Semantic Data Management

IMPLEMENTATION OF GRAPH ALGORITHMS FOR INSIGHT DISCOVERY ON FOOD WEBS FOR ENVIRONMENTAL IMPACT ASSESSMENTS

Group

Andrés Espinal

Nicole Kovacs

June 2022

TABLE OF CONTENTS

1 BUSINESS SCENARIO BACKGROUND

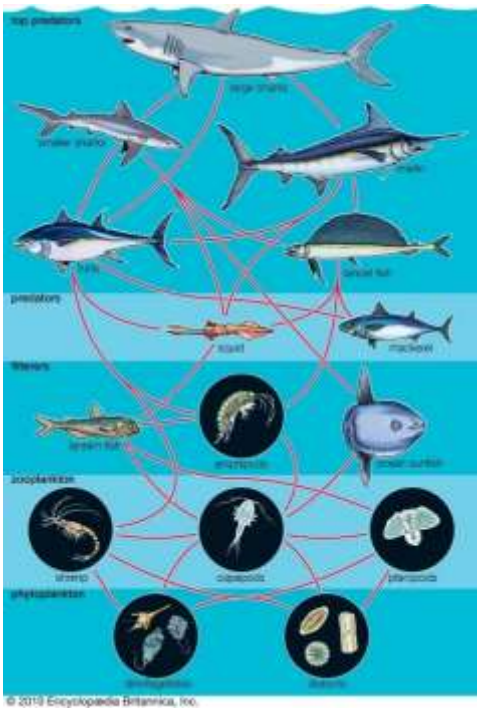
Every time a construction company wants to develop any type of infrastructure project (for instance, a road, bridge, building, park, etc.) they need to comply to strict environmental regulations to be able to acquire a permit for said project. To do this, a study known as *Ecological Impact Assessment*[1] must be performed in the intended location of the project to understand the possible side-effects it might cause. Since construction companies are not domain experts in ecology, they usually rely on other companies to perform these ecological studies for them. These companies provide insights regarding *species distribution, impact on the environmen*t and *other factors* that might be of importance to get a clear idea of the impact of the project. For this purpose, companies that perform ecological impact assessment gather massive amounts of data from all kinds of different sources and schemas. As such, they often encounter that 80% of the time spent in the study is used on collecting, wrangling, and storing the data efficiently. Clearly, this is a problem that deals with *Volume, Variety* and *Variability* to different extents, which makes it an adequate scenario to apply *big data solutions* to tackle most of the problems found in the process. Furthermore, there is a need to acquire insights from this data once it is wrangled. This project intends to demonstrate how graph technologies can be applied to food web data to acquire insights on ecosystem balance, which is a critical aspect of environmental impact assessment studies.

2 PROJECT PROPOSAL

All the code used in this project is presented in the GitHub repository:
https://github.com/nicolezk/sdm_project

The use case that we have chosen for this project is applying graph algorithms to a food web to identify species that are critical for the preservation of ecosystems. Food webs describe the relationships that species share amongst others in a food chain, indicating which species feed on others. Naturally, these relationships can be represented using a directed graph, as shown in the following image:

[1] More information about how Ecological Impact Assessments work can be found here: Ecological Impact Assessment Guidelines



The purpose of our project is to perform insight discovery on a directed property graph to identify species that are tightly connected in a feeding relationship in the food chain. This is an important problem as the reduction or disappearance of a particular species can lead to deep changes in the ecosystem. For instance, some examples of the possible negative side-effects are listed below:

The reduction of members of a species might cause species that depend on it to overfeed on other species to compensate (which can cause a similar problem on other parts of the chain)

The reduction of members of a species might cause other species to proliferate (in the case the feeding species is positioned higher in the food chain). This can be a threat to the balance of the food chain that further endangers the existence of species that are predated by the proliferating species.

Species that depend on the affected species for feeding might not find a secondary food source and could see their numbers decimated in consequence.

