DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Comparison of Sister-Cities and Airline-Routes networks

Alberto Ursino, Marco Mariotto, and Fabio Marangoni

University of Padova

1 INTRODUCTION

1.1 Project idea

Town-twinning is a particular relationship stipulated between two localities, whose goal is to promote a mutual enrichment of cultural, diplomatic, social and/or economic nature. The fact that such agreement is not exclusive allows city administrations to negotiate more relations of this kind, more likely on an international level. All these links build a dense and capillary network between cities across the world.
A "sister city" bond often materializes in trading opportunities and cultural exchanges, encouraging travels that put in contact citizens and companies from different localities. For example, Boston University fosters study abroad programs for college students in the twinned city of Padua. At this point, a question rises: do these partnerships have any influence on the intensity of air traffic between sister cities (taking into account the fact that generally some destinations have to be reached through one or more stopovers)?
The purpose of the project is to analyze two graphs:

- The first of them is representative of the network of twinning relations around the globe, where each vertex stands for a city;
- The second graph depicts a global mapping of all the airline routes connecting different cities, weighted with the amount of flights during a certain time lapse.

1.2 Intended Experiments

The experiments we decided to do and their goals are the following:

- Extract both local and global features from the two graphs, such as centralities, degree sequences, clustering coefficients, etc. Define a similarity measure between these sets of features (see section 3.2);
- Assess the similarity of paths between pair of nodes on both graphs. Section 3.1 proposes a method which elaborates more on this idea.
- Use clustering techniques. Compare the results to see whether the obtained clusterings are similar or not (see section 3.3);
- Perform the same experiments on modified versions of these graphs.

1.3 Environment

All the work we've done can be found at our GitHub repository [1]. The code is written in Python and the main packages we used are the following: networkx, csv, geopy, cartopy, matplotlib, numpy, math and scipy.

2 DATA

2.1 Graphs

Airline Routes Graph (AR): each node represents an airport in the world: it includes the city name, the geographic coordinates of the airport and the country. Two nodes are linked if there exists a route between the airports. Each edge is weighted proportionally to the air

