Small World Phenomenon

[Problem Definition 2](#_Toc435829300)

[Descriptions and Suggestions 2](#_Toc435829301)

[Main Goals 4](#_Toc435829302)

[Optimization Goal 4](#_Toc435829303)

[Project Requirements 4](#_Toc435829304)

[Required Implementation 4](#_Toc435829305)

[Input 5](#_Toc435829306)

[Output 5](#_Toc435829307)

[Test Cases 5](#_Toc435829308)

[Deliverables 6](#_Toc435829309)

[Implementation (60%) 6](#_Toc435829310)

[Document (40%) 6](#_Toc435829311)

[Allowed Codes 6](#_Toc435829312)

[Milestones 6](#_Toc435829313)

[BONUSES 7](#_Toc435829314)

# Problem Definition

1. Idea, usages and requirements
2. Related terminologies/description

The *small-world phenomenon* is the principle that we are all linked by short chains of acquaintances. The pioneering work of [Stanley Milgram](http://www.stanleymilgram.com/) in the 1960's was among the first to make the phenomenon quantitative, allowing people to speak of the *six degrees of separation* between any two people in the United States. Since then, a number of network models have been proposed as frameworks in which to study the problem analytically. One of the most refined of these models was formulated in recent work of [Watts and Strogatz](http://pup.princeton.edu/titles/6768.html); their framework provided compelling evidence that the small-world phenomenon is pervasive in a range of networks arising in nature and technology, and a fundamental ingredient in the evolution of the World Wide Web.

Two main terminologies are of interest to us in this project:

#### Degree of Separation

Between any two persons, is defined as the shortest chain of links that leads one person to the other.

#### Relation Strength

Between any two persons, is defined as the total number of common "factors" between each two persons in the shortest chain of links. If there're many shortest chains of links with the same length, the relation strength is then the one with max common "factors".

# Descriptions and Suggestions

In this project, we will investigate this phenomenon on the movies' actors and actresses by measuring both the **degree of separation** and **relation strength** between any two actors/ actresses.

Two actors or actresses are linked if they appeared in a movie together. The degree of separation between two actors/actresses is the shortest chain of links that leads one of them to the other.

### Example1: Degree of Separation

Robert De Niro has a **one** degree of separation with Kevin Bacon because he appeared in *Sleepers* with Kevin Bacon. Elvis Presley has a **two** degree of separation with Kevin Bacon: Elvis was in *Change of Habit* with Edward Asner, and Asner appeared in *JFK* with Kevin Bacon.

1 degree of separation

2 degree of separation

Figure 1: Examples for the degree of separation

The relation strength here is defined as the number of common movies along the shortest chain of links between the two actors/actresses. If there are multiple chains of links with the same length, the relation strength is defined as the chain with max common movies along it, as shown in the following example.

### Example2: Relation Strength

Actor A and actor B are directly linked with a degree of separation equal 1. The relation strength between them is three movies since they are directly appeared together in the three movies.

Actor C and actor E are linked with a degree of separation equal 2 through either actor A, actor B or actor D. The relation strength should equal the max of the three options. Following are the three chains and the relation strength of each one:

|  |  |  |
| --- | --- | --- |
| **I** | **Chain** | **Relation Strength** |
| 1 | Actor C 🡺 Actor A 🡺 Actor E | C, A has relation strength = 2 (movie 1, 3)  A, E has relation strength = 1 (movie 2)  **Total = 3** |
| 2 | Actor C 🡺 Actor B 🡺 Actor E | C, B has relation strength = 2 (movie 1, 3)  B, E has relation strength = 1 (movie 2)  **Total = 3** |
| 3 | Actor C 🡺 Actor D 🡺 Actor E | C, D has relation strength = 1 (movie 3)  D, E has relation strength = 1 (movie 2)  Total = 2 |

Thus, the relation strength between actor C and actor E is 3 (either through actor A or actor B)

Relation Strength between A & B = 3 movies

Relation Strength between C & E = 3 movies (via A or B)

Figure 2: Examples for the relation strength

## Main Goals

Your task is to read in a file containing a list of movies and the actors/actresses that appeared in them. Then, compute the degree of separation between any two actors/actresses. You will read in a list of actor-pairs, then, for each pair, print out the following:

1. The degree of separation between the two actors/actresses.
2. The relation strength (total number of common movies) between them.
3. The shortest chain of movies that leads the first actor/actress to the second.