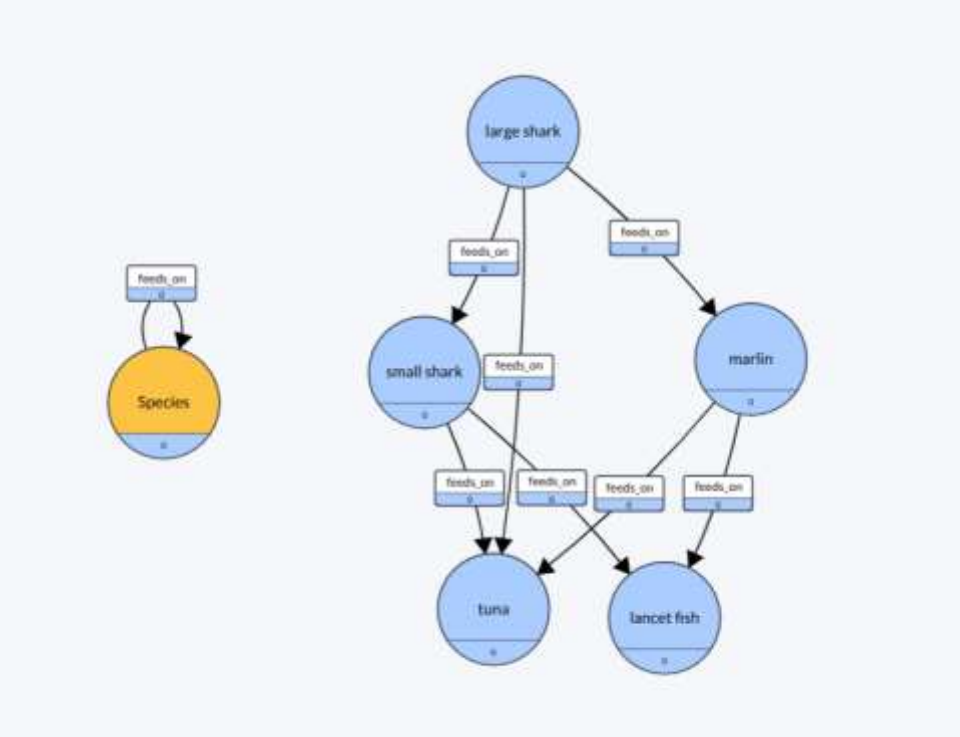
Forcing species to rely on alternate food sources might cause deep evolutionary changes on species as time passes by.

For this purpose, we have decided to apply *centrality algorithms* to a food web graph, which allows us to understand which nodes participate the most in relationships. Considering that species can be represented as nodes and relationships of type *'feeds from'* can be defined between these species, we can model this problem as a property graph.

The end goal is to identify those species that are vital for an ecosystem (those that feed several other species), whose removal would imply a negative side effect for other species that depend on them to feed. Understanding this is vital for infrastructure projects, as the impact in the ecosystem because of such project might cause the reduction or disappearance of species that are critical for other species. We want to understand through this research up to what extent an infrastructure project can be developed while keeping the balance of the food chain in the ecosystem.

3 GRAPH MODELING

The graph being proposed is presented in the image below. The schema is represented using the orange node and an example with instances is represented using the blue nodes.



The graph used was composed of:

- 47163 edges
- 6882 vertices

Details on how it was constructed are specified in the section below.

4 DATA PREPARATION

4.1 DATA SOURCES

The graph has been populated using the following data sources:

1. **Food Web DB** *(*https://www.globalwebdb.com/): Resource from the *University of Canberra* of Australia. Claims to be one of the biggest databases compiling food webs. Food webs can be downloaded from the site as a zipped file containing separate CSV files per publication *(PDF's to the source publication are included as well)* which contain a matrix of data specifying the predation relationship between species. A sample of a CSV file is shown below:

Parin, N.V. (1970). Ichthyofauna of the epipelagic zone. Israel Program for Scientific Translations, Jerusalem. [U.S. Department of Commerce Clearinghouse for Federal Scientific and Technical Information, Springfield, VA 22151], from p. 354.																
Tropical seas - epipel euphausiic copepods shrimps			vertically r flying fishe hyperiid a lanternfish ocean sun other mes snake mac squid								dolphin Cc tuna		tuna		marlin	medium-si large sharks	
coccolithophores	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
dinoflagellates	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
euphausiids	0	0	0	1	1	0	0	0	1	0	0	0	1	0	0	0	0
copepods	1	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
shrimps	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0
vertically migrating m	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0
flying fishes	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0
hyperiid amphipods	0	0	0	0	1	0	1	0	1	0	0	0	1	1	0	0	0
lanternfish	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0
ocean sunfish	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
other mesopelagic fis	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
snake mackerel	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
squid	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0
tuna	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
tuna	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0
marlin	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
medium-sized sharks	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

The first row in every food web matrix file contains the citation for the publication in which the food web was published. The second row and first column contain the species/taxa names lists. An entry of 1 or any other positive number in the matrix indicates a trophic link between the organism in that column with the organism in the corresponding row. In most cases the species row consumes the species column. An entry of 0 indicates the absence of a trophic interaction.

