A Gravitational Approach for Ranking Autonomous Systems in Large Autonomous System Networks

Jotte Sonneveld
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
j.j.sonneveld@student.utwente.nl

ABSTRACT

An Autonomous System (AS) is a group of host IP addresses, known as routing prefixes, that share common routing policies. Multiple ASes interact with each other through a massive network of links which reflect customer-provider and peer-to-peer partnerships among network operators. Such partnerships form an Autonomous System Network (ASN) that connects millions of hosts around the world and gives shape to the topology of the Internet. The ranking of ASes in an ASN allows researchers to acquire important insights into the complex structure of the Internet. ASes are ranked by the Center for Applied Internet Data Analysis (CAIDA) by their customer cone size, which is the number of direct and indirect customers. While the customer cone size and other similar metrics represent an intuitive way to measure the rank of ASes, they suffer from low monotonicity, making it difficult to discriminate among ASes with the same measurements. In this research we propose a new approach for ranking ASes within the ASN, measuring customer cones by exploiting a gravitational approach used in Network Theory to quantify the influence of nodes in a complex graph and their capacity to become good spreaders. We will also propose an efficient algorithm to measure gravitational metrics by exploiting the fact that customer cones form large Directed Acyclic Graphs, in order to handle large ASNs with dozens of thousands of nodes and links.

Keywords

Autonomous System Network, Autonomous System Ranking, Customer Cone, Gravitational Approach, Monotonicity index, Correlation Coefficient

1. INTRODUCTION

The backbone of the internet consists of tens of thousands smaller networks called Autonomous Systems (AS). An AS is a network of connected devices all sharing a dedicated IP prefix, the starting bits of an IP address. They are owned by a network operator who provides them with a single routing policy [9]. Owners of such a system are e.g. Internet Service Providers, universities or large companies. Every AS is connected to one or more other ASes in the Autonomous System Network (ASN), and communicate with each other via the Border Gateway Protocol (BGP). The ASN is designed in such a way that every AS can communicate with every other AS via one or more links in the network.

There are two main types of links between ASes: customer to provider (c2p, or p2c when travelling in reverse) and peer to peer (p2p). The c2p principle is simple: the *provider* provides access through its network to the *customer*, for which the customer pays the provider. A p2p connection between AS A and AS B means that the traffic from A and their customers may flow through B at no cost to A, and vice versa. At the core of the internet there lies a *clique* of a few dozen ASes which have mostly a p2p connection with each other. Their revenue model comes not from the other clique members, but from their customers, and indirectly their customers' customers and so on.

As of April 2020, over 95.000 ASes have been registered at the Internet Assigned Numbers Authority [17][10]. The vast size of this network makes it hard to comprehend or analyse, for both research and economical purposes. One way to ease this is to perform Autonomous System Ranking. This produces a list of all ASes, in a particular order. The most useful ranking criterion is ranking by 'importance'. This, however, is prone to subjectivity, and to produce a ranking an objective criterion must be used. A logical method of ranking is how *central* a certain AS is located. The more central an AS, the more others will depend on it. This could be measured by how many shortest paths travel through this node, which is called *betweenness centrality*. Another commonly used criterion is ranking by *customer cone* [3], used by for example the Center for Applied Internet Data Analytics (CAIDA)[5]. In Figure 1 the customer cone for each AS is shown. The customer cone is good at capturing the amount of ASes which directly or indirectly pay a certain AS and therefore form its cash flow. However, when applying this to the ASN a lot of nodes will have the same customer cone size, and therefore the same rank, whilst they not necessarily have the same importance or centrality. We express this resolution

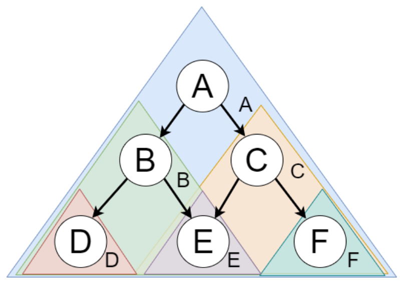

Figure 1. The customer cone for each AS

33rd *Twente Student Conference on IT* July 3rd, 2020, Enschede, The Netherlands.

of a ranking via the *monotonicity index*, which is further elaborated upon in section 3.2.

