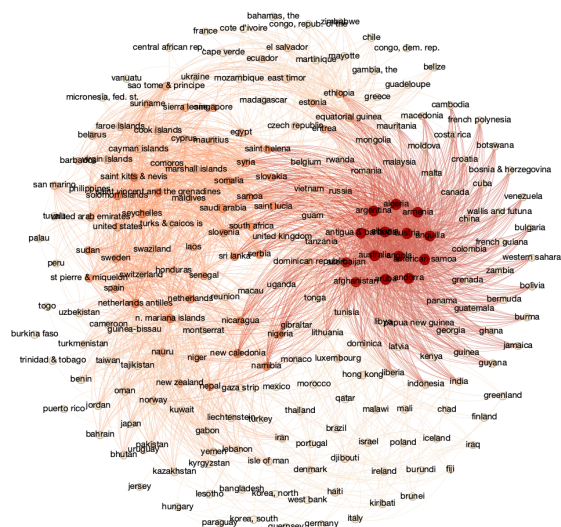


3. A-T-L-A-S Atlas!

Graphs are all around us, and nearly everything can be represented as a graph. Some common examples are social networks, traffic flows, molecules, etc. But, something more fun that can also have a graphical representation is *games*. Specifically, the game of Atlas. For those unfamiliar, *The Game of Atlas is a word game where players take turns naming places (e.g., cities, countries, or states) that start with the last letter of the previous place. For example, if one player says "India," the next player must say a place starting with "A," like "Australia." Repeating a place or failing to come up with a valid name results in elimination or loss of points. The game continues until one player is left with no possible places to say out loud.*

For simplicity, let us assume there are only two players in the game. Then, Atlas can be represented as a **directed** graph, where there exists an edge from A to B if you can say B after your opponent says A. This leads you to a graph like this. What do you think the color of the nodes signify?



In graph terms, we can gauge that the end condition for the game would be saying a place name that will have no *valid, non-repeatable, outgoing* edges to other nodes, thus trapping your opponent.

Dataset Creation

For this task, you are expected to collect and create your own dataset. A large and important part of research is the quality of data you use to confirm hypotheses. You will be trying to solve the game of Atlas on 3 graphs, with increasing difficulty.

1. **Country Only:** Atlas graph created only with the names of ***all officially recognised countries*** (what “officially recognised” means is up to you, but please do cite your source (:).
2. **City Only:** Atlas graph created with the 500 most ~~densely~~ populated cities in the world.
3. **Country+City:** Atlas graph with data combined from 1 and 2.

[Changed from MOST DENSELY POPULATED to MOST POPULATED]

Use the Networkx library to create the graph from your collected data. Ensure that the graph is directed, and follows the rule “there exists an edge from A to B if you can say B after your opponent says A”. Visualise the graphs using the library’s plotting functions (make them pretty, use GPT 😊).

Task 1 - Graph Analysis

With this information, we now ask: ***“Only using tools and concepts from Graph Theory, can we analyse the graph of the game Atlas, come up with unique, interesting, amusing insights, and, if possible, build a “good” strategy for a given set of places?”***

A few simple properties include different types of centrality, density, and diameter. Feel free to use properties of graphs that feel well connected to the task at hand. This task is focused on exploratory data analysis, and you are, at the very least, **expected to show plots and metrics to support your findings**. However, more emphasis will be placed on the **qualitative** insights that you come up with. Please perform this task for all 3 graphs, and analyse whether it gets easier or harder to find a “good” strategy as we increase the domain space of the game, ***only through the use of graph theory***. Good strategy does not necessarily mean that you will always win. You just have to convince us qualitatively and quantitatively that your analysis gives you a competitive advantage.

Use **at most 6** different properties/concepts, ideally very different from each other to cover more insights.

Task 2 - Community Detection

Community detection or clustering is an important analysis for graphs. In the study of complex networks, a network is said to have community structure if the nodes of the network can be easily grouped into disjoint sets of nodes such that each set of nodes is densely connected internally, and sparsely between different communities. In this task, you are required to perform community detection on the **Country Only graph**. This is a well-studied problem, and various static algorithms as well as machine learning methods exist for community detection.

1. Implement any two algorithms/ ML methods for community detection on the graph (No need to do it from scratch).
2. You will be answering questions such as: Do the computed communities represent actual concepts or exhibit any patterns that align with human thought? Can we utilise this concept of choosing country names that are between or within communities to get an intuition of a “good” strategy?
3. Objectively assess the quality of your community detection through metrics like *Modularity*.

[Bonus] Link Prediction!

For this task, the directed graph edges are based on the last letter of one country and the first letter of the other. Can a neural network find out whether edges exist between two countries? Take your original graph, mask out some edges, and train a link prediction network on the rest. You are supposed to use 1. Node2vec, and 2. Any GNN. Try to include both, but don't worry if you're only able to do one of them!

1. Note that for this task, you will have to construct node features. These features can be as simple as you want (e.g: Just the first and last letter) or as complex as you wish (e.g: common NLP Techniques).
2. Look into PyG (PyTorch Geometric), a library built upon PyTorch to easily write and train Graph Neural Networks (GNNs) for a wide range of applications related to structured data.
3. Note that your data does not have any labels. This means your link prediction model must be trained in an *unsupervised* manner.


Alongside the code, you are expected to provide:

1. Performance of implemented link prediction GNN/node2vec on your chosen dataset
2. Analysis and intuition behind the choice of constructed features
3. Details on unsupervised training objective

Paper Reading Task: [node2vec: Scalable Feature Learning for Networks](#)

Resources:

- [Networkx Python Library](#)
- <https://networkx.org/documentation/stable/reference/algorithms/community.html>
- <https://pytorch-geometric.readthedocs.io/en/latest/index.html>
- <https://www.youtube.com/playlist?list=PLoROMvodv4rPLKxlpqhjhPgqQy7imNkDn>
- https://en.wikipedia.org/wiki/Graph_property
- <https://en.wikipedia.org/wiki/Centrality>
- <https://www.youtube.com/watch?v=JheGL6uSF-4&t=117s>
- https://pytorch-geometric.readthedocs.io/en/2.6.1/tutorial/shallow_node_embeddings.html?highlight=unsupervised

-  9. Link Prediction on MovieLens.ipynb

For any doubts regarding this task please directly email - akshit.sinha@students.iiit.ac.in, with cc to debangana.mishra@students.iiit.ac.in.