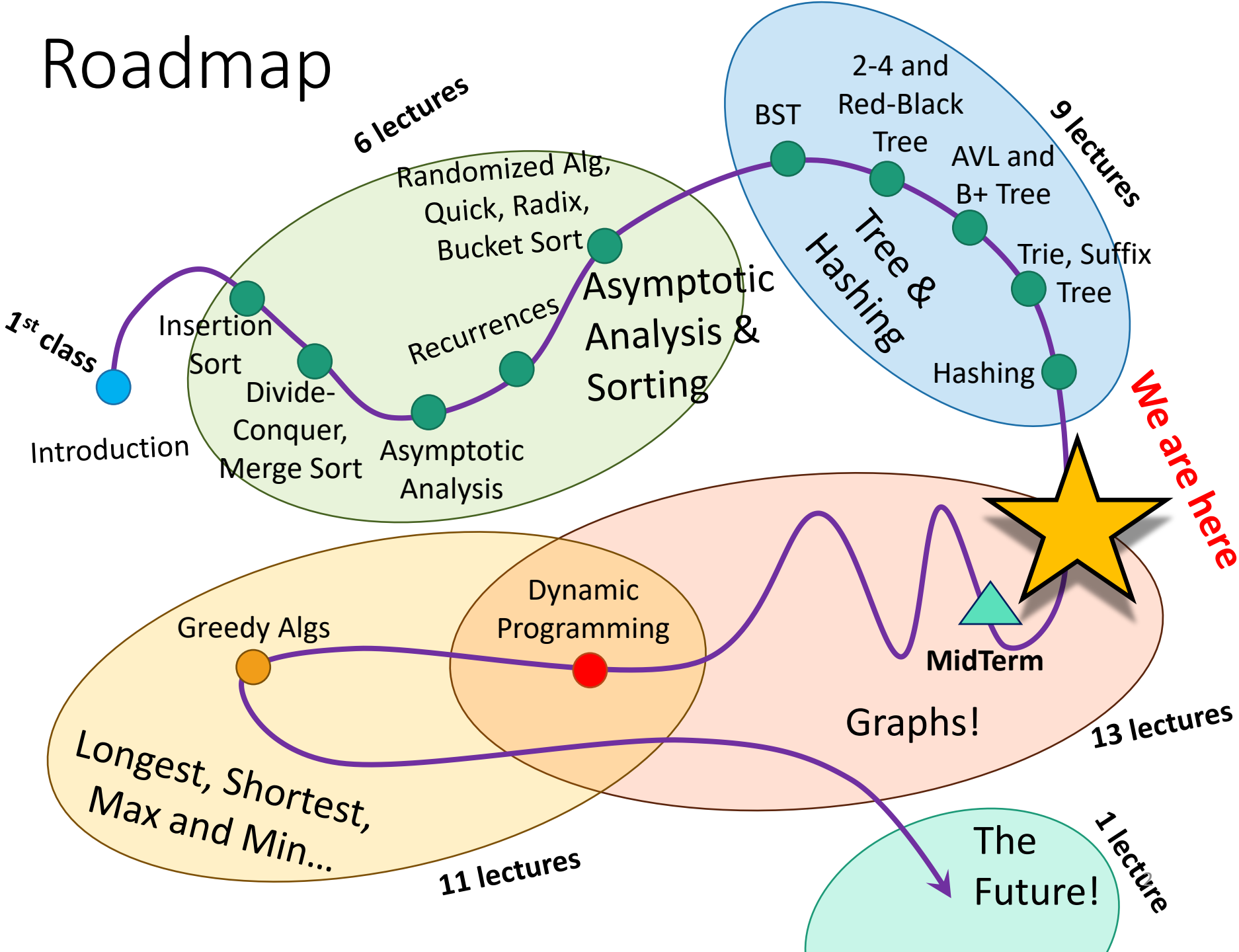
Advanced Data Structures and Algorithms

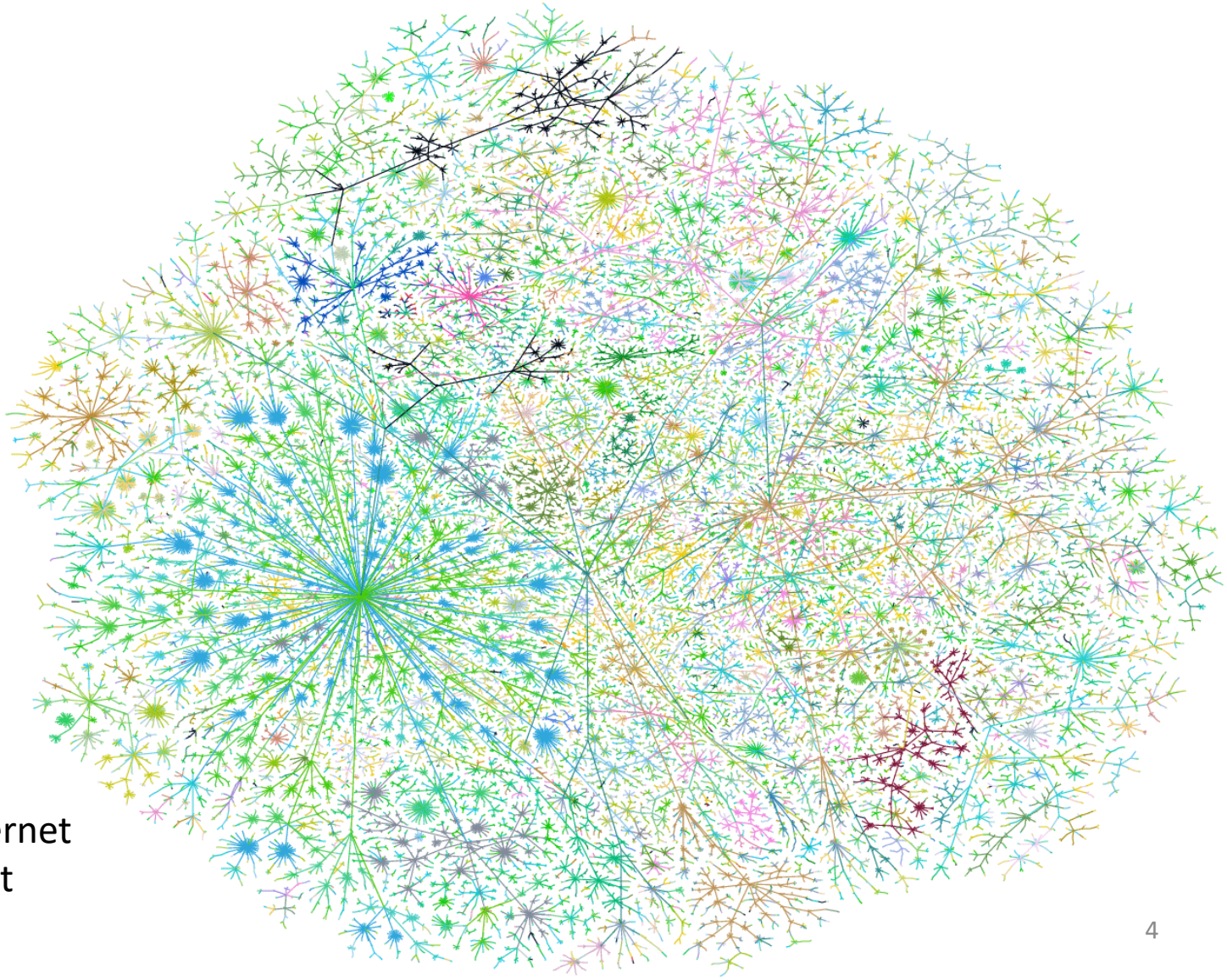
Graphs

Roadmap


6 lectures
Randomized Alg, Quick, Radix, Bucket Sort
Recurrences
Asymptotic Analysis & Sorting
1st class
Insertion Sort
Divide-Conquer, Merge Sort
Asymptotic Analysis
Introduction
9 lectures
BST
2-4 and Red-Black Tree
AVL and B+ Tree
Trie, Suffix Tree
Hashing
Tree & Hashing
We are here
MidTerm
Dynamic Programming
Greedy Algs
Graphs!
13 lectures
Longest, Shortest, Max and Min...
11 lectures
The Future!
1 lecture