Ma et al. [15] proposed a new method of ranking vertices within a graph by applying the gravitational formula. We will use this method to rank the ASes within the ASN, analyse this ranking and compare it to the customer cone metric. The main question which is answered in this paper is the following:

Main Research Question: How does the gravitational approach to ranking autonomous systems in large autonomous system networks compare to the conventional customer cone approach?

In order to guide this research and reason a conclusion, these two questions will first be answered:

Research Question 1: Is it possible to develop an efficient algorithm for the gravitational-based AS ranking metrics?

Research Question 2: How does the gravitational ranking approach relate to the customer cone ranking in terms of monotonicity, correlation and ordering of ASes?

We will first show the design of a gravitational ranking algorithm of linear complexity. This algorithm is applied to the ASN using different centrality metrics to produce several rankings. These rankings are compared to each other and the conventional customer cone approach using on the monotonicity index, correlation coefficients and general ordering. We present the results of this comparison, discuss its impact on the state of the art and conclude the answer to our research question.

2. RELATED WORKS

There have been multiple works which have introduced novel metrics to rank ASes within the ASN, or rank nodes within a network in general.

Kitsak et al. [12] applied the k-shell decomposition method to identify the most efficient spreaders within a graph. K-shell decomposition iteratively removes edges from nodes with the lowest degree to place them in a certain 'shell'. Comparing it to the susceptible-infectious-recovered model, a model to simulate the spearing of disease through a network of people, they demonstrated that the k-shell decomposition is a good method to locate the core of a network. This also showed how the most influential spreaders are not necessarily the ones with the highest number of edges or most central location within the graph. As the ASN is a directed acyclic graph, the concept of 'centrality' differs from its non-directed or cyclic counterparts. The k-shell decomposition alone would most likely have a bad resolution on the ranking. Bae et al. [4] recognised the bad resolution of the k-shell when applying it to complex networks such as the Barabási-Albert (BA) type[2]. The BA type networks are scale-free, meaning they have a low number of nodes with a high degree and a high numbers of nodes with a low degree, which is similar to the ASN. Bae et al. proposed identifying influential spreaders through the *coreness centrality* metric, which constructs a node's score by the k-shell indices of its neighbours. This showed a better performance on BA type networks compared to other centrality metrics. Wang et al. [20] recognised the need for efficient rankings algorithms, as other centrality metrics were usually of more than linear complexity. They proposed a ranking method based on node position and neighbourhood. This uses the k-shell decomposition iterations and neighbourhood attributes to measure the influence of a node. This resulted in an increase in monotonicity compared to other centrality metrics, whilst having linear complexity.

The monotonicity of several centrality metrics to rank ASes within the ASN were compared by Tozal et al. [19]. They show how different metrics lead to different ranking orders. They find a high level of agreement between customer cone size and outdegree, a moderate level of agreement between outdegree and betweenness, and a moderate level of agreement betweenness and customer cone size. This demonstrates how different centrality metrics prioritise some ASes above others, which is why as part of research we want to measure the correlation between the different gravitational metrics.

Our research uses the ranking approach described by Ma et al. [15], which applies the gravitational formula to rank nodes within a network. In this approach the k-shell of a node is its mass, and the shortest paths between two nodes is the distance. This leads to an increase in monotonicity compared to other centrality metrics. They only applied the k-shell as node mass to apply the formula, however in our research we will apply multiple centrality-based metrics, compare them to each other and to the customer cone approach.

3. METHODOLOGY

3.1 Gravitational algorithm

For an algorithm to be considered linear, $O(n)$, the amount of operations it performs has to be a constant c times the number of vertices in a graph. This means that when the graph doubles in size, the calculation may not take more than two times its previous computation time. This is especially important when applying algorithms to the ASN as this network is ever-increasing.

