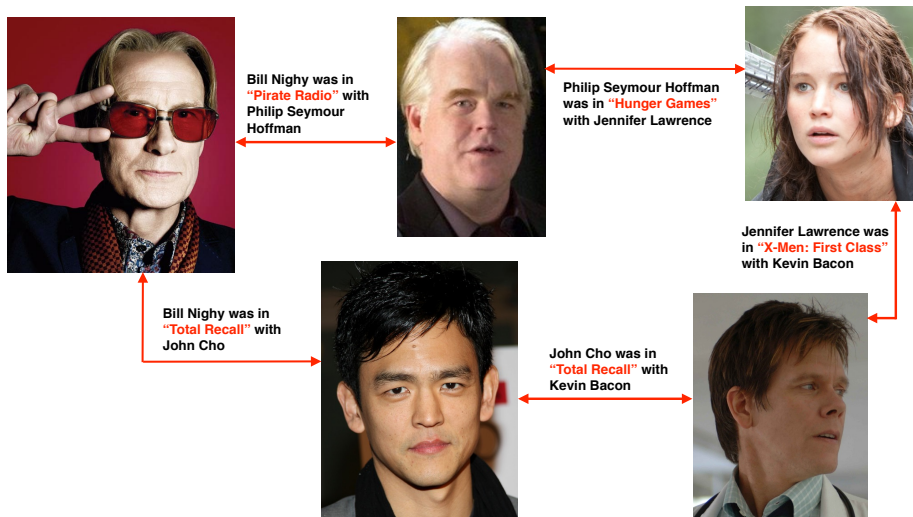
Roberts, Julia
Blood Red (1989)
I Love Trouble (1994)
Mexican, The (2001)
Ocean's Eleven (2001)

Eisgruber, Christopher
```

```
% java MovieFinder mpaa.txt
Bacon, Kevin
Air I Breathe, The (2007)
Air Up There, The (1994)
Animal House (1978)
Apollo 13 (1995)
Balto (1995)
Beauty Shop (2005)
Big Picture, The (1989)
...
Sleepers (1996)
Starting Over (1979)
Stir of Echoes (1999)
Telling Lies in America (1997)
Trapped (2002)
Tremors (1990)
We Married Margo (2000)
Where the Truth Lies (2005)
White Water Summer (1987)
Wild Things (1998)
Woodman, The (2004)
X-Men: First Class (2011)
```

21

Kevin Bacon Numbers



22

Oracle of Kevin Bacon

The Oracle of Bacon

Kevin Bacon to Buzz Mauro

Find link More options >>

Help Credits How it Works Contact Us Other games >

© 1999-2008 by Patrick Reynolds. All rights reserved.

Buzz Mauro

Sweet Dreams (2005)

Tatiana Ramirez

Interior de un silencio, El (2005)

Andres Suarez

Carlita's Secret (2004)

Paula Lemes (I)

Frost/Nixon (2008)

Kevin Bacon

23

Kevin Bacon Game

Game. Given an actor or actress, find **shortest** chain of movies connecting them to Kevin Bacon.

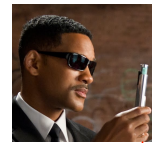
Actor	Was in	With
Matthew MacFadyen	Pride and Prejudice	Keira Knightley
Keira Knightley	Love Actually	Bill Nighy
Bill Nighy	Pirate Radio	Philip S. Hoffman
Philip S. Hoffman	Hunger Games	Jennifer Lawrence
Jennifer Lawrence	X-Men: First Class	James McAvoy
James McAvoy	Last King of Scotland	Forest Whitaker
Forest Whitaker	The Air I Breathe	Kevin Bacon
Kevin Bacon		



Peter Evans was in "Arthur" with Paul Gleason



Paul Gleason was in "She's Having a Baby" with Kevin Bacon



Will Smith was in "Men in Black 3" with Emma Thompson



Emma Thompson was in "Love Actually" with Billy Bob Thornton



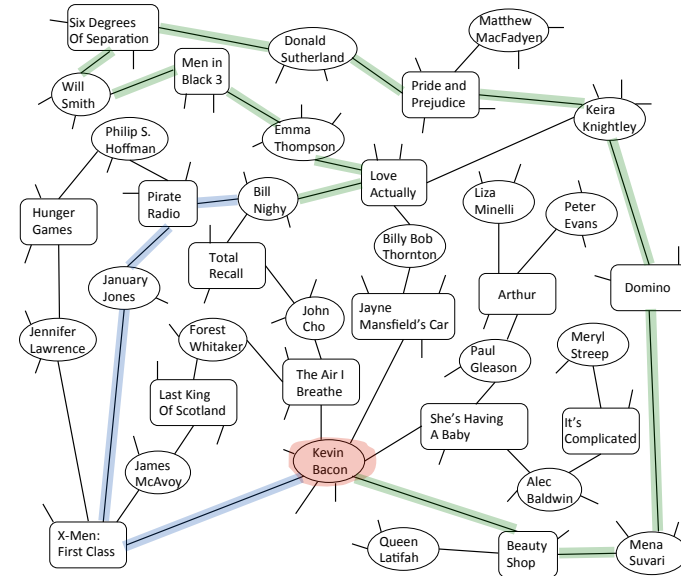
Billy Bob Thornton was in "Jayne Mansfield's Car" with Kevin Bacon



24

Computing Bacon Numbers

How to compute. Find shortest path in performer-movie graph.



26

PathFinder API

PathFinder API.