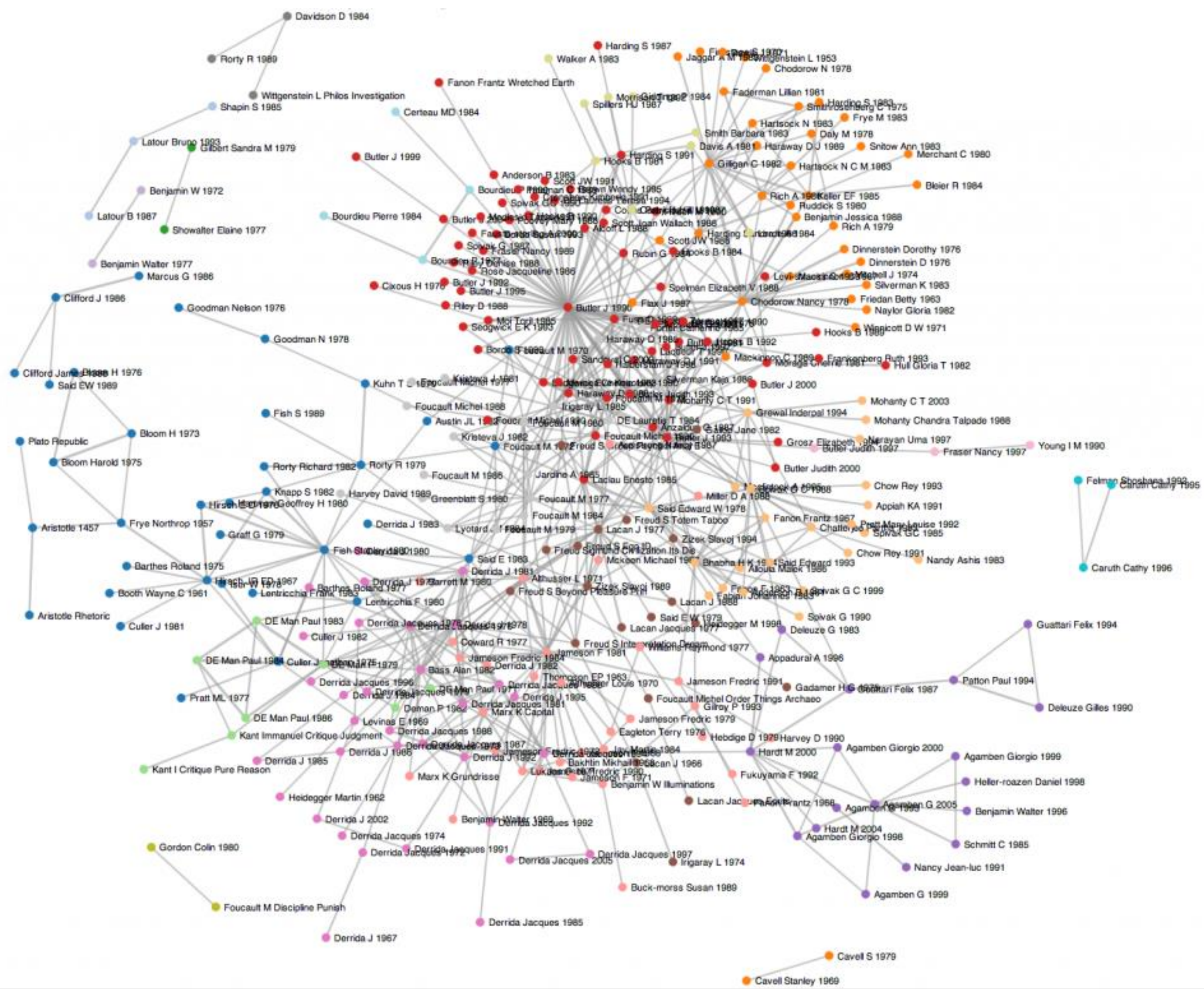
Graphs

Graphs



Graph of the internet (in 1999…it's a lot bigger now…)