The gravitational approach ranks vertices in a graph according to Equation 1 [15]:

$$G(i) = \sum_{j \in \Psi_i} \frac{ks(i)ks(j)}{d_{ij}^2} \quad (1)$$

where d_{ij} is the shortest path distance between node i and node j. The ks value, the mass of a node, can be constructed in different ways. In this paper we will use the several different centrality indices as mass metric, all of the paths take into account the directedness of the edges, shown in Table 1. The used `igraph` methods can be found in the corresponding parts of the documentation manual [1], and the algorithm is presented later in the paper.

Using the gravitational metric, an extended gravity index can be calculated by adding all the G-values of an AS's direct customers. This formula is represented by Equation 2 [15]:

$$G_{+}(i) = \sum_{j \in \Lambda_i} G(j) \quad (2)$$

where Λ in an ASN is the set of direct customers of i.

3.1.1 Topological sort

Traversing the vertices within a graph in a certain order allows for efficiency within algorithms. The ASN has a structure known as a DAG (Directed Acyclic Graph). The ASN graphs used is this research are constructed via the method described by Luckie et al. [14]. In this research, BGP routing policies between ASes are used to infer relationships. It does not make economical sense for an AS to export provider routes to providers and peers, as this

Mass metric	Definition	Implementation
Out-degree	Amount of links directed out of a node, in the ASN: direct customers	igraph `degree( mode= "OUT")`
Customer Cone size	The total amount of nodes which can be reached from one node, including itself	Algorithm 2
Closeness	The average length of the shortest path between a node and all other nodes	igraph `closeness( mode= "OUT")`
Betweenness	The fraction of shortest paths which flow through a node	igraph `betweenness()`
K-shell	Applying K-shell decomposition [12], taking into account the combination of in- and out-degree	igraph `shell_index( mode="ALL")`

Table 1. The centrality metrics used in the gravitational ranking approach

would lead to free transit. Therefore, in their inference method the assumption is made that the ASN is valley-free, which means there are no cycles possible following the directed edges. This may not fully represent reality, but the inferred AS relationship graph is therefore certain to contain no loops. Because there are no loops within a DAG, a topological order can be created. Traversing vertices within this order means that when visiting a vertex, it is certain that all vertices which have a directed edge towards this vertex have already been visited. This allows for top-down and bottom-up walking through graphs, which in Figure 1 would lead to traversing in respectively alphabetical and reversed alphabetical order. A topological order is not necessarily unique, but always adheres to the above described principle. The sorting of vertices in a DAG in topological order can be done in linear time using Algorithm 1 [8]:

Algorithm 1: Topological sort

```
input  : A list of vertices
output: The list of vertices in topological
        ordering
Create linked list toposorted;
/* A node is considered 'explored' in DFS
   once all its successors have been
   explored                                  */
while there are still unexplored nodes do
    Take arbitrary unexplored node v;
    Call DFS(G) on v;
    Insert every explored node onto the front of
    toposorted;
return toposorted
```

3.1.2 Attributing mass

In order to calculate the gravitational value of every vertex, they have to be attributed a mass. The only mass metric for which we wrote a separate algorithm is the customer cone size. This allows us to elaborate the first characteristics of the actual gravitational ranking algorithm.

For this algorithm to be linear every vertex may only be visited a constant amount of times. To achieve this, we make use of an AncSet (Ancestor Set). The ancestor set of a vertex is the opposite of a customer cone. Instead of all customers, customers of customers, etc. we look at all the providers, providers of providers and so on. In graph terminology those are called *ancestors*. If v has an ancestor a, then v is part of a's customer cone. This is relevant as it therefore contributes to a's customer cone size/score. When visiting node v, because of the topological order all its ancestors have previously been visited. If therefore v's direct parents p have kept track of all their own ancestors, v can derive their ancestors from its parents' AncSets. If whilst traversing the graph in topological order every node keeps track of their AncSet, at the end every node will have a set of all of their ancestors. To compute the customer cone size, every node simply loops over all their ancestors and increments their customer cone size by one. The pseudocode for this is presented in Algorithm 2. `CC_sizes[v]` is the customer cone size of vertex v, and `AncSet[v]` is the ancestorset for vertex v which contains no duplicates. `AncSet[a]` $+=$ `AncSet[b]` means that `AncSet[a]` becomes the union of both sets. The operator `set(v)` creates a new set containing item v.