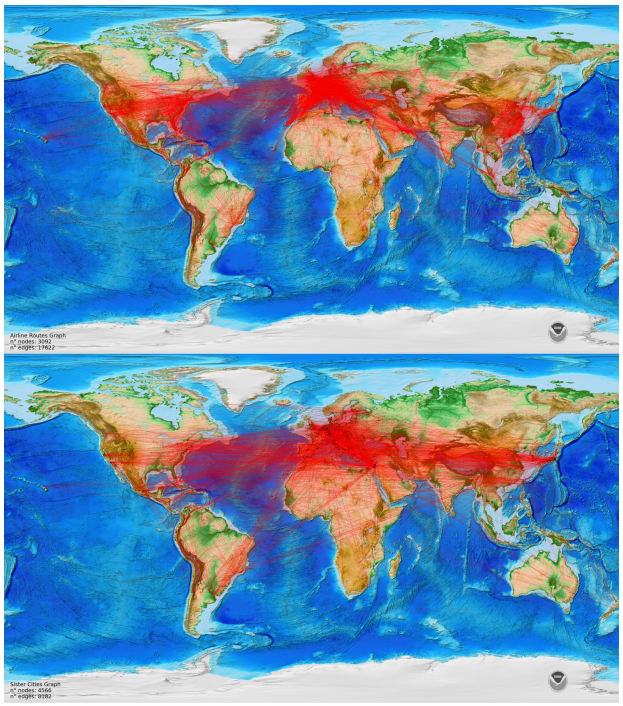

Figure 1: *Airline Routes (AR) and Sister Cities (SC) graphs*

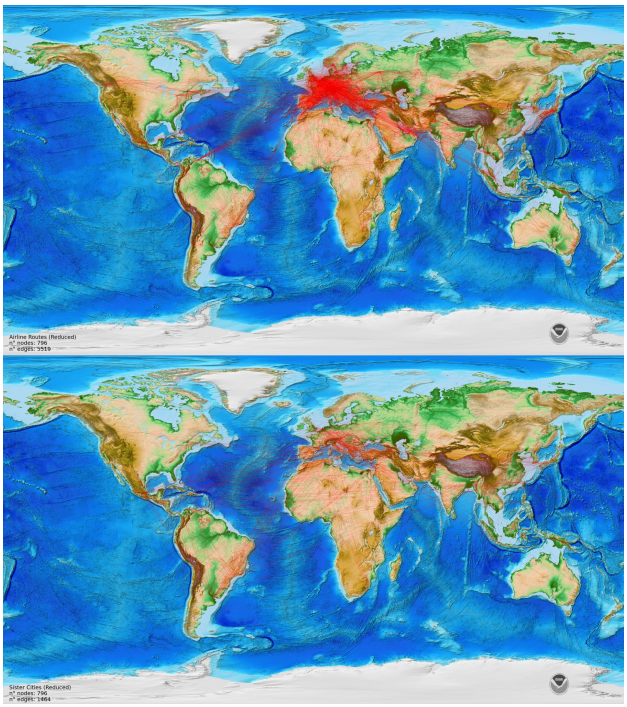

Figure 2: *AR and SC reduced*

traffic between the incident nodes. All the data we used to create this graph is taken from the Tyler Woebkenberg's dataset [2].

Sister Cities Graph (SC): each node represents a city in the world: it includes the city name, the geographic coordinates, the population and the country. Two nodes are linked if they are twinned. All the necessary data for its implementation will be extracted by executing a SPARQL query using the Wikidata's Query Service, since no raw dataset on this topic has been found.

Reduced graphs ($\boldsymbol{R}_{AR}$ & $\boldsymbol{R}_{SC}$): we built a reduced version for the two graphs above. Only nodes appearing on both graphs are kept, the others are discarded (i.e. each twinned city must have an airport and vice-versa). These versions are useful because some algorithms we are going to discuss later (i.e. the DeltaCon method in the section 3.1) require the same set of nodes on both the graphs to be compared as input.

Countries graph ($\boldsymbol{C}_{AR}$ & $\boldsymbol{C}_{SC}$): in addition to the previously described graphs, we decided to integrate an alternative graph structure in our analysis. For both the AR and SC we built a corresponding one as follows: the vertices are representative of the countries around the world; every edge between two countries is weighted with the number of occurrences of edges between the two locations in the original graph. For example, let's say that in the AR country graph there exists an edge between the nodes "Italy" and "Germany"; its weight is equals to the number of cities (from Italy and Germany) connected in the AR graph.
The reason behind this choice is the will of adopting a more macroscopic perspective. Even if this option causes us to lose information about links within a single country, it gives us the advantage of studying the case on the international level, allowing to overlook all those cases in which the air routes graph fails to represent people travels between two locations (e.g. when a traveler leaves from Amsterdam and lands at the Venice airport in order to reach Verona as the final destination by other means).

2.2 Datasets issues

Since the two datasets (OpenFlights and the one created using Wikidata) don't use the same scheme, there were a series of issues, which in part have not been solved yet: we cannot compare cities using Wikidata id, so we relied just on names, which however sometimes are not always equal, due to different encodings, like *Tel Aviv* and *Tel-Aviv* or *San José* and *San Jose*. When we retained common cities, in order to compare the two graphs, we took into account these different encodings, by lower casing the strings and neglecting the diacritic characters. Another problem of these two datasets is that common nodes are very few (≈ 800) with respect to the total number of cities (≈ 4500) or airports (≈ 3000) due to the fact that most "sister cities" are localities without airports.

3 METHODS

In this section we describe all the methods we used for computing a similarity score between the graphs of interest. The results obtained from these methods will be presented later in chapter 4.

3.1 Similarity score by DeltaCon

Our first method of comparison uses the similarity function defined in DeltaCon [3]. According to this paper, the fundamental concept of this method is "to compute the pairwise node affinities in the first graph, and compare them with the ones in the second graph". These measures of affinities are stored in a similarity matrix S where the entry s_{ij} indicates the influence node i has on node j. The node affinity is measured by the so-called Fast Belief Propagation (FaBP) method which is an approximation of the Belief Propagation. In order to compute the affinity between two nodes, BP takes into account the neighbors of both nodes but not only the direct ones, it considers also 2,3,..-step-away neighbors with decreasing weight. As you can see in the DeltaCon pseudocode 1, both graphs should have the same set of nodes V; D is the diagonal matrix with the degree of node i as the d_{ii} entry, the function $RootED(S_1, S_2)$ measures the root euclidean distance (also called Matusita distance) and $\varepsilon = 1/(1 + max_i(d_{ii}))$ is a constant < 1 which encodes the influence between neighbors. Note