4.2 POPULATING THE GRAPH

Given that the data is given in a format that is not prepared to be fed to graph processing frameworks (A set of matrixes with the feeding relationships), some pre-processing was done to get the data in the required shape. Specifically, a *melt operation* (or unpivot) was applied to each of the matrixes to acquire a table that contained columns for *the predator species*, *the predated* and the *feeding relationship* between them. Some publications encode a weight in the matrix values, but we found this value highly inconsistent between publications, so decided not to use it and instead opted for transforming anything in the *feeding relationship* column that wasn't a '0' or a null value to a 1, to specify there is a feeding relationship between the predator and predated species.

Later, all these datasets are consolidated through a *union* operation and the final dataset is filtered to include only rows where *feeding relationship* is equal to 1 and had no duplicates; this ensures a feeding relationship already specified in one study is not considered twice when found in other study. The resulting dataset was split into two (one for *vertices* and one for *edges*) and fed to different graph processing frameworks to acquire metrics for node importance.

Specific details regarding the transformations carried out can be seen in the python file ***food-web-data-wrangling.py***

5 APPROACH

One of our objectives was to compare the performance of Spark's distributed graph algorithms (GraphX or GraphFrames) with Neo4j's centrality algorithms. For the first implementation (Spark) the same Virtual Machine used for our BDM project was used (A virtual machine powered by UPC's OpenNebula https://virtech.fib.upc.edu/ with a configuration of 1 physical CPU, 4GB's of Ram and 500GB's of disk). For the second implementation (Neo4J) results were run locally (On a laptop with 16Gb's of RAM and a 4 Core professor).

Given that we want to identify which species are the most important in terms of the number of species that feed on them, using centrality algorithms like *PageRank* and *Betweenness* would provide some insight on this matter. On one hand, *PageRank* measures the importance of the species in the graph based on the number of species that feed on them and the importance of each feeding relationship *(In this case we are calculating PageRank only based on incoming edges, as we don't have weights for the edges).* On the other hand, *Betweenness* detects the amount of influence a species has over the rest in the graph, allowing us to find species that act as a *bridge* to traverse the graph to reach other species.

The following graph algorithms were applied in the configurations specified below. Times of execution were recorded for each of the algorithms for comparison.

- ✓ **GraphFrame PageRank through Apache Spark:** We chose a version of this algorithm that auto-calibrates the number of iterations based on a tolerance value (We chose Tol=0.0001 as it was a small enough value to give us accurate results but didn't take long to compute the scores). The standard damping factor of 0.85 was used.
- ✓ **Neo4j PageRank:** To compare the performance of Neo4j's implementation with GraphFrame's implementation, we used the same tolerance value and damping factor.
- ✓ **Neo4j Betweenness:** The algorithm's default parameters (concurrency of 4 threads and sampling size of the node count) were used.

The Apache Spark code is available in the ***graphframes-pagerank.py*** file, while the Neo4j code is available in the ***neo4j-pagerank-betweenness.py*** file. The Neo4j code reads the data from the default Neo4j import directory, so the data needs to be moved there before executing the code.

6 RESULTS AND CONCLUSIONS

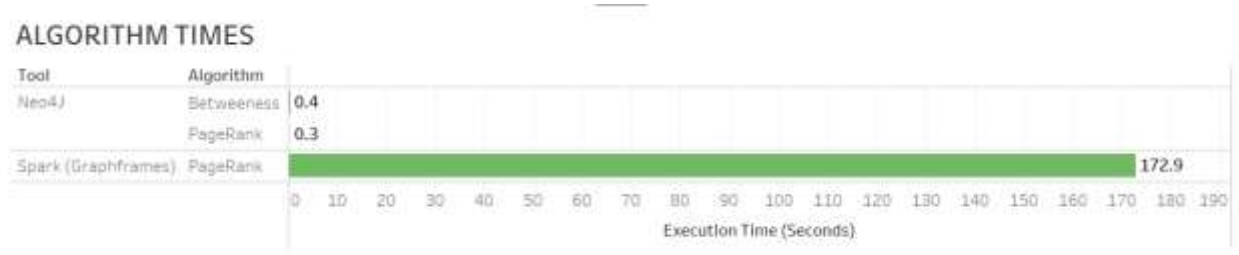
Results of the algorithms can be seen through an interactive dashboard published here *(Scroll at the end of the dashboard)*:

Link: https://public.tableau.com/shared/Y27XTNSQR?:display_count=n&:origin=viz_share_link