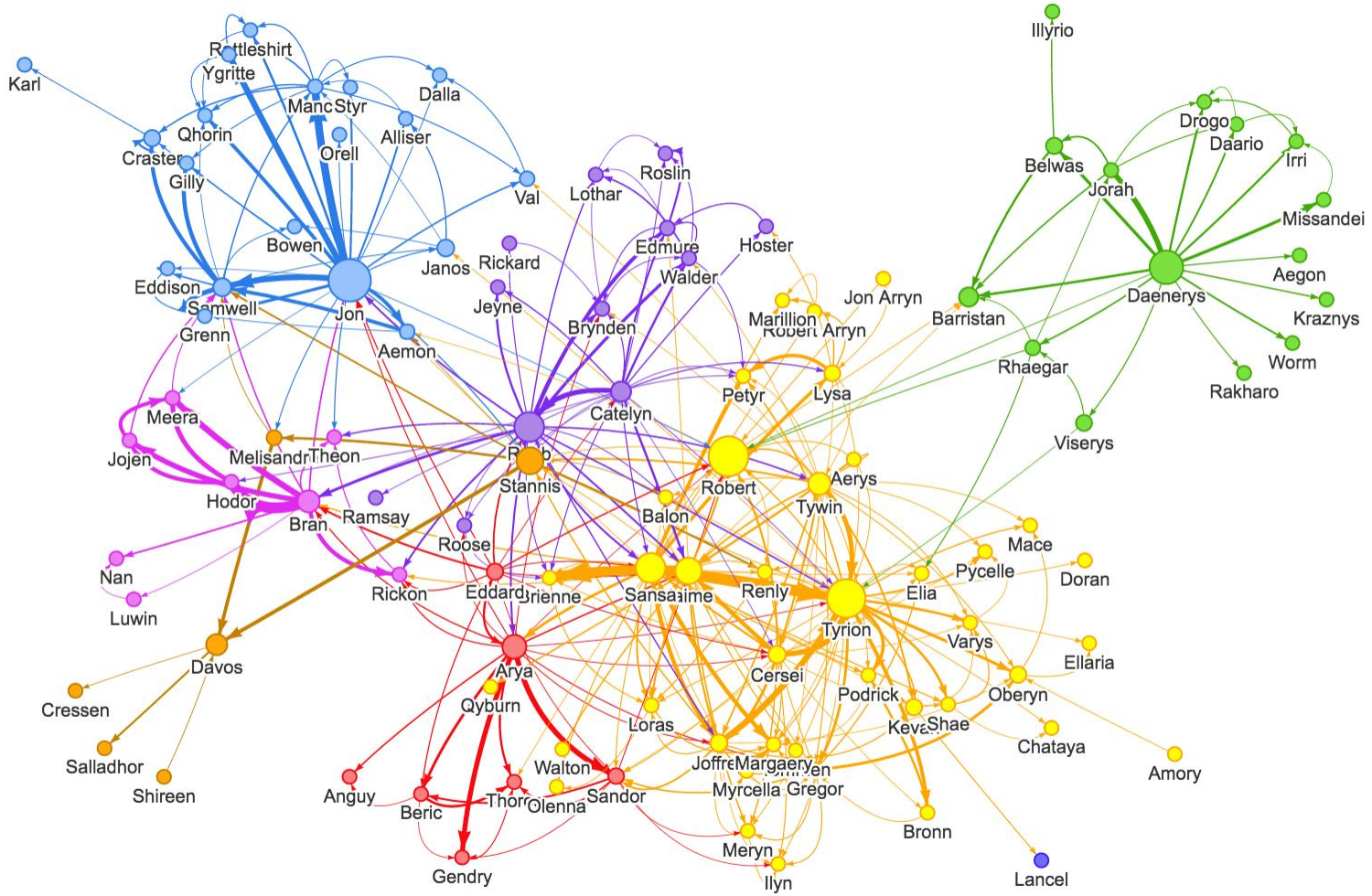
Graphs



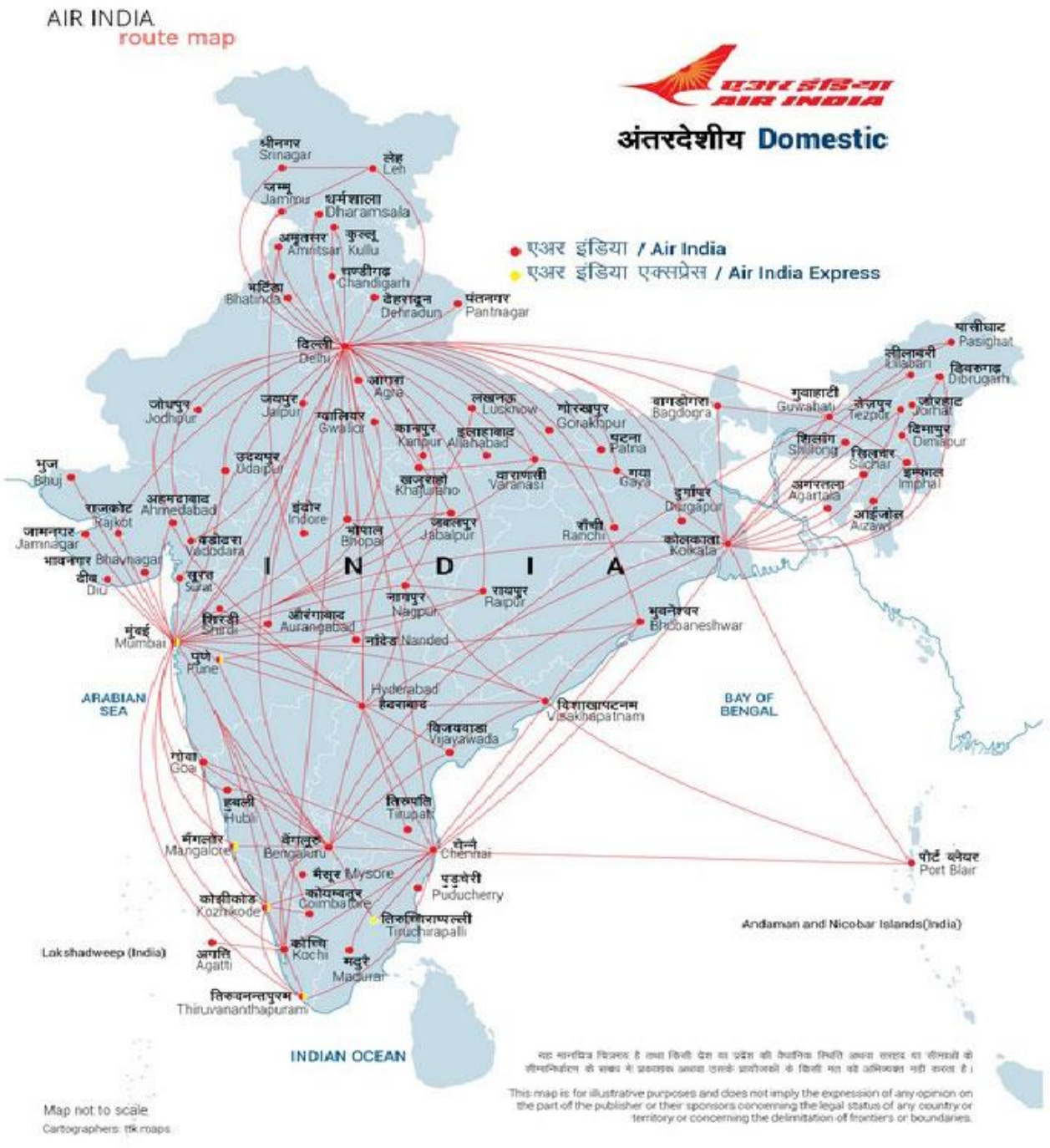
Citation graph of literary theory academic papers