Algorithm 1 DeltaCon

1: INPUT = $G_1(V, E_1)$ and $G_2(V, E_2)$
2: $S_1 = [I + \varepsilon_1^2 D_1 - \varepsilon A_1]^{-1}$
3: $S_2 = [I + \varepsilon_2^2 D_2 - \varepsilon A_2]^{-1}$
4: $d(G_1, G_2) = RootED(S1, S2)$
5: return $sim(G_1, G_2) = \frac{1}{1+d}$

that, by temporarily ignoring the term $\varepsilon^2 D$ in the formula for calculating the similarity matrix S, we obtain the following approximation (see the geometric series):

$$S \approx [I - \varepsilon A]^{-1} \approx I + \varepsilon A + \varepsilon^2 A^2 + ...$$

Since A^k has information about the k-step paths and recalling that $\varepsilon < 1$, we can see the intuition behind the influences of 1,2,3..-step-away neighbors with decreasing weights.
Now, once the similarity score d is computed, it is normalized with $\frac{1}{1+d}$ in order to obtain a score between 0 (= totally different graphs) and 1 (= identical graphs).

3.2 Comparing features extracted from the graphs

We based our comparison upon paper [4], whose method we will describe here shortly; for simplicity, and even for our goal, we consider only two graphs $G_1 = (V_1, E_1), G_2 = (V_2, E_2)$ (it is straightforward to compare multiple graphs, just by comparing each pair; results visualization can be done using hierarchical dendrograms). It is not necessary to know any correspondence between nodes in G_1 and G_2, so $|V_1| = |V_2|$ is not required. For each graph, the authors define a matrix of features vectors, one for each node: given a node $i \in V_1$, f_i is a vector of 7 elements, namely: the degree $d_i = \deg(i)$, the clustering coefficient of i, the average number of i's two-hop away neighbors (computed as $\frac{1}{d_i}\sum_{j \in N(i)} d_j$), the average clustering coefficient for nodes in $N(i)$, the number of edges in node i's egonet (denoted ego(i)), the number of outgoing edges from ego(i) and finally the number of neighbors of ego(i). All these vectors, one for each node, are stacked horizontally, obtaining a $|V_1| \times 7$ matrix. After this step, the matrix is kind of shrank: for each column (i.e. each feature) j we form a vector v_j by appending the following

statistics of the elements in the column: median, mean, standard deviation, skewness, kurtosis. All these vectors v_j are then appended together: the result is a signature vector s_1, representative of the graph G_1, consisting of $7 \cdot 5 = 35$ elements. Denoting by s_2 the signature vector for G_2, NetSimile may now assess the similarity between s_1 and s_2 using any available distance metric: in the paper, the authors suggest using the Canberra distance, which is more sensible to values near zero. Of course, once the distance has been computed, we prefer to normalize this value, by giving instead a similarity score in $[0,1]$; a way to do this, given distance d, is returning the value $\frac{1}{1+d}$.
An advantage of this algorithm is of course its scalability to huge networks, since the complexity is linear on the total number of edges; moreover the signatures are independent on the size of the networks and can be also used in many graph mining tasks.
One may wonder if computing a distance between signature vectors is "inferior" to statistical hypothesis testing, or more precisely less discriminating, i.e. less able to make fine distinctions among graphs. In the paper, the authors performed statistical tests, such as Kolmogorov-Smirnov and Mann-Whitney across the features of the graphs: for instance, one can assess whether the two degree columns are similar or not. Doing this for all features produces p-values, from which the maximum one is taken. They showed that these tests do not have enough discriminating power to effectively capture differences between graphs.
We also tried a modification of this algorithm: for instance we added centralities measures. For more details, see 4.

3.3 Comparing clusterings of the graphs

After having applied the same clustering algorithm on each graph, we thought it could be interesting to compare the two results and to quantify their differences. In this respect, we first adopted the Louvain algorithm for community detection, developed by Blondel *et al.* [5] and aimed at the application on large networks. This method is an iterative heuristic for modularity optimization, being

$$Q = \frac{1}{2m} \sum_{i,j} \left[A_{i,j} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j)$$

the definition of the modularity of a partition, where $A_{i,j}$ is the weight between nodes i and j reported on the adjacency matrix A of the graph, $m = \frac{1}{2} \sum_{i,j} A_{i,j}$, $k_i = \sum_j A_{i,j}$, c_i is the cluster to which i belongs, $\delta(c_i, c_j)$ is 1 when $c_i = c_j$ and 0 otherwise. Its execution follows two phases that are repeated until it can't reach a higher modularity. Starting from the situation in which each node belongs to a different community, the first phase aims to optimize modularity moving nodes to other clusters in such a way that the modularity gain ΔQ is maximum and positive; for a node i moving into the community C the gain is computed as