Algorithm 2: Calculation of customer cone size

```
input  : The directed acyclic graph g
output: The list of all Customer Cone sizes
/* Initialise CC_sizes array with as many
   0's as there are vertices in the graph
   */
CC_sizes = [0]* amt_of_vertices ;
/* Initialise ancestorset for each vertex,
   including self-entry                      */
foreach v ∈ g do
    AncSet[v]= set(v)
Compute topological ordering;
/* Loop over all vertices in topological
   order                                     */
foreach v ∈ g in topological order do
    /* Inherit the parents' ancestorsets    */
    foreach parent p of v do
        AncSet[v]+ = AncSet [p]
    /* Increment the CC size of each
       ancestor by 1                         */
    foreach a ∈ AncSet[v] do
        CC_sizes[a]+= 1
return CC_sizes
```

3.1.3 Gravitational approach ranking

Once a useful visiting order has been created and all vertices have a mass, the actual G and G+values are computed. This is done in a similar fashion to the customer cone size calculation, however solely saving the set of ancestors is not enough to compute G-values. The gravity formula uses the distance between two nodes as part of the equation. Therefore, next to the ancestor itself, a node needs to keep track of the distance to that ancestor. For this we use an Ancestor Distance HashTable (ADHT). We define a node's ADHT as a hashtable containing ancestor nodes as keys, and the shortest link distance to the respective ancestor as values. When inheriting an ADHT from a parent, all distances are incremented by 1. There are cases where an AS can reach an ancestor via different paths, possibly with different link lengths. For this case we only use the shortest path length to compute the G-value of the ancestor. Therefore, when inheriting an

ADHT from a parent node, we first check for duplicates. When an ancestor is in both sets, only the version with the shortest link length to is saved to the AS's ADHT. For example, the ADHT of node E in Figure 1 is constructed in two iterations. In this network B's ADHT is `{B:0,A:1}` and C's ADHT is `{C:0,A:1}`. E first inherits B's ADHT which makes it `{B:1,A:2}`. Second E inherits C's ADHT, skips the duplicate A entry which does not provide a shorter link distance, which results in E's final ADHT of `{B:1,A:2,C:1}`.
Once these steps have been completed, a node can calculate their part of the G-value for all of their ancestors. Computing all G_+ values requires one extra loop through the entire graph where they sum the G-values of their direct children/customers. The full algorithm is stated in Algorithm 3. `ADHT[v]`, `mass[v]`, `G_val[v]` and `G+_val[v]` are the corresponding values of vertex v. `ADHT[v].ancestors` is the set of ancestors of v and `ADHT[v][a]` is the shortest distance between v and its ancestor a. The inheritance process is expressed via the `merge(ADHT[v],ADHT[p])` function. This function returns an ADHT which is the union of v's and p's ADHT, with p's ancestor distances incremented by 1. In the case where both ADHT's contain the same ancestor a, this ancestor is included only once, with the distance `min(ADHT[v][a],ADHT[p][a]+1)`, where `min()` returns the lowest value of the two arguments.

Algorithm 3: Gravitational ranking

```
input  : The directed acyclic graph g
output: The lists of G and G+ values for every
        node in the graph
/* Initialize the ADHT for every node v in
   graph g, including self-entry          */
foreach v ∈ g do
    Initialize ADHT[v];
    Add v{v:0} to ADHT[v]
Compute topological ordering;
/* Initialise G_val array with as many 0's
   as there are vertices in the graph     */
G_val = [0]* amt_of_vertices ;
/* Fill all ADHT tables and compute
   G-values                               */
foreach v ∈ g in topological order do
    foreach parent p of v do
        ADHT[v]= merge(ADHT[v],ADHT[p])
    foreach ancestor a ∈ ADHT[v].ancestors do
        if a! = v then
            G_val[a]+ =
            (mass[a]* mass[v]) / ADHT[v][a]^2
/* Compute G+ values                      */
G+_val = [0]* amt_of_vertices ;
foreach node v ∈ g do
    foreach child c from v do
        G+_val[v]+= G_val[c]
return G_val, G+_val
```