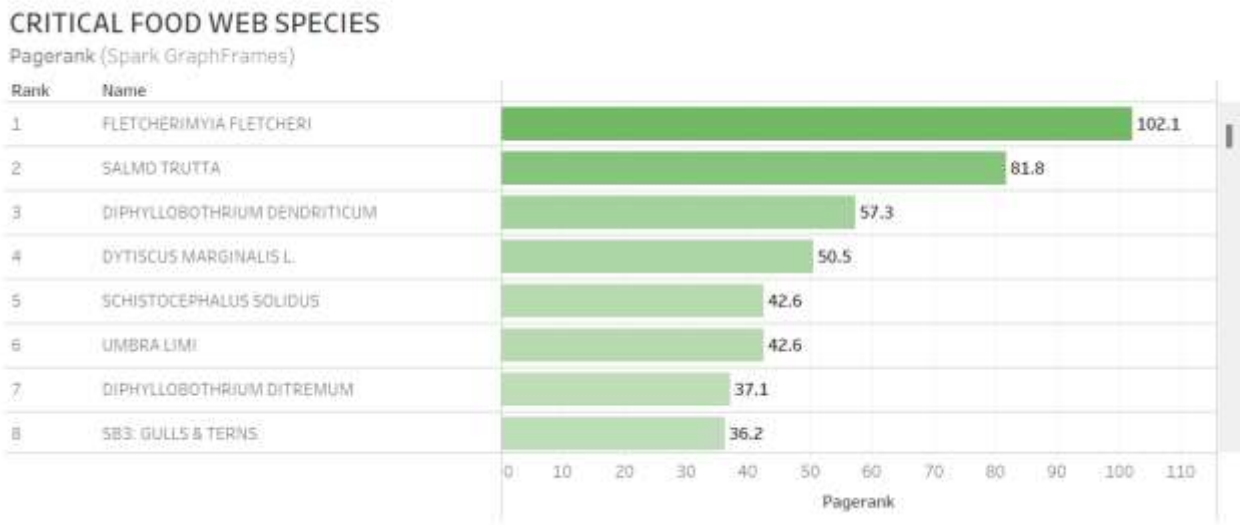
For convenience, results are summarized here:

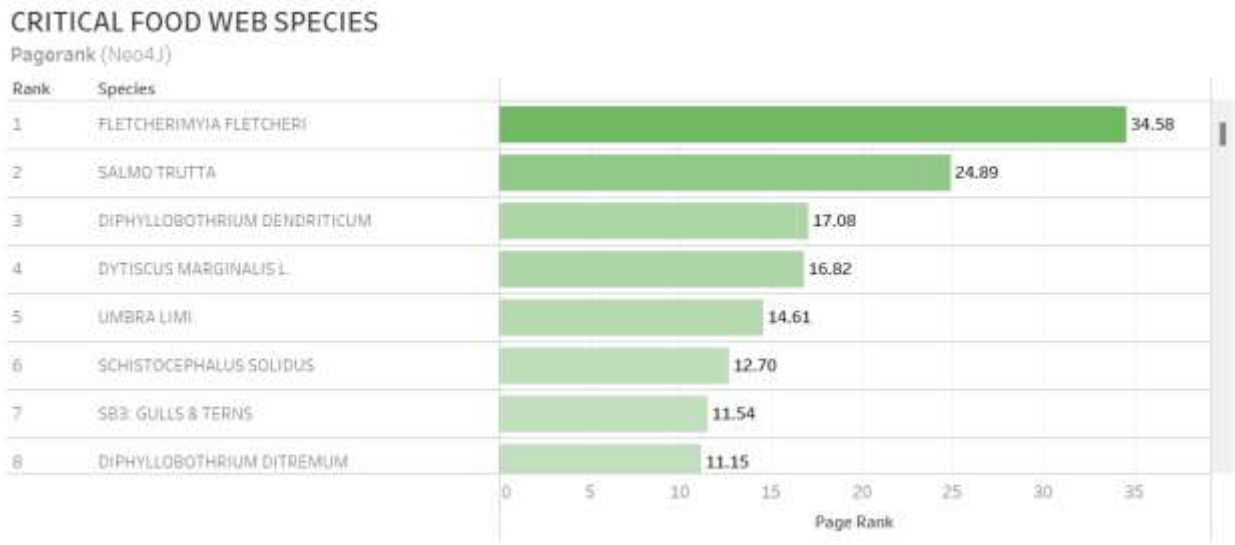
As it is evident in the graph below, Neo4J runs faster than Spark Graphframes implementation of PageRank. This could be caused due to:

- Penalties in starting Spark Jobs (Having to lift the JVM's for the jobs)
- Communication delays (We had to connect through a VPN to the Virtual Machines)
- Iterative nature of the PageRank algorithm based on tolerance measures (It runs until it achieves convergence)

