1.6 Case Study: Random Surfer

Theoretical hypertext computer system; pioneering concept for world wide web.
- Follow links from book or film to another.
- Tool for establishing links.

World Wide Web


Web Browser

Web browser. Killer application of the 1990s.
When it was proclaimed that the Library contained all books, the first impression was one of extravagant happiness... There was no personal or world problem whose eloquent solution did not exist in some hexagon.

This inordinate hope was followed by an excessive depression. The certitude that some shelf in some hexagon held precious books and that these precious books were inaccessible seemed almost intolerable.
Google’s PageRank™ algorithm. [Sergey Brin and Larry Page, 1998]

- Measure popularity of pages based on hyperlink structure of Web. Revolutionized access to world’s information.

**Web Graph Input Format**

**Input format.**
- N pages numbered 0 through N-1.
- Represent each hyperlink with a pair of integers.

**Transition Matrix**

**Transition matrix.** $p[i][j] = \text{prob. that surfer moves from page } i \text{ to } j.$

**Caveat.** Crude, but useful, web surfing model.
- No one chooses links with equal probability.
- No real potential to surf directly to each page on the web.
- The 90-10 breakdown is just a guess.
- It does not take the back button or bookmarks into account.
- We can only afford to work with a small sample of the web.
- ...

**90-10 Rule**

**Model.** Web surfer chooses next page:
- 90% of the time surfer clicks random hyperlink.
- 10% of the time surfer types a random page.

**Transition matrix.** $p[i][j] = \text{prob. that surfer moves from page } i \text{ to } j.$

**surfer on page 1 goes to page 2 next 38% of the time**
public class Transition {
    public static void main(String[] args) {
        int N = StdIn.readInt();  // number of pages
        int[][] counts = new int[N][N];  // # links from page i to j
        int[] outDegree = new int[N];  // # links from page

        // accumulate link counts
        while (!StdIn.isEmpty()) {
            int i = StdIn.readInt();
            int j = StdIn.readInt();
            outDegree[i]++;
            counts[i][j]++;
        }

        // print transition matrix
        StdOut.println(N + " \times " + N);
        for (int i = 0; i < N; i++) {
            for (int j = 0; j < N; j++) {
                double p = Math.round(counts[i][j]/outDegree[i] * 100) / 100;
                StdOut.printf("%7.5f ", p);
            }
            StdOut.println();
        }
    }
}

Monte Carlo Simulation

Surfer starts on page 0.
Repeatedly choose next page, according to transition matrix.
Calculate how often surfer visits each page.
Random Surfer

Random move. Surfer is on page page. How to choose next page j?

- Row page of transition matrix gives probabilities.
- Compute cumulative probabilities for row page.
- Generate random number r between 0.0 and 1.0.
- Choose page j corresponding to interval where r lies.

```java
public class RandomSurfer {
    public static void main(String[] args) {
        // read in transition matrix ...
        double[][] p = new double[N][N];
        // transition matrix

        // simulate random surfer and count page frequencies
        int[] freq = new int[N];
        for (int t = 0; t < T; t++) {
            // make one random move
            freq[page]++;
        }

        // print page ranks
        for (int i = 0; i < N; i++) {
            StdOut.printf("%8.5f", (double) freq[i] / T);
        }
    }
}
```

Mathematical Context

Convergence. For the random surfer model, the fraction of time the surfer spends on each page converges to a unique distribution, independent of the starting page.
Mixing a Markov Chain

The Power Method

Q. If the surfer starts on page 0, what is the probability that surfer ends up on page \(i\) after one step?

A. First row of transition matrix.

\[
\begin{align*}
\text{rank} & \quad \mathbf{p} & \quad \text{newRank} \\
\text{first move} & \quad \begin{bmatrix}
0.02 & 0.92 & 0.02 & 0.02 \\
0.02 & 0.38 & 0.38 & 0.20
\end{bmatrix} \\
\text{second move} & \quad \begin{bmatrix}
0.02 & 0.92 & 0.02 & 0.02 \\
0.02 & 0.38 & 0.38 & 0.20 \\
0.02 & 0.02 & 0.92 & 0.02 \\
0.47 & 0.02 & 0.47 & 0.02
\end{bmatrix}
\end{align*}
\]

The Power Method

Q. If the surfer starts on page 0, what is the probability that surfer ends up on page \(i\) after two steps?

A. Matrix-vector multiplication.

\[
\begin{align*}
\text{rank} & \quad \mathbf{p} & \quad \text{newRank} \\
\text{first move} & \quad \begin{bmatrix}
0.02 & 0.92 & 0.02 & 0.02 \\
0.02 & 0.38 & 0.38 & 0.20
\end{bmatrix} \\
\text{second move} & \quad \begin{bmatrix}
0.02 & 0.92 & 0.02 & 0.02 \\
0.02 & 0.38 & 0.38 & 0.20 \\
0.02 & 0.02 & 0.92 & 0.02 \\
0.47 & 0.02 & 0.47 & 0.02
\end{bmatrix}
\end{align*}
\]

The Power Method

Power method. Repeat until page ranks converge.
Mathematical Context

Convergence. For the random surfer model, the power method iterates converge to a unique distribution, independent of the starting page.

Random Surfer: Scientific Challenges

Google’s PageRank™ algorithm. [Sergey Brin and Larry Page, 1998]

- Rank importance of pages based on hyperlink structure of web, using 90-10 rule.
- Revolutionized access to world’s information.

Scientific challenges. Cope with 4 billion-by-4 billion matrix!

- Need data structures to enable computation.
- Need linear algebra to fully understand computation.