Graphs

Game of Thrones Character Interaction Network

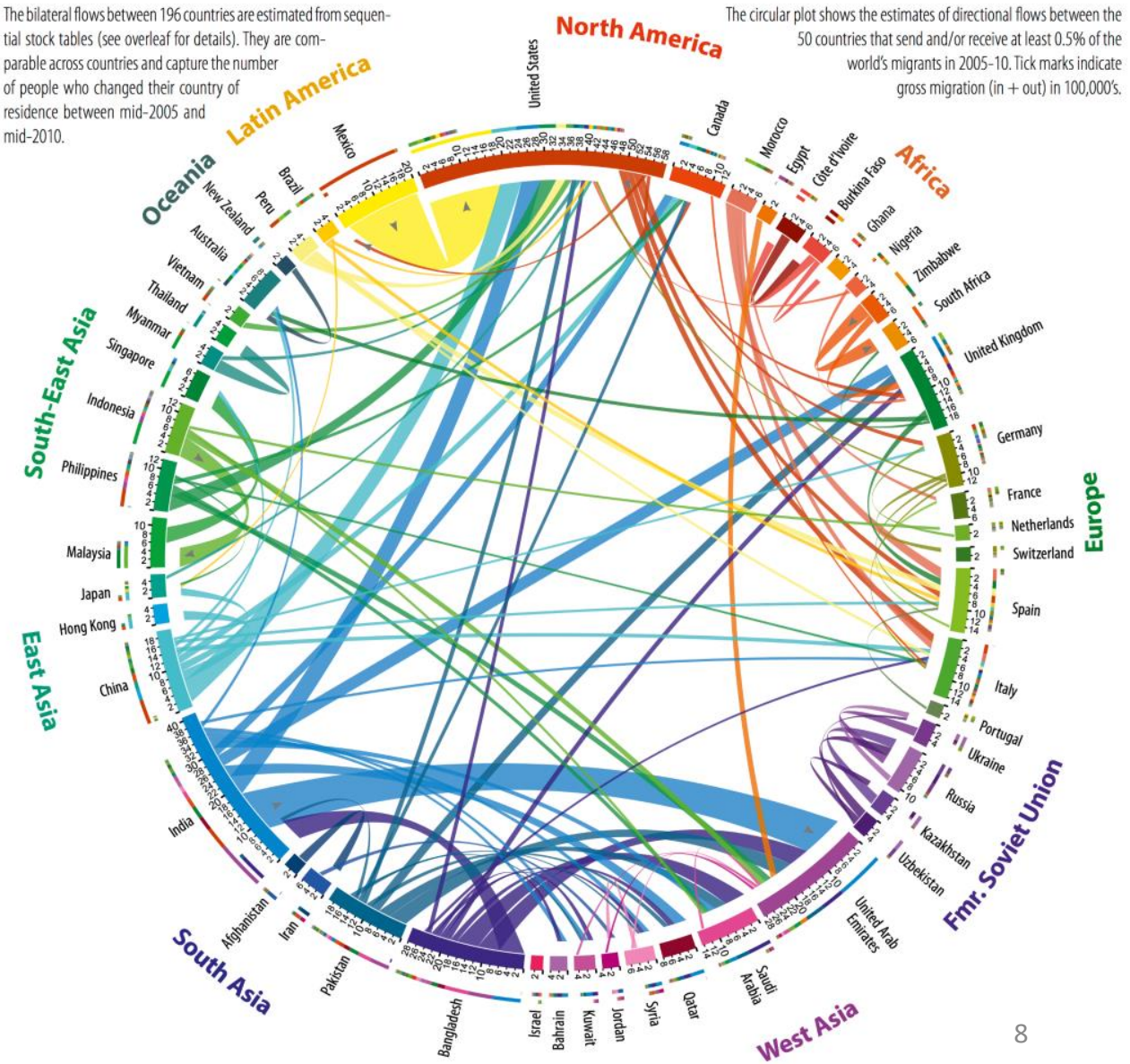


Graphs


AIR INDIA
route map
एअर इंडिया
AIR INDIA
अंतरदेशीय Domestic
एअर इंडिया / Air India
एअर इंडिया एक्सप्रेस / Air India Express
श्रीनगर Srinagar
लेह Leh
जम्मू Jammu
धर्मशाला Dharamsala
अमृतसर Amritsar
कुल्लू Kullu
चण्डीगढ़ Chandigarh
भटिंडा Bhatinda
देहरादून Dehradun
पंतनगर Pantnagar
दिल्ली Delhi
आगरा Agra
जोधपुर Jodhpur
जयपुर Jaipur
ग्वालियर Gwalior
लखनऊ Lucknow
गोरखपुर Gorakhpur
बागडोगरा Bagdogra
गुवाहाटी Guwahati
तेजपुर Tezpur
जोरहाट Jorhat
पासीघाट Pasighat
डिब्रुगढ़ Dibrugarh
दिमापुर Dimapur
शिलांग Shillong
सिलचर Silchar
इम्फाल Imphal
अगरतला Agartala
आईजोल Aizawl
कानपुर Kanpur
इलाहाबाद Allahabad
पटना Patna
उदयपुर Udaipur
खजुराहो Khajuraho
वाराणसी Varanasi
गया Gaya
दुर्गापुर Durgapur
भुज Bhuj
अहमदाबाद Ahmedabad
राजकोट Rajkot
इंदौर Indore
भोपाल Bhopal
जबलपुर Jabalpur
राँची Ranchi
कोलकाता Kolkata
जामनगर Jamnagar
वडोदरा Vadodara
भावनगर Bhavnagar
दीव Diu
सूरत Surat
I N D I A
नागपुर Nagpur
रायपुर Raipur
शिरडी Shirdi
औरंगाबाद Aurangabad
भुवनेश्वर Bhubaneshwar
मुंबई Mumbai
नांदेड Nanded
पुणे Pune
ARABIAN SEA
Hyderabad
हैदराबाद
विशाखापटनम Visakhapatnam
BAY OF BENGAL
विजयवाड़ा Vijayawada
गोवा Goa
हुबली Hubli
तिरुपति Tirupati
मंगलोर Mangalore
बेंगलूरु Bengaluru
चेन्नै Chennai
पोर्ट ब्लेयर Port Blair
मैसूर Mysore
पुडुचेरी Puducherry
कोझिकोड Kozhikode
कोयम्बतूर Coimbatore
तिरुचिरापल्ली Tiruchirapalli
Andaman and Nicobar Islands(India)
Lakshadweep (India)
अगत्ति Agatti
कोच्चि Kochi
मदुरै Madurai
तिरुवनन्तपुरम Thiruvananthapuram
INDIAN OCEAN
Map not to scale
Cartographers: ttk maps
This map is for illustrative purposes and does not imply the expression of any opinion on the part of the publisher or their sponsors concerning the legal status of any country or territory or concerning the delimitation of frontiers or boundaries.