```
public class PathFinder
    PathFinder(Graph G, String s)    constructor
    int distanceTo(String v)         length of shortest path
                                    from s to v in G
    Iterable<String> pathTo(String v) shortest path
                                    from s to v in G
```

Design principles.

- Decouple graph algorithm from graph data type.
- Avoid feature creep: don't encrust Graph with search features; instead make a new datatype.

27

Computing Bacon Numbers: Java Implementation

```
public class Bacon {
    public static void main(String[] args) {
        In in = new In(args[0]); ← read in the graph from a file
        Graph G = new Graph(in);
        String s = "Bacon, Kevin";
        PathFinder finder = new PathFinder(G, s); ← create object to
        return shortest paths
        while (!StdIn.isEmpty()) { ← process queries
            String actor = StdIn.readLine();
            for (String v : finder.pathTo(actor))
                StdOut.println(v);
        }
    }
}
```

```
% java Bacon top-grossing.txt
Stallone, Sylvester
Rocky III (1982)
Tamburro, Charles A.
Terminator 2: Judgment Day (1991)
Berkeley, Xander
Apollo 13 (1995)
Bacon, Kevin
```

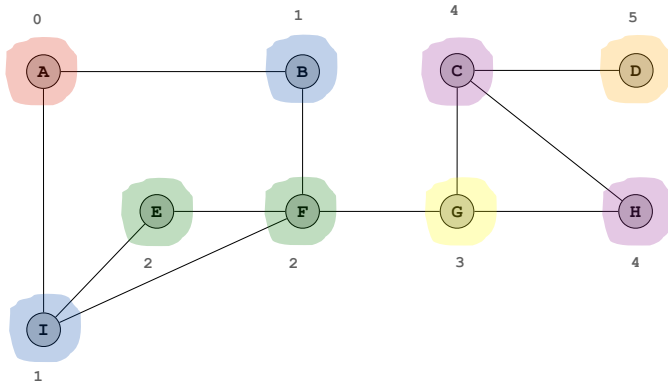
```
% java Bacon top-grossing.txt
Goldberg, Whoopi
Sister Act (1992)
Grodénchik, Max
Apollo 13 (1995)
Bacon, Kevin
Eisgruber, Christopher
```

28

Computing Shortest Paths

To compute shortest paths:

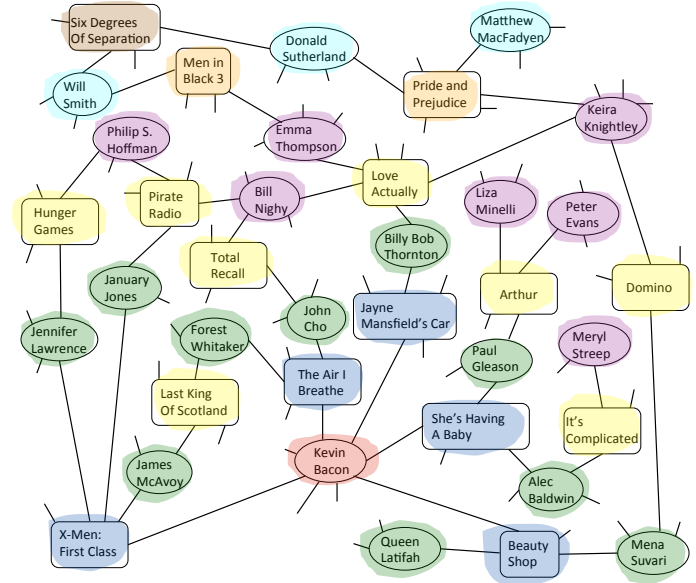
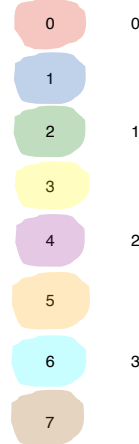
- Source vertex is at distance 0.
- Its neighbors are at distance 1.
- Their remaining neighbors are at distance 2.
- Their remaining neighbors are at distance 3.
- ...



29

Computing Shortest Paths

distance Bacon number



Breadth First Search

Goal. Given a vertex s , find shortest path to every other vertex v .

BFS from source vertex s

Put s onto a FIFO queue.

Repeat until the queue is empty:

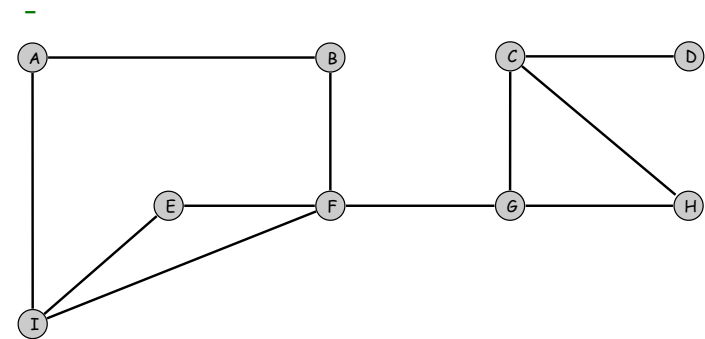
- dequeue the least recently added vertex v
- add each of v 's unvisited neighbors to the queue, and mark them as visited.



Key observation. Vertices are visited in increasing order of distance from s because we use a FIFO queue.

31

Breadth First Search



front

FIFO Queue

32

Breadth First Searcher: Preprocessing