3.2 Monotonicity

The first way of comparing the outcome of these new ranking metrics is via their monotonicity index [15]. This formula measures the resolution of a ranking: the less shared ranks there are amongst vertices, the higher the monotonicity. As the goal for a ranking metric is to show the difference between vertices, a higher monotonicity is desired. The formula for this is represented by Equation 3:

$$M(X) = [1 - \frac{\sum_{c \in V} N_c(N_c - 1)}{N(N-1)}]^2 \tag{3}$$

where N_c is the number of nodes with the same rank and N is the size of the entire network. Calculating the monotonicity value from a ranking is done following Algorithm 4. Within the ranking *ranking*, the content of each rank is expressed as the list *ranklist*, containing one or more vertices.

Algorithm 4: Monotonicity computation

```
input  : A ranking list with a list of nodes at
         every rank
output: The monotonicity value
total_node_amt = 0 ;
sigma_val = 0 ;
/* Loop over every rank and add its values
   to the sigma equation and total amount
   of nodes                               */
foreach ranklist ∈ ranking do
    rank_node_amt += length(ranklist);
    sigma_val += rank_node_amt *
    (rank_node_amt - 1);
    total_node_amt += rank_node_amt;
monotonicity =
(1 - sigma_val / (total_node_amt*(total_node_amt-1)))^2;
return monotonicity
```

3.3 Correlation

The next gravitational metrics comparison is performed by computing all pairwise correlations. Every ranking metric produces a list of scores, corresponding to the ASes. The lists of every two pairs of ranking metrics will be compared using different correlation coefficients. Such coefficients are Pearson[16], Spearman[13] and Kendall[11], which compute a value between 0 and 1 on every ranking metric pair. The higher the correlation coefficient, the more correlated two ranking metrics are.
Where Kendall and Spearman do not have any requirements for the dataset, the Pearson correlation coefficient assumes that the values are normally distributed. Therefore to decide how relevant the Pearson correlation coefficient is, the Shapiro-Wilk test[18] will be applied to the rankings metrics. This is a test on normality which indicates how normally distributed the scores of the ASes are.

3.4 AS ordering

Lastly, we want to find out which AS characteristics are ranked highest by the metrics. The correlation research will show which ranking metrics are heavily correlated, and which are not. We will take several rankings which are barely correlated and compare their top 10. We will look into the differences and what leads to an AS being ranked higher in one ranking metric compared to the others.

4. RESULTS

4.1 Gravitational algorithm

The result of this part of the research is the algorithm demonstrated in Section 3.1. This is a gravitational ranking algorithm of linear complexity, which was the target. They key part of the algorithm was the use an ADHT for every node. This made it possible to construct all G-values in a single traversal through the graph in topological order. The different mass metrics were used in the gravitational ranking of five different AS relationship datasets from CAIDA[7], all one year apart.

Ranking metric: Dataset	Out-degree G	Out-degree G+	CC size G	CC size G+	Close-ness G	Close-ness G+	Between-ness G	Between-ness G+	K-shell G	K-shell G+	Customer Cone
05-2020	0.0123	0.0033	0.0780	0.0123	0.0780	0.0123	0.0115	0.0033	0.0795	0.0123	0.0778
05-2019	0.0121	0.0032	0.0778	0.0121	0.0779	0.0121	0.0113	0.0032	0.0793	0.0121	0.0777
05-2018	0.0120	0.0032	0.0774	0.0120	0.0774	0.0120	0.0111	0.0031	0.0789	0.0120	0.0772
05-2017	0.0113	0.0027	0.0751	0.0113	0.0751	0.0113	0.0104	0.0026	0.0764	0.0113	0.0749
05-2016	0.0115	0.0028	0.0757	0.0114	0.0756	0.0114	0.0107	0.0028	0.0771	0.0115	0.0755