## Optimization Goal

You are asked tooptimize the calculation of the degree of separation so that it can process **thousands of queries** for a given set of movies. Once you read in (and optionally preprocess) the list of movies with their actors/actresses, your program should find the degree of separation in ***sublinear*** time.

One method would be to pre-compute the degree of separation for all pairs of actors/actresses; however you cannot afford the quadratic space required to store all of this information. Your goal is to reduce the amount of work involved per degree of separation computation, without using excessive space. There is one suggestion of potential idea below which you may implement. Or you can develop and implement your own idea(s).

### Suggested Idea:

Starting from the first actor/actress in the query pair, stop the search as soon as you discover the shortest chain of links to the other one. With this approach, you can make the running time per query proportional to the number of actors and movies examined during this query. However, this requires some care when re-initializing your data for the next query. Since you are doing repeated queries, you can speed things up dramatically by only re-initializing those values that changed in the previous query.

# Project Requirements

## Required Implementation

1. **Read** in a file containing a list of movies and their actors/actresses
2. **Read** in a file with a list of actor-pairs,
3. For each pair, **print out** the following:
4. The degree of separation between the two actors/actresses.
5. The relation strength (total number of common movies) between them.
6. The shortest chain of movies that leads the first actor to the second.
7. **Optimize** the calculation of the degree of separation for multiple queries, so that each query runs in ***sublinear*** time.

## Input

1. **Movies file:** Each line consists of
   * a movie title, followed by
   * a list of actors and actresses that appeared in that movie, delimited by the character '/'.

Here is an abbreviated example:

Picture Perfect (1997)/Aniston, Jennifer/Bacon, Kevin/Dukakis, Olympia/Mohr, Jay

Planes, Trains & Automobiles (1987)/Bacon, Kevin/Candy, John/Martin, Steve/Robins, Laila

Beach, The (2000)/DiCaprio, Leonardo/York, Daniel/Patarakijjanon, Patcharawan

1. **Queries file:** Each line consists of the names of two actors/actresses separated by '/'

Here is an abbreviated example:

Aniston, Jennifer/Bacon, Kevin

DiCaprio, Leonardo/Martin, Steve

Bacon, Kevin/Candy, John

## Output

For each query pair, print out:

1. The degree of separation between the two actors/actresses.
2. The relation strength (total number of common movies) between them.
3. The shortest chain of movies that leads the first actor/actress to the second.

## Test Cases

* Sample Case:
* Few test cases with small values that can be traced.
* Complete Test:

1. Small: hundreds of movies (order of 1000 movies)
2. Medium: thousands of movies (order of 20,000 movies)
3. Large: hundred thousand of movies (order of 200,000 movies)

# Deliverables

## Implementation (60%)

1. **Read** and parse two files (Movies & Queries)
2. For each pair, **print out** the following:
3. The **degree of separation** between the two actors/actresses.
4. The **relation strength** (total number of common movies) between them.
5. The **shortest chain of movies** that leads the first actor to the second.
6. **Optimize** the calculation of the degree of separation to be run in ***sublinear*** time

## Document (40%)

1. Entire source code
2. Detailed analysis of your code
3. Execution time of "**Complete Test**" cases: before and after applying the optimization idea

## Allowed Codes

No external code is allowed.

# Milestones

|  |  |  |
| --- | --- | --- |
|  | **Deliverables** | **Due to** |
| **Milestone1** | 1. **Read** and parse two files (Movies & Queries) 2. Calculate **degree of separation** for query pair 3. Calculate **relation strength** for the query pair 4. Print **shortest chain** of movies 5. Apply **optimization** idea 6. Documentation | **(START of Week Before Practical Exam)** |
| **For Milestone1:**   * + **MUST** deliver the required tasks and **ENSURE** it’s worked correctly   + **MUST** deliver in your scheduled time (TO BE ANNOUNCED) | | |

# BONUSES

* 1. Calculate and print (or draw) the distribution of the degree of separation between a given actor/actress and all other actors/actresses.

#### Ex.:

* Query actor: Kevin Bacon
* Distribution of the degree of separation:

Deg. of Separ. Frequency

-------------------------

0 1

1 1494

2 127791

3 239671

4 36475

5 2965

6 275

7 39

8 47

9 99

10 15

11 2

>11 9703

* 1. Find and display the **strongest path** (based on the relation strength) between the query pair of actors/actresses.
  2. Find and display the **min number of movies** that link all actors/actresses together.