```

public class Pathfinder {
    private ST<String, String> prev = new ST<String, String>();
    private ST<String, Integer> dist = new ST<String, Integer>();

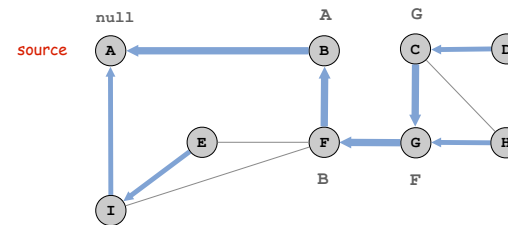
    public Pathfinder(Graph G, String s) {
        Queue<String> q = new Queue<String>();
        q.enqueue(s);
        dist.put(s, 0);
        while (!q.isEmpty()) {
            String v = q.dequeue();
            for (String w : G.adjacentTo(v)) {
                if (!dist.contains(w)) {
                    q.enqueue(w);
                    dist.put(w, 1 + dist.get(v));
                    prev.put(w, v);
                }
            }
        }
        // other Pathfinder methods go here
    }
}
    
```

82

Breadth First Searcher: Finding the Path

To find **shortest path**: follow `prev[]` from vertex `v` back to source `s`.

- Consider vertices: `v`, `prev[v]`, `prev[prev[v]]`, ..., `s`.
- Ex: shortest path from `c` to `A`: `C - G - F - B - A`
- Use stack to reverse order



key	prev	dist
A	-	0
B	A	1
C	G	4
D	C	5
E	I	2
F	B	2
G	F	3
H	G	4
I	A	1

symbol tables

```

public Iterable<String> pathTo(String v) {
    Stack<String> path = new Stack<String>();
    while (dist.contains(v)) {
        path.push(v);
        v = prev.get(v);
    }
    return path;
}
    