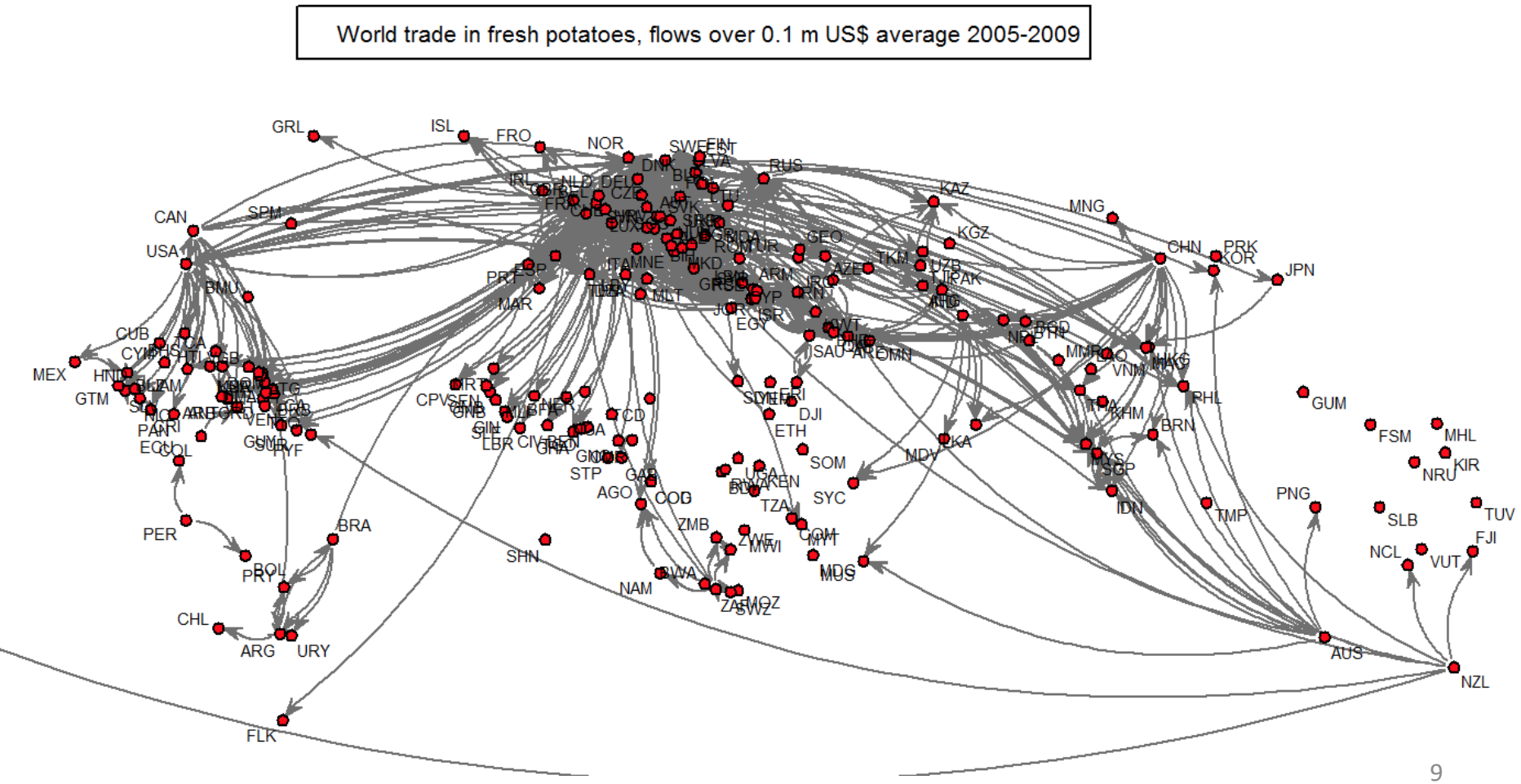
Graphs

Immigration flows



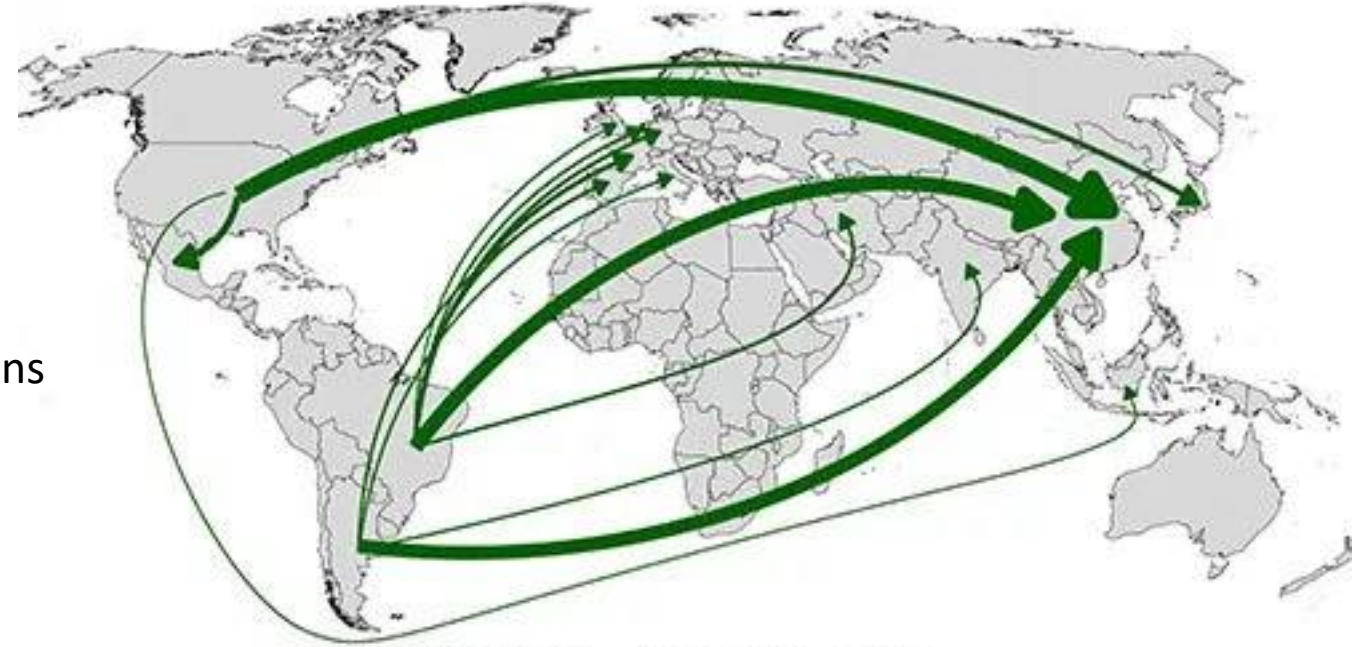
Graphs

Potato trade

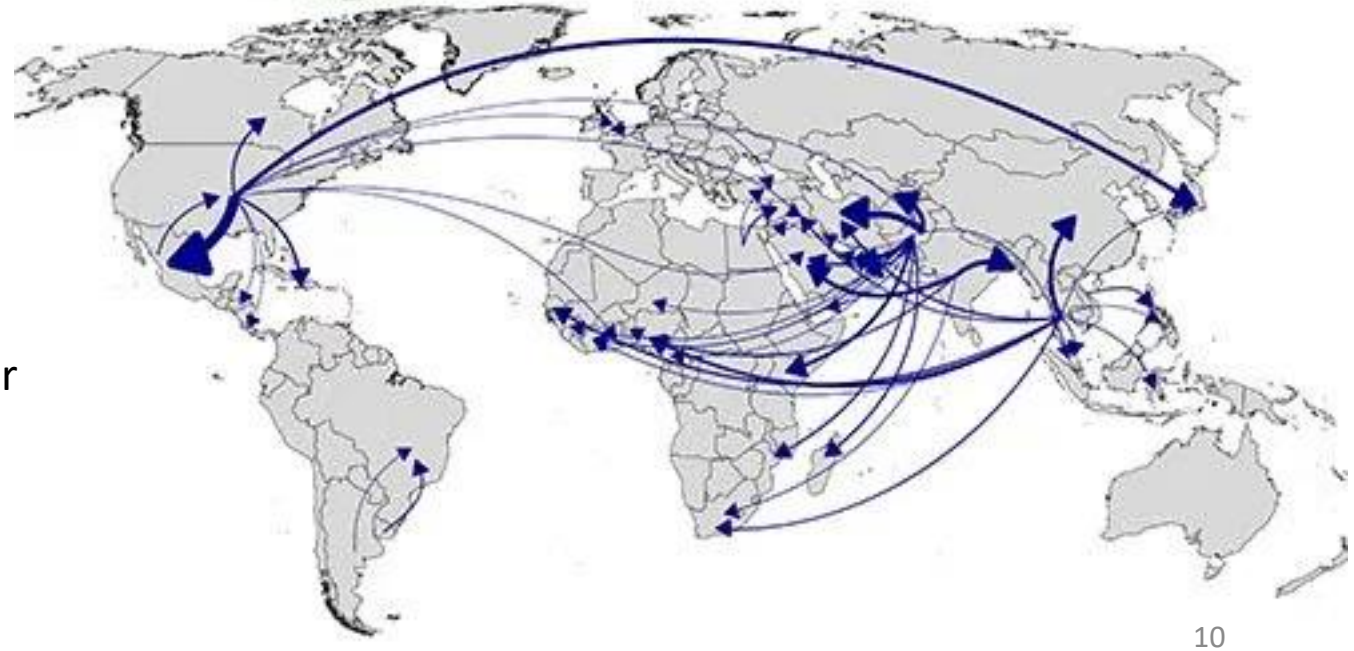


Graphs

Soybeans

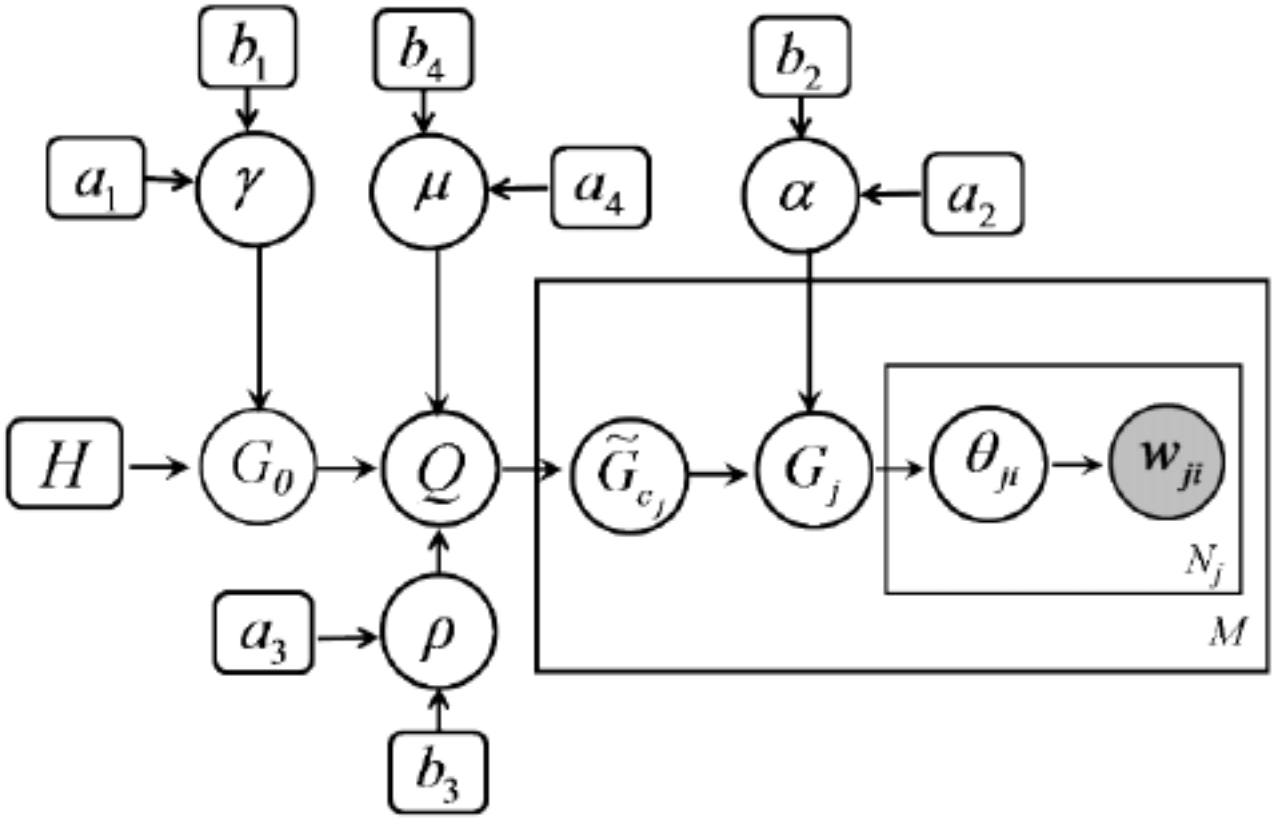


Water



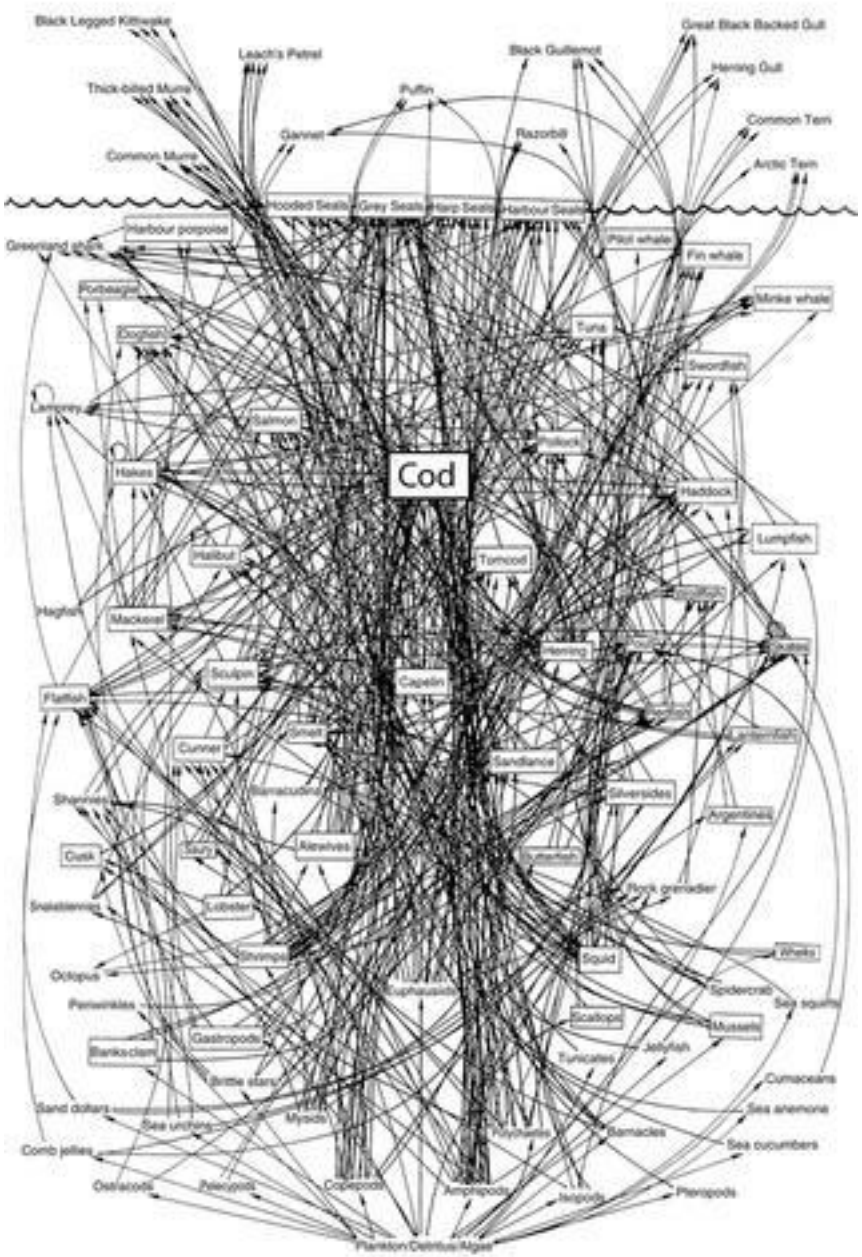
Graphs

Graphical models



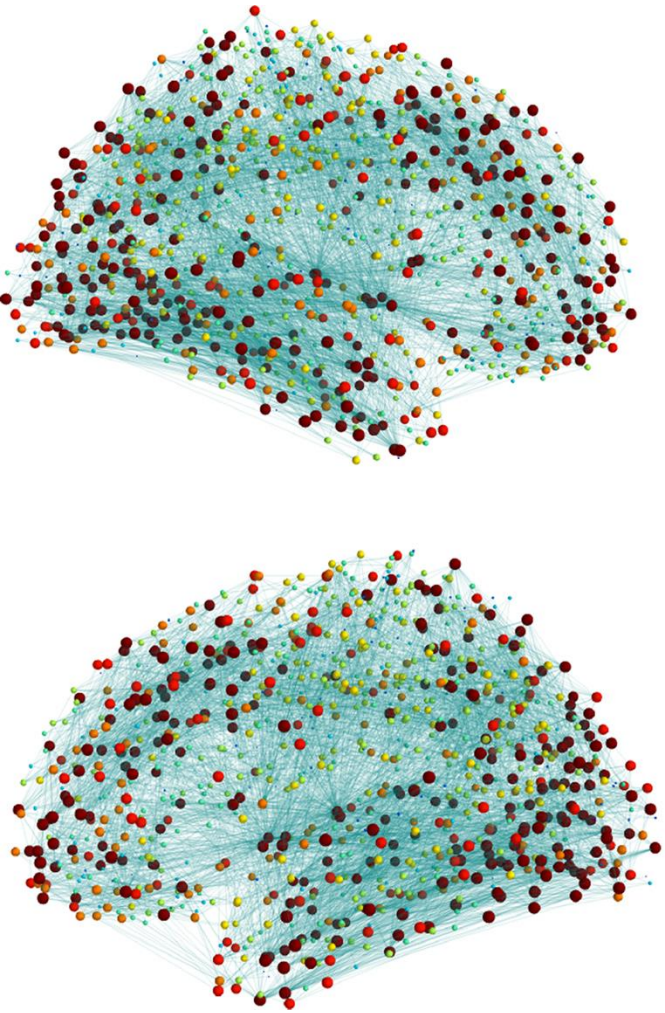
Graphs

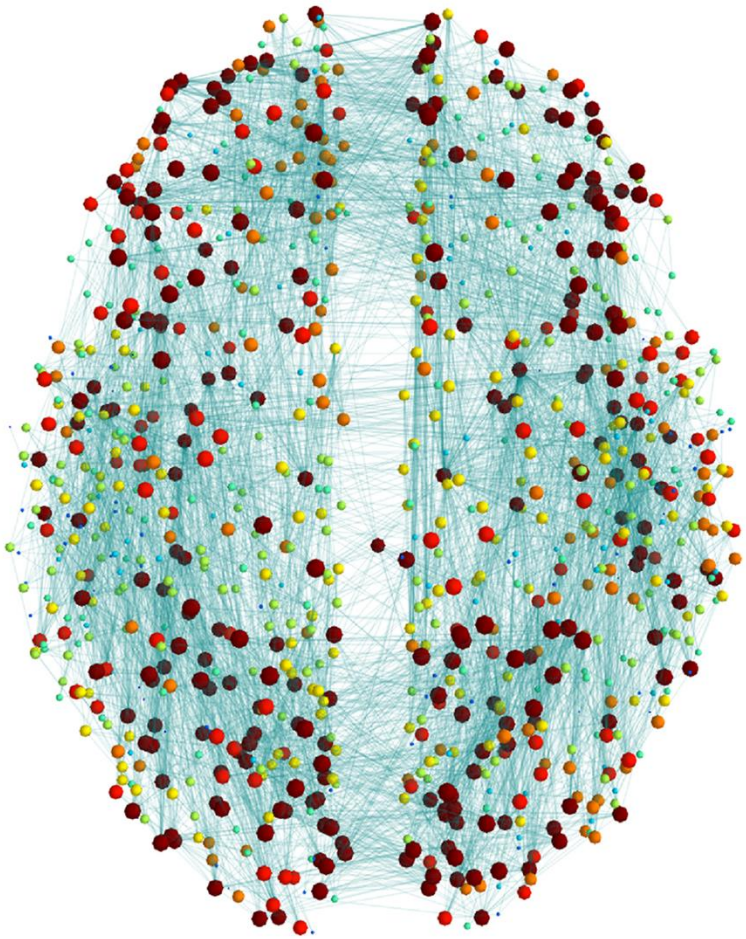
What eats what in the Atlantic ocean?



Graphs

Neural connections
in the brain





Graphs

- **There are a lot of graphs.**

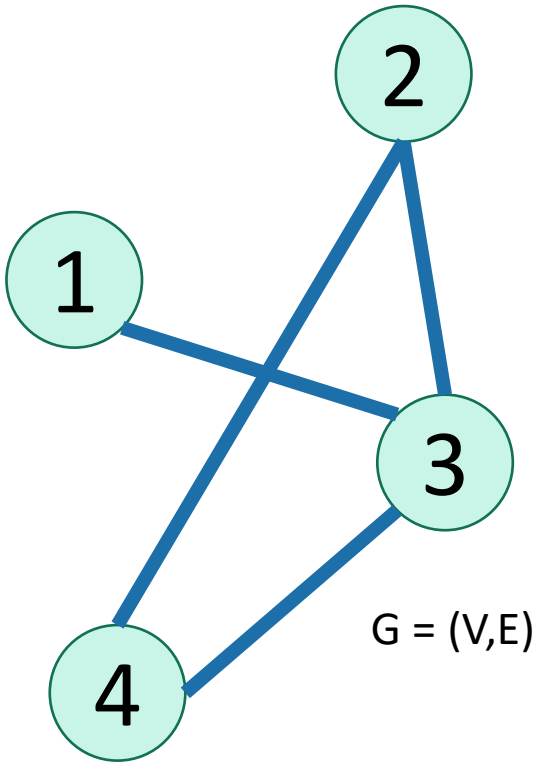
