Significance of Network Metrics

Afonso Ramos & Roger Pujol

October 28, 2019

1 Introduction

In this lab session, we are going to practice on determining the significance of network metrics using collections of global syntactic dependency trees from different languages. In our particular case we are going to study the Closeness Centrality of those graphs and the importance of network metrics applied to a global syntactic dependency tree, generated from large samples of 10 different languages.
The Closeness Centrality is defined as:

$$C = \frac{1}{N} \sum_{i=1}^{N} C_i$$

Where:

$$C_i = \frac{1}{N-1} \sum_{j=1(j \neq i)}^{N} \frac{1}{d_{ij}}$$

Being d_{ij} the shortest path (distance) from node i to node j.
For each language, we are going to compare its original model to two others, which will be our null models N_0:

1. A binomial (**Erdos-Renyi**) graph with the number of vertices and edges the same as those of the real network.

2. A randomised graph generated using the **switching** method. The degree distribution of the original network is maintained in this model.

During this development we opted to mainly use **'R'** due to the big number of libraries available graph manipulation with a simplified multi-edge and loop removal using the one and only iGraph package, our saviour.

2 Results

At table 1, we can see a summary of the properties of each language graph. At table 2, we can see the obtained closeness centrality and the p-values from each model and each language.

Language	N	E	$\langle k \rangle$	δ
Arabic	21532	68743	6.385194	0.0002965582
Basque	12207	25541	4.184648	0.0003428353
Catalan	36865	197074	10.691659	0.0002900298
Chinese	40298	180925	8.979354	0.0002228293
Czech	69303	257222	7.423113	0.0001071125
English	29634	193067	13.030101	0.0004397159
Greek	13283	43961	6.619137	0.0004983540
Hungarian	36126	106681	5.906051	0.0001634893
Italian	14726	55954	7.599348	0.0005160848
Turkish	20409	45620	4.470577	0.0002190600

Table 1: Summary of the properties of the degree sequences.

Language	Closeness Centrality	p-value(binomial)	p-value(switching)
Arabic	0.324366	0	0.4
Basque	0.267085	0	0.2
Catalan	0.337334	0	0.4
Chinese	0.3198	0	0.2
Czech	0.294689	0	?
English	0.345672	0	0.35
Greek	0.314436	0	0.55
Hungarian	0.287117	0	0.75
Italian	0.326907	0	0.4
Turkish	0.353105	0	0.25

Table 2: Closeness Centrality for each language and the according p-values using binomial model and using the switching model.

3 Discussion

3.1 Binomial Model (Erdos-Renyi)

When using the Erdos-Renyi Model, we get very small Closeness Centrality values, which makes sense because all nodes tend to have a similar degree (near the mean degree, $\langle k \rangle$ in table 1), which results in a very distributed net. This is obvious since Erdos-Renyi have the same probability to have an edge between any pair of nodes (probability is δ from table 1). For this reason, the p-value when using this model is always 0.

3.2 Switching Model

For the switching model, in a big contrast to the binomial model, the switching model produces a varying p-value such that: $0 < p - value < 1$ for all languages. In this graph, we are taking two random edges and switching their vertices:

$$\begin{cases} u \sim v \\ s \sim t \end{cases} \Rightarrow \begin{cases} u \sim t \\ s \sim v \end{cases}$$

We can then observe that for all languages p-value ≥ 0.05, which means that many of the randomised models created with the switching method have a larger closeness metric, thus resulting in a p-value above 0.05. This is logical, due to the switching model starting from the original graph, maintaining its closeness. We haven't, however, been able to run the Czech language due to some memory constraints, since for almost all languages up to 14GB of RAM were used due to the extreme size of the adjacency matrix R creates, and this one seems to be the biggest graph.
Regarding the Closeness Centrality, we can observe that the best connected languages in regards to this metric are Turkish and English, with Basque and Czech, having the lowest values.
As previously mentioned on the 1 Introduction, the use of iGraph allowed us to make sure that no loops or hyperedges were introduced during the switching process, which means that a switch when taking two edges, u v and s t and, only creates two new edges u t and s v, if it is safe to do so.
For this process, two separate data structures to implement this process, as it is common on problems like these: an edge list and an adjacency matrix, with the first being used to create the pairings to switch, and with the latter being used to check that the switch was safe to make. As recommended by the guide we first generated two vectors of size Q*E (E = number of edges, Q = log(E)) with numbers chosen at random from 1:E, *ie*, the edge-pairs to be switched, if possible.
For these verifications, we checked if loops wouldn't be created by making sure:

$$t! = u$$
$$v! = s$$

We then verified that no multiedges would be created by making sure: Edges $s \sim v$ or $u \sim t$ do not exist. After this, only then did we proceeded to do the switch and count the failures and successes.

4 Methods

4.1 Optimizations

To optimize the computation of the Closeness Centrality, we first removed the vertices with only one edge and then use the distances from the node attached to it plus one. This way we can avoid recalculating the whole BFS for those nodes. With this optimization we get around a 2× speedup.
We thought about ordering the graph vertices in order to speed up the execution, but we with our implementation the improvement wouldn't be too high (only a bit of cache hit rate improvement) and the cost of ordering in our data structure would be quite expensive. For this reasons we chose no to use ordering.