$$\Delta Q = \left[\frac{\sum_{in} + k_{i,in}}{2m} - \left(\frac{\sum_{tot} + k_i}{2m} \right)^2 \right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m} \right)^2 - \left(\frac{k_i}{2m} \right)^2 \right]$$

where $\sum_{in}$ is the sum of the weights of the edges inside C, $\sum_{tot}$ is the sum of the weights of all the edges incident to the nodes in C, k_i is the sum of the weights of the edges incident to i, $k_{i,in}$ is the sum of the weights of the edges connecting i to the nodes in C. Then the second phase follows: the network is rebuilt, merging each community into a single new vertex, and the algorithm reiterates the procedure as long as other moves for the increase of modularity are possible. This is how the hierarchical clustering is performed by the Louvain method.
The community partitions of the sister-cities and the airline-routes graphs, achieved through the previously described algorithm, have then been compared using the Adjusted Rand Index (ARI) [6] as a similarity score. Referring to the following contingency table for two given clusterings $X = \{X_1, X_2, ..., X_r\}$ and $Y = \{Y_1, Y_2, ..., Y_s\}$ for a set of n nodes

	Y_1	Y_2	$\cdots$	Y_s	$\sum_{j=1}^{s} n_{ij}$
X_1	n_{11}	n_{12}	$\cdots$	n_{1s}	a_1
X_2	n_{21}	n_{22}	$\cdots$	n_{2s}	a_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
X_r	n_{r1}	n_{r2}	$\cdots$	n_{rs}	a_r
$\sum_{i=1}^{r} n_{ij}$	b_1	b_2	$\cdots$	b_s	

where the elements n_{ij} express the number of nodes in common between the clusters X_i and Y_j, the Adjusted Rand Index is defined as:

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right]/\binom{n}{2}}$$

and its value ranges between -1 and +1: for completely identical partitions the score given by ARI is +1, while it is close to 0 when a partition is compared to a random one.

4 RESULTS

	SC & ***AR***	$\boldsymbol{R}_{SC}$ & $\boldsymbol{R}_{AR}$	$\boldsymbol{C}_{AR}$ & $\boldsymbol{C}_{SC}$
DeltaCon 3.1	-	0.032	0.108
NetSimile 3.2 modified Euclidean	0.008	0.013	0.067
NetSimile 3.2 modified Cosine	0.761	0.777	0.948
NetSimile 3.2 modified Canberra	0.067	0.063	0.141
NetSimile 3.2 original Canberra	0.042	0.041	0.080
Clustering 3.3	-	0.062	0.281

Table 1: *Similarity scores between the three main pair of graphs*

4.1 DeltaCon results

As you can see from the Table 1, the DeltaCon method does not find great similarity between any pair of graphs. That means, according to DeltaCon's way of calculating the similarity score, that the pairwise node affinities in the first graph are not similar to the ones in the second graph. For the pair of graphs ***SC*** & ***AR*** we can't obtain a similarity score with DeltaCon because they don't have the same set of nodes.

We can do an interesting consideration on the DeltaCon's ε parameter: by changing it we can obtain very different scores. For example, with $\varepsilon = 0.00001$ we have $sim(\boldsymbol{R}_{SC}, \boldsymbol{R}_{AR}) = 0.604$ and $sim(\boldsymbol{C}_{SC}, \boldsymbol{C}_{AR}) = 0.830$ which makes the graphs look very similar. This is a trivial result since with a smaller ε we are reducing the importance of the matrices D and A (according to the DeltaCon's algorithm 1) but it makes us wonder what the ideal ε value might be. Maybe it doesn't exist since it depends on how much we want to consider the neighbors influence; however we preferred to keep as "ideal" the parameter provided by the DeltaCon's creators, that is $\varepsilon = 1/(1 + max_i(d_{ii}))$.

4.2 NetSimile results

So here we introduced the slight modification we talked about in the end of 3.2: for each node we removed the three features related to the egonet, as well as the two hop away average neighbors and we added the two more common centrality measures, i.e. betweeness and closeness centralities; results are shown in Table 1. Different features, such as those suggested by the authors were also considered but led to even lower similarities. We tried different distance metrics, such as euclidean distance and cosine distance: the latter is obviously not appropriate, and of course it led to very high similarities; the former, instead, gave low results. In order to understand better whether these similarities are actually good or not, a comparison between graphs generated at random with the same number of nodes should be performed. However since we obtained similar results between different methods, we decided not to proceed further. We give our final thoughts in 5.