```

83

Running Time Analysis

Analysis. BFS *scales* to solve huge problems.

data File	movies	performers	edges	read input	build graph	BFS	pathTo
G.txt	1,288	21,177	28K	0.26 sec	0.52 sec	0.32 sec	0 sec
PG13.txt	2,538	70,325	100K	0.31 sec	0.99 sec	0.72 sec	0 sec
action.txt	14,938	139,861	270K	0.72 sec	2.8 sec	2.0 sec	0 sec
mpaa.txt	21,861	280,624	610K	2.1 sec	7.5 sec	5.5 sec	0 sec
all.txt	285,462	933,864	3.3M	15 sec	56 sec	39 sec	0 sec

60MB

data as of April 9, 2007

84

Data Analysis

Exercise. Compute histogram of Kevin Bacon numbers.

Input. ~2.6 million movies, ~5.7 million actors.

Bacon #	Frequency
0	1
1	2,769
2	305,215
3	1,021,901
4	253,177
5	20,060
6	2,033
7	297
8	25
9	7
∞	>32,000

These are very hard to find!

Fred Ott, solo actor in
Fred Ott Holding a Bird (1894)

data as of April 28, 2013

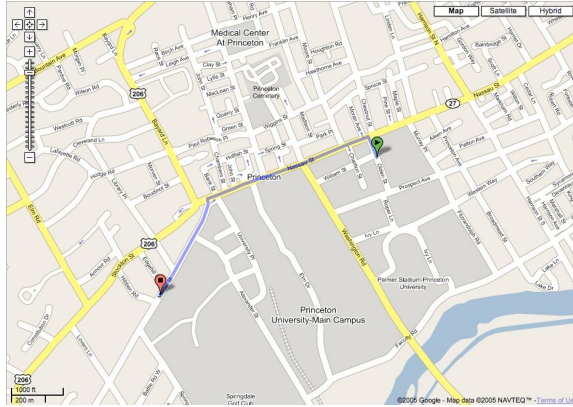
85

Applications of Breadth First Search

More BFS applications.

- Particle tracking.
- Image processing.
- Crawling the Web.
- Routing Internet packets.
- ...

Extensions. Google maps.



86

Erdős Numbers

87

Erdős Numbers

Paul Erdős. *Legendary, brilliant, prolific mathematician who wrote over 1500 papers!*

What's your Erdős number?

- Co-authors of a paper with Erdős: 1.
- Co-authors of those co-authors: 2.
- And so on ...

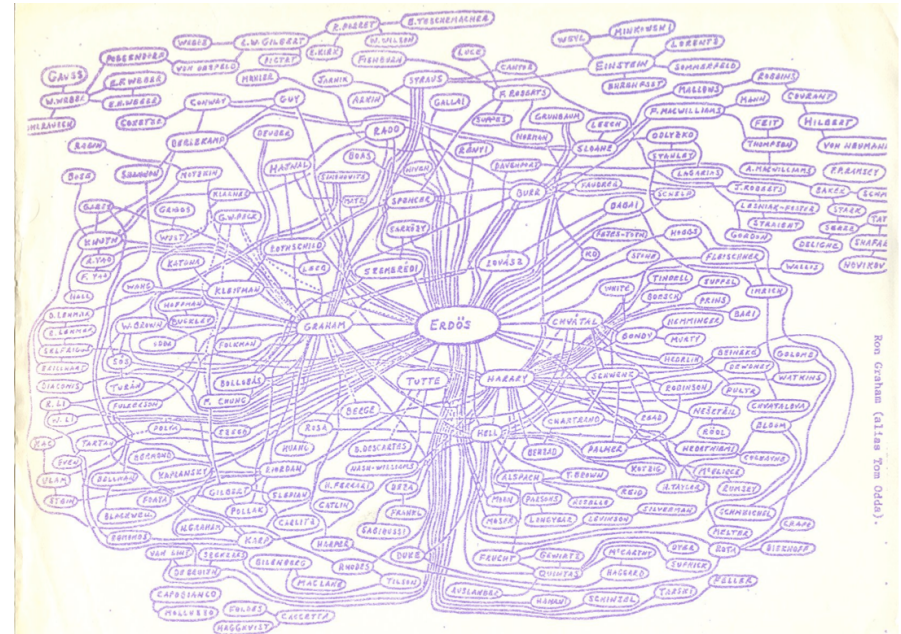


Paul Erdős (1913-1996)

Erdős #	Frequency
0	1
1	502
2	5,713
3	26,422
4	62,136
5	66,157
6	32,280
7	10,431
8	3,214
9	953
10	262
11	94
12	23
13	4
14	7
15	1
∞	4 billion +

88

Erdős Graph



from Graham (after Tom Odell).

89

Conclusions

Linked list. Ordering of elements.

Binary tree. Hierarchical structure of elements.

Graph. Pairwise connections between elements.

Data structures.

- Queue: linked list.
- Set: binary tree.
- Symbol table: binary tree.
- Graph: symbol table of sets.
- Breadth first searcher: graph + queue + symbol table.

Importance of data structures.

- Enables us to build and debug large programs.
- Enables us to solve large problems efficiently.