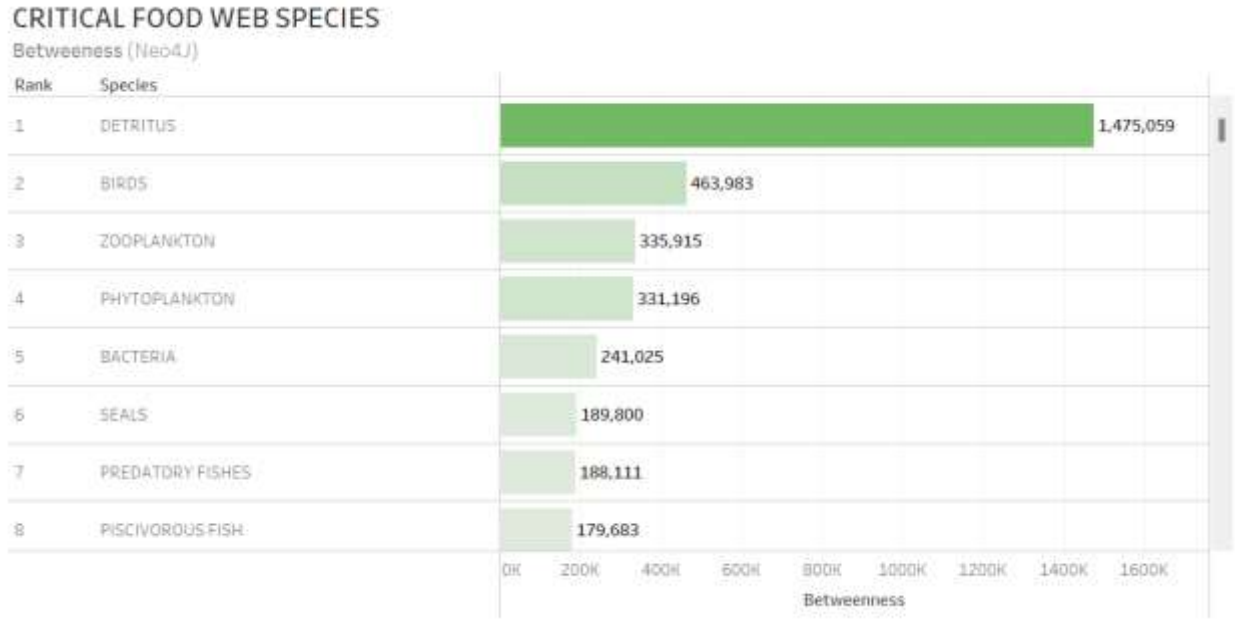


This difference in execution times hints that the size of the graph wasn't big enough to justify using distributed technologies for the calculation of PageRank *(As a refresher, the graph contained 47163 edges and 6882 vertices).* Furthermore, as shown in the graphs below; even though both implementations came up with different values for the PageRank algorithm, the relative ranking of node importance (Species) is kept between both implementations (Neo4J and Spark Graphframe). Only the rows 5 to 8 differ slightly from both implementations (See how *DIPHYLLOBOTHRIUM DITREMUM* is ranked higher <7> by Spark Graphframes and one rank lower in Neo4J <8>). Given that we could get similar results on less time, Neo4J is favoured for this implementation.





The betweenness algorithm from Neo4J exhibited different results, as shown below:



7 RECOMMENDATIONS/NEXT STEPS

- One area that would be worth exploring for this research is applying a weighted algorithm *(one that considers edge importance)* on the graph. Currently we applied unweighted algorithms, but it might be more insightful to use some sort of weight for the edges. For instance, this could be metrics like the inverse of the distance between species are *(it is very unlikely for a turtle for example, to look for food 1000 miles away)* or the number of predated species units a predating species consumes *(some species feed more on some species than others)*. These weights can be fed to algorithms like PageRank to further constraint which nodes are ranked higher.

- Given the way that *FoodWebDB* exposes the data *(through matrices acquired from publications)* a big effort needs to be done on validation. For this project we decided to perform just some simple cleansing, but an improvement that can be done is cross validating the data with other sources to make sure the food web matrices are oriented in the right direction.

- Furthermore, a particular challenge in this project was the difference in notation for species each publication has. For instance, we tried matching the species names to data sources like *iNaturalist* and *GBIF (two of the biggest crowdsourced data sources for species data)* and a very low percentage of species matched through their scientific name. For instance, some species named *BIRDS* and *MAN* appeared often in the dataset, which are very broad. Matching with this data would provide means to acquire geolocation information for species, which can then be used to calculate edge weights. Some techniques like entity resolution could be used here to acquire a more robust dataset.