- We want to answer questions about them.
  - Efficient routing?
  - Community detection/clustering?
  - Signing up for classes without violating pre-req constraints
  - How to distribute fish in tanks so that none of them will fight.

- This is what we'll do for the next several lectures.

Undirected Graphs

- Has vertices and edges
  - V is the set of vertices
  - E is the set of edges
  - Formally, a graph is G = (V,E)

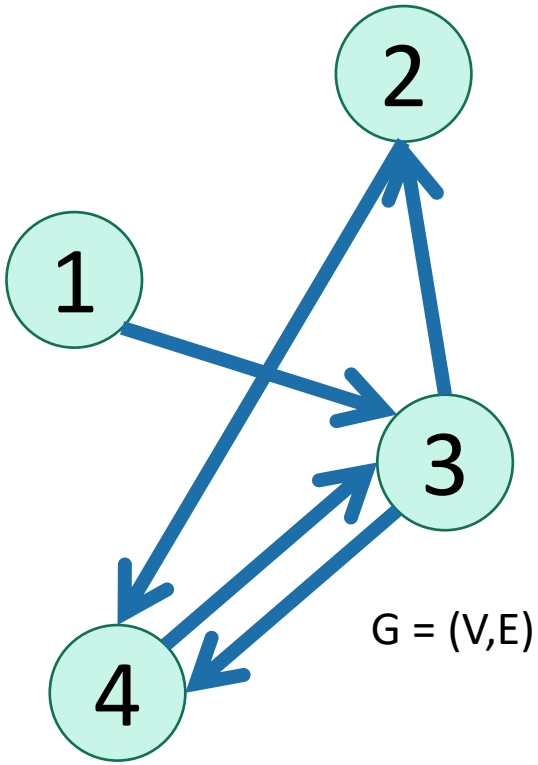
- Example
  - V = {1,2,3,4}
  - E = { {1,3}, {2,4}, {3,4}, {2,3} }