Table 2. The monotonicity values considering all ASes

Ranking metric: Dataset	Out-degree G	Out-degree G+	CC size G	CC size G+	Close-ness G	Close-ness G+	Between-ness G	Between-ness G+	K-shell G	K-shell G+	Customer Cone
05-2020	0.3672	0.1200	0.7342	0.3608	0.7344	0.3608	0.3489	0.1182	0.9377	0.3673	0.7122
05-2019	0.3627	0.1164	0.7340	0.3555	0.7344	0.3557	0.3444	0.1150	0.9380	0.3629	0.7115
05-2018	0.3610	0.1148	0.7327	0.3536	0.7329	0.3538	0.3408	0.1131	0.9367	0.3613	0.7093
05-2017	0.3548	0.1025	0.7315	0.3473	0.7317	0.3476	0.3353	0.1007	0.9346	0.3547	0.7091
05-2016	0.3555	0.1044	0.7326	0.3484	0.7329	0.3484	0.3384	0.1037	0.9362	0.3556	0.7102

Table 3. The monotonicity values without taking into account the leaf ASes

4.2 Monotonicity comparison

The results of the initial monotonicity comparison are presented in Table 2. These values are very low due to the way the ASN is structured. At the outer ends of the ASN, there are a lot of ASes which do not have any customers. Imagining the ASN as a tree structure, these ASes would be the leaves at the bottom, illustrated in Figure 2. We define a *Leaf AS* as an AS which has an outdegree of 0. As these ASes have no customers, through the gravitational formula they score a 0 by default, therefore within the gravitational ranking approach about 85% of all ASes have a score of 0. The score of the providers of leaf ASes are reliant on the metric used to assign mass to an AS. When the mass metric is *outdegree*, this will be 0 for all leaf ASes. In the gravitational approach, the G-value of an AS is constructed through a multiplication with the mass of its customers as can be seen in Equation 1. Since this mass for leaf ASes is 0, any AS which only has leaf ASes as customers also has a G-value of 0. Therefore, mass metrics which assign the value of 0 to leaf ASes will by default classify leaf ASes and the providers of solely leaf ASes as the lowest rank, heavily influencing the monotonicity.

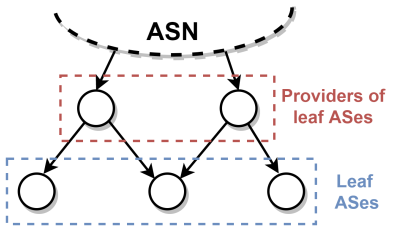

Figure 2. The outskirts of the ASN

The G+ value is constructed by adding the G-values of its direct customers, therefore with the *outdegree* mass metric even the providers of the providers of the leaf ASes will get a score of 0.
What can be argued however, is that the 85% of leaf ASes are not the ones that we are interested in. They may have p2p connections, but serve no purpose in the economical analysis as no customer pays them. Therefore, we decided to compute the monotonicity index again, but after removing all ASes with an outdegree of 0. The results of this are presented in Table 3.

4.3 Ranking metric correlations

For this part of the research, only the CAIDA AS relationship dataset of 05-2020 was used. As the monotonicity research demonstrated, there is a significant difference between rankings when considering the entire ASN or when leaving out the leaf ASes. Therefore, we applied the correlation metrics to both the 'score list' (G-value and G+ values) of the entire ASN, and the score list after filtering out the leaf ASes.
Firstly, the Shapiro-Wilk values were computed for every metric, the results of which are provided in Table 4. As this was mainly to indicate how accurate the Pearson correlation coefficient would be and not a test on normality, no null-hypothesis alpha value was constructed beforehand. From the Shapiro-Wilk values however, we see that the lists of G, G+ and customer cone values do not show much resemblance of a normal distribution, with the possible exception of the customer cone with leaf ASes removed. Therefore, we decided not to use the Pearson correlation coefficient for this research.