4.3 Clustering results

The value reported in Table 1 relating to the Adjusted Rand score for the similarity of the clusterings on the graphs $\boldsymbol{R}_{SC}$ and $\boldsymbol{R}_{AR}$ appears to be very close to 0: this result is an indication that there is almost no meaningful resemblance between the two partitions, since it would be about the same ARI obtainable when comparing one of the graphs' clustering with a random partition of the

nodes. A more significant similarity has been observed studying the clustering on the countries graphs, meaning that only from the national level a slight likeness on the forming of communities is noticeable.
A note follows, regarding the ARI values written on the table: due to the heuristic nature of the Louvain algorithm, if we run its execution multiple times we get slightly different results; however the various ARI values returned from the comparison can be considered equivalent if rounded to the first decimal place. Thus what is shown in Table 1 are just samples of the obtainable results, since their meaning for our considerations doesn't change.

5 CONCLUSIONS

The achieved results show us that there is almost no significant resemblance between the analyzed networks. This goes against our expectations: as stated in our introduction, we believed to be able to observe some interesting analogies on how each city was linked to the others in the two graphs. The outcome of this investigation can be explained with the fact that generally there is no causal relationship between the twinning-towns partnerships and the air traffic that connects sister cities. However, a more promising result showed up when studying the comparison of the lower-granularity versions of our networks (the ones which consider countries as vertices): the analysis of the phenomenon among nations revealed a slightly higher similarity of the graphs, meaning that our purpose of capturing the previously inexpressible relations (due to the condition of the air-traffic dataset not always being capable of representing true travel routes between cities not directly connected by any airline) was somewhat successful, since it did not contradict our expectations.
The comparison results discussed above have been consolidated by using three different methods for measuring graphs' similarity, whose returned values proved themselves coherent with each other.
As a future development of this study, one possible way involves the adoption of a dataset that can express people's migratory routes more precisely and completely than the airline-routes network we used.

REFERENCES

[1] Ursino, A., Mariotto, M. & Marangoni, F. Comparison between the sister cities and the airline routes graphs (2021). URL https://github.com/albertoursino/GraphsComparison.git.

[2] Woebkenberg, T. Airports, airlines, and routes (2019). URL https://data.world/tylerudite/airports-airlines-and-routes.

[3] Koutra, D., Vogelstein, J. T. & Faloutsos, C. DELTACON: A principled massive-graph similarity function. *CoRR* **abs/1304.4657** (2013). URL http://arxiv.org/abs/1304.4657. 1304.4657.

[4] Berlingerio, M., Koutra, D., Eliassi-Rad, T. & Faloutsos, C. Netsimile: A scalable approach to size-independent network similarity. *CoRR* **abs/1209.2684** (2012). URL http://arxiv.org/abs/1209.2684. 1209.2684.

[5] Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. Fast unfolding of communities in large networks. *J. statistical mechanics: theory experiment* **2008**, P10008 (2008).

[6] Hubert, L. & Arabie, P. Comparing partitions. *J. classification* **2**, 193–218 (1985).

[7] Koutra, D., Parikh, A., Ramdas, A. & Xiang, J. Algorithms for graph similarity and subgraph matching. In *Proc. Ecol. Inference Conf*, vol. 17 (Citeseer, 2011).

[8] Tantardini, M., Ieva, F., Tajoli, L. & Piccardi, C. Comparing methods for comparing networks. *Sci. reports* **9**, 1–19 (2019).

[9] Wills, P. & Meyer, F. G. Metrics for graph comparison: a practitioner's guide. *Plos one* **15**, e0228728 (2020). URL https://arxiv.org/abs/1904.07414.

6 CONTRIBUTION OF EACH MEMBER

Except for ”Methods” and ”Results” sections, this project report has been written collectively.

1. **Introduction**: Alberto Ursino, Marco Mariotto and Fabio Marangoni;
2. **Data**: Alberto Ursino, Marco Mariotto and Fabio Marangoni;
3. **Methods**: DeltaCon (Alberto Ursino), NetSimile (Marco Mariotto), Clustering (Fabio Marangoni);
4. **Results**: DeltaCon results (Alberto Ursino), NetSimile results (Marco Mariotto), Clustering results (Fabio Marangoni);
5. **Conclusions**: Alberto Ursino, Marco Mariotto and Fabio Marangoni;
$*$. **Code**: Alberto Ursino, Marco Mariotto and Fabio Marangoni.