- The **degree** of vertex 4 is 2.
  - There are 2 edges coming out.
- Vertex 4's **neighbors** are 2 and 3

Directed Graphs

- Has vertices and edges
  - V is the set of vertices
  - E is the set of **DIRECTED** edges
  - Formally, a graph is G = (V,E)



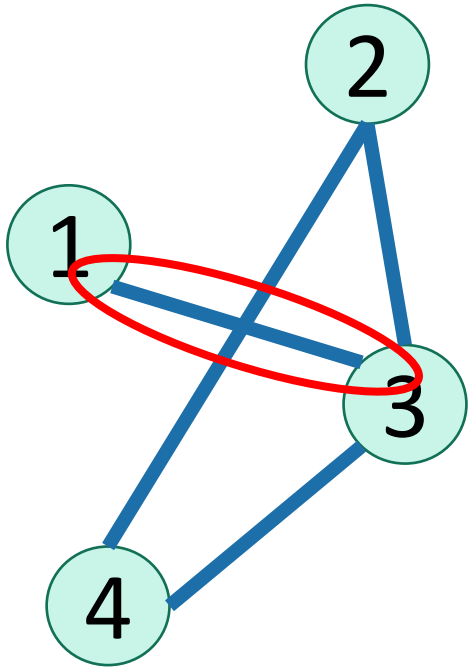
- Example
  - V = {1,2,3,4}
  - E = { (1,3), (2,4), (3,4), (4,3), (3,2) }

- The **in-degree** of vertex 4 is 2.
- The **out-degree** of vertex 4 is 1.
- Vertex 4's **incoming neighbors** are 2,3
- Vertex 4's **outgoing neighbor** is 3.

How do we represent graphs?

