Introduction to Graphs

Building up visualizations for networks

Plan for the week

- HW 8: Huffman
    - Due Friday 3/12
    - No Resubs
- Office Hours end on **Friday, 6:30pm**
- Simulated Final due **Sunday, 11:59pm PST**
    - Key and tests will be released tomorrow!
- TA choice tomorrow
- Course Evaluations

Movies with Chris Pratt

Guardians of the Galaxy

Chris Pratt, Vin Diesel, Bradley Cooper, Zoe Saldana

Passengers

Jennifer Lawrence, Chris Pratt, Michael Sheen, Laurence Fishburne

The Magnificent Seven,

Denzel Washington, Chris Pratt, Ethan Hawke, Vincent D'Onofrio

Jurassic World

Chris Pratt, Bryce Dallas Howard, Ty Simpkins, Judy Greer

The Lego Movie

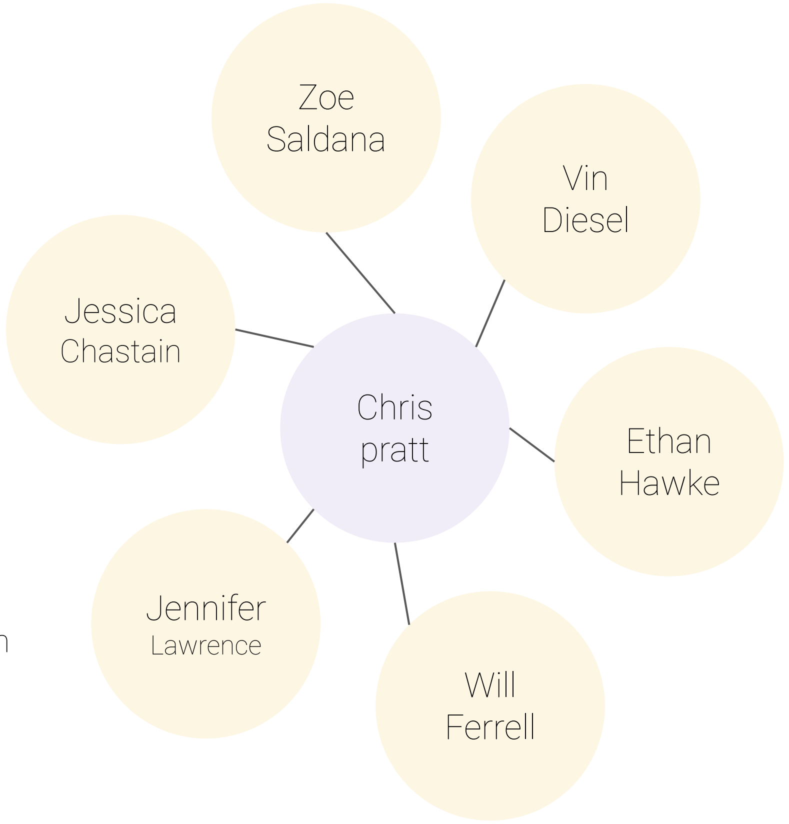
Chris Pratt, Will Ferrell, Elizabeth Banks, Will Arnett

Zero Dark Thirty

Jessica Chastain, Joel Edgerton, Chris Pratt, Mark Strong

10 Years

Channing Tatum, Rosario Dawson, Chris Pratt, Jenna Dewan Tatum



Graph Representations in Java

The `java.util` package doesn't have a `Graph` data type :(

- What data or relationship do I want to store?
  - Given the name of an actor, what is important to track?
- What Java data structure can I use?
  - Does this structure process my query "fast"?
  - Do I care about order?

Adjacency List

Given an actor, keep track of the "neighbors" of this actor

`Interface:` **`Map<String, Set<String>>`**

`Implementation:` **`HashMap`**

Adjacency Lists

Key	Values
Chris Pratt	Bryce Dallas Howard, Vincent D'Onofrio, Will Ferrell, Joel Edgerton, Bradley Cooper
Bryce Dallas Howard	Emma Stone, Anna Kendrick, Seth Rogen, Chris Pratt, Paul Giamatti, Robert Redford, Jeffrey Wright, Viola Davis
Will Ferrell	Mark Wahlberg, Eva Mendes, Will Arnett, Sacha Baron Cohen,Elizabeth Banks, Brad Pitt

Search for “Brad Pitt” starting from “Chris Pratt”

Adjacency Lists

Key	Values
Chris Pratt	**Bryce Dallas Howard**, Vincent D'Onofrio, Will Ferrell, Joel Edgerton, Bradley Cooper
Bryce Dallas Howard	Emma Stone, Anna Kendrick, Seth Rogen, Chris Pratt, Paul Giamatti, Robert Redford, Jeffrey Wright, Viola Davis
Will Ferrell	Mark Wahlberg, Eva Mendes, Will Arnett, Sacha Baron Cohen,Elizabeth Banks, Brad Pitt

Search for “Brad Pitt” starting from “Chris Pratt”

Adjacency Lists

Key	**Values**
Chris Pratt	Bryce Dallas Howard, **Vincent D'Onofrio,** Will Ferrell, Joel Edgerton, Bradley Cooper
Bryce Dallas Howard	Emma Stone, Anna Kendrick, Seth Rogen, Chris Pratt, Paul Giamatti, Robert Redford, Jeffrey Wright, Viola Davis
Will Ferrell	Mark Wahlberg, Eva Mendes, Will Arnett, Sacha Baron Cohen,Elizabeth Banks, Brad Pitt

Search for “Brad Pitt” starting from “Chris Pratt”

Adjacency Lists

Key	Values
Chris Pratt	Bryce Dallas Howard, Vincent D'Onofrio, **Will Ferrell**, Joel Edgerton, Bradley Cooper
Bryce Dallas Howard	Emma Stone, Anna Kendrick, Seth Rogen, Chris Pratt, Paul Giamatti, Robert Redford, Jeffrey Wright, Viola Davis, Will Ferrell
Will Ferrell	Mark Wahlberg, Eva Mendes, Will Arnett, Sacha Baron Cohen,Elizabeth Banks, **Brad Pitt**

Search for “Brad Pitt” starting from “Chris Pratt”

Breadth First Search

```
search(start, target)
    queue = { start }
    while queue not empty:
        vertex = remove_first(queue)
        for each neighbor of vertex:
            add neighbor to end of queue
            if neighbor == target:
                done!
```

Breadth First Search

```
search(start, target)
    queue = { start }
    while queue not empty:
        vertex = remove_first(queue)
        for each neighbor of vertex:
            add neighbor to end of queue
            if neighbor == target:
                done!
```

What can go wrong if I keep exploring vertices naively?

Is it okay if I visit the same vertex twice?

Breadth First Search

```
search(start, target)
    mark start as “visited”
    queue = { start }
    while queue not empty:
        vertex = remove_first(queue)
        for each neighbor of vertex:
            if neighbor is not visited:
                mark neighbor as “visited”
                add neighbor to end of queue
            if neighbor == target:
                done!
```

Breadth First Search

```
search(start, target)
    mark start as "visited"
    queue = { start }
    while queue not empty:
        vertex = remove_first(queue)
        for each neighbor of vertex:
            if neighbor is not visited:
                mark neighbor as "visited"
                add neighbor to end of queue
            if neighbor == target:
                done!
```

Keep track of all the visited vertices in a set!

- Mark vertex as “visited” = add vertex to visited set
- Check if a vertex is “visited” = does the set contain the vertex?