- Option 1: adjacency matrix

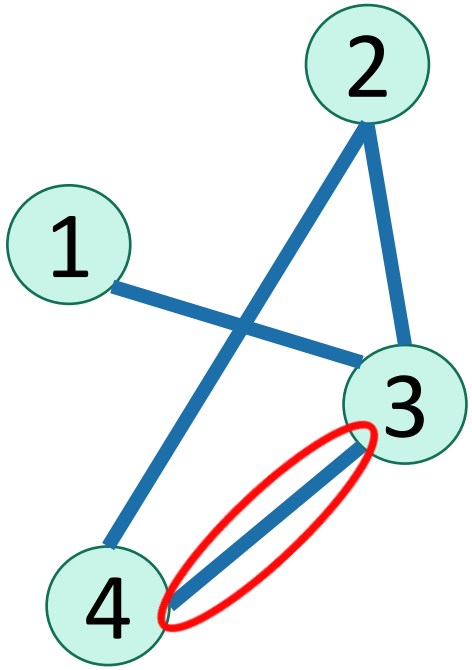
	1	2	3	4
1	0	0	1	0
2	0	0	1	1
3	1	1	0	1
4	0	1	1	0



How do we represent graphs?

- Option 1: adjacency matrix

	1	2	3	4
1	0	0	1	0
2	0	0	1	1
3	1	1	0	1
4	0	1	1	0

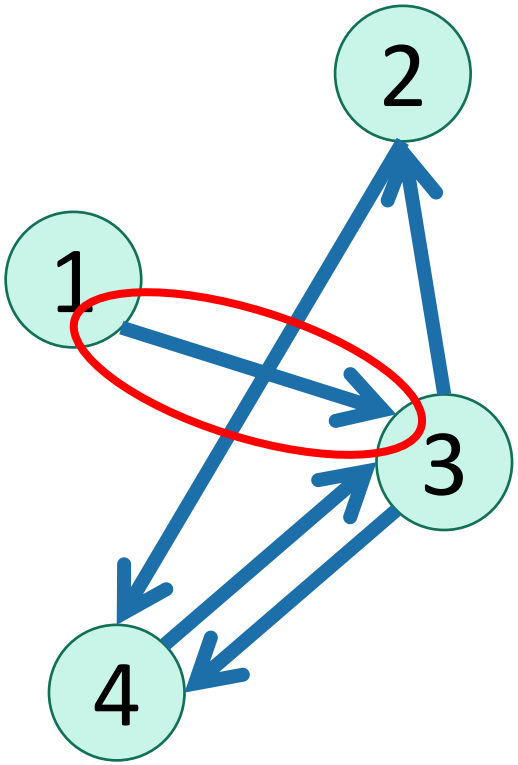


How do we represent graphs?

- Option 1: adjacency matrix

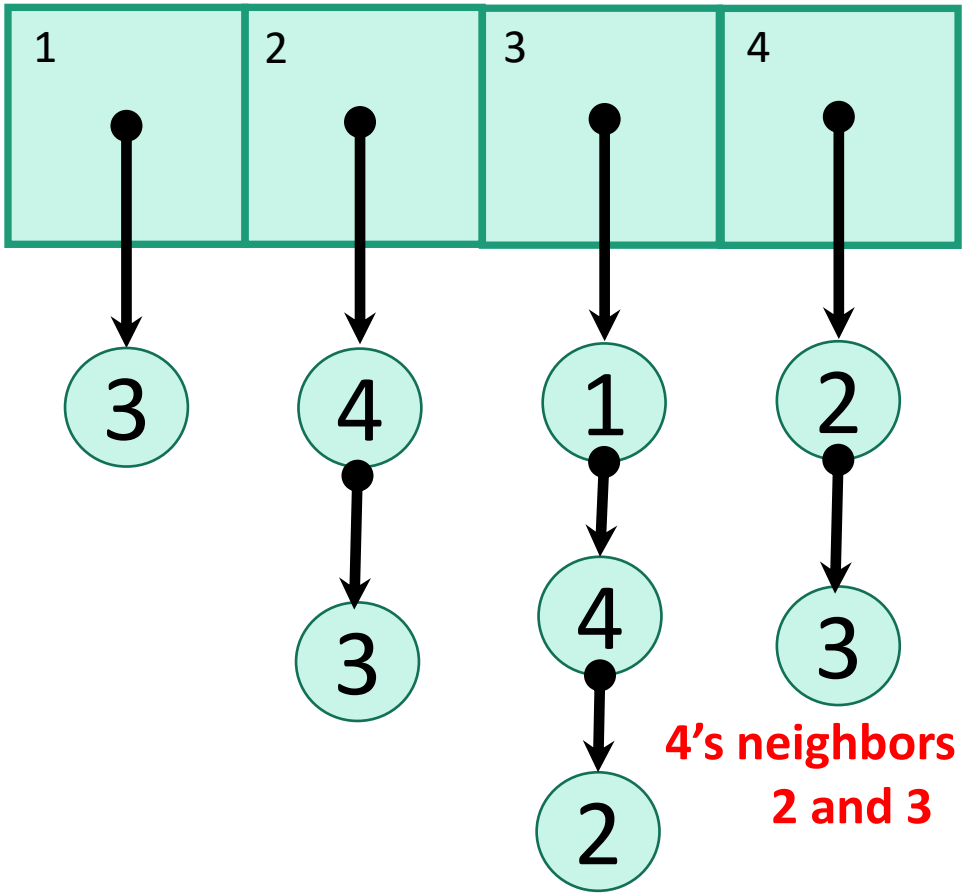
Destination

Source	1	2	3	4
1	0	0	1	0
2	0	0	0	1
3	0	1	0	1
4	0	0	1	0

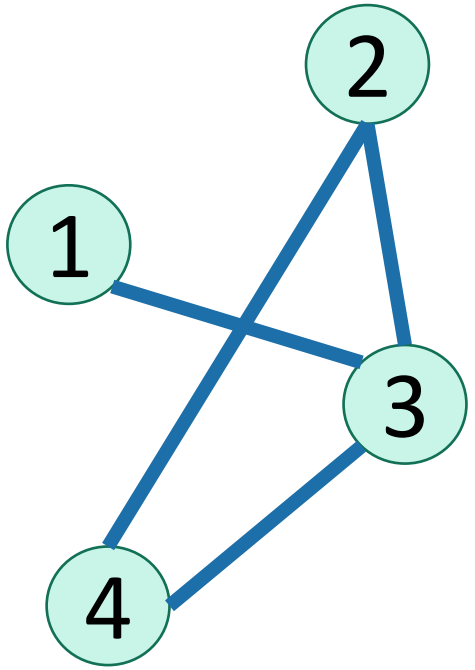


How do we represent graphs?

- Option 2: adjacency lists.



4's neighbors are 2 and 3



How would you modify this for directed graphs?

In either case

- Vertices can store other information
  - Attributes (name, IP address, …)
  - helper info for algorithms that we will perform on the graph

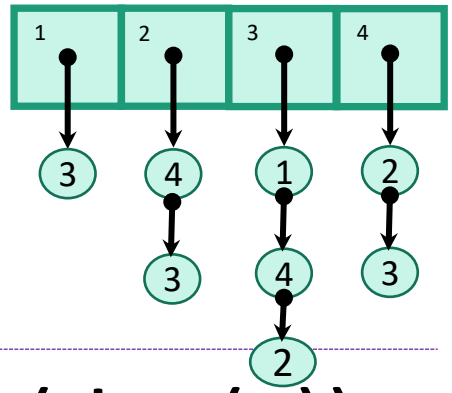
- Want to be able to do the following operations:
  - **Edge Membership**: Is edge e in E?
  - **Neighbor Query**: What are the neighbors of vertex v?

Trade-offs

Say there are n vertices and m edges.

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Generally better for **sparse** graphs



Edge membership Is e = {v,w} in E?	O(1)	O(deg(v)) or O(deg(w))
Neighbor query Give me v's neighbors.	O(n)	O(deg(v))
Space requirements	$O(n^2)$	O(n + m)

We'll assume this representation for the rest of the class

Acknowledgement

- Stanford University