We computed the clustermaps of the Spearman and Kendall correlation metric, applied to the entire ASN ranking as well as the ranking with leaf ASes removed. Figure 3 shows the correlation clustermap with the highest resolution. The other clustermaps can be found in Appendix A. Both correlation coefficients can show a better resolution of ranking metric correlation when applied to the dataset with leaf ASes removed. This is because the 85% of 0-scores is removed from the ranking lists and therefore their differences are highlighted more. In all four figures there are three highly correlated clusters of ranking metrics visible:

- Outdegree G+ and Betweenness G+
- CC size G, Closeness G and Customer Cone
  - K-shell G is somewhat correlated, but can also be seen as a different group
- Outdegree G, CC size G+, Closeness G+, K-shell G+
  - Betweenness G is somewhat correlated, but can also be seen as a different group

Ranking metric:	Out-degree G	Out-degree G+	CC size G	CC size G+	Close-ness G	Close-ness G+	Between-ness G	Between-ness G+	K-shell G	K-shell G+	Customer Cone
Entire ASN	0.0015	0.0029	0.0046	0.0041	0.0041	0.0049	0.0046	0.0063	0.0105	0.0051	0.0240
Leaf ASes removed	0.0075	0.0146	0.0224	0.0201	0.0191	0.0233	0.0222	0.0299	0.0467	0.0242	0.1041

Table 4. The Shapiro-Wilk test values of the score lists produced by the different ranking metrics, applied the 05-2020 AS relationship dataset

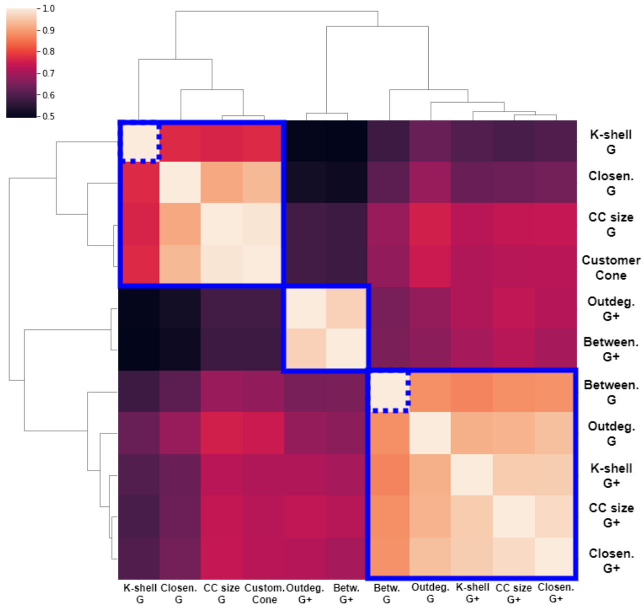

Figure 3. Clustermap of Kendall correlation coefficient, applied to the score lists of various ASN ranking metrics with leaf ASes removed, using the CAIDA AS relationship dataset of 05-2020 [7] as ASN. Lighter colors correspond to higher correlation coefficients. The blue contours represents the different correlation clusters. The dotted line highlights the metric which only has a moderate correlation with the other metrics within that cluster.

For each pairwise comparison, besides the correlation coefficient the two-sided p-value is also computed. For all of the correlation coefficients this value was near or equal to 0.

4.4 Top ranked AS characteristics

We manually looked up the information corresponding to the selected ASes using the CAIDA ASRank tool [6]. Table 5 displays the organisation and AS id of the top 10s of five minimally correlated ranking metrics.

5. DISCUSSION

5.1 Monotonicity comparison

As illustrated by Table 2 and 3, the monotonicity drastically increases when leaving out the 'irrelevant' part of the ranking, namely the leaf ASes. Both *outdegree* and *betweenness* assign the mass of 0 to the leaf ASes, which results in their direct providers scoring 0 through the multiplication in the gravitational formula. This effect is amplified when computing the G+ value, which adds the 0 scores of the leaf-providers, resulting in the three outer layers of ASes all being ranked at the bottom. This heavily decreases the monotonicity of such rankings, as this includes roughly 97% of all ASes.

There are two mass metrics which have roughly the same monotonicity as the conventional *customer cone* approach: *CC size G* and *closeness G*. All three of them do not bring much resolution to the score of the leaf AS's providers, however the gravitational metrics seem to be better at showing differences between the ASes which are not in the outer two shells of the ASN. Their G+ counterparts however, will still rank the outer two shells at the same bottom rank, comparable to the G-values of the *outdegree* and *betweenness* metrics.

Finally, there is the K-shell G-value which ranks highest in monotonicity of all metrics. As shown in the Methodology section, for the K-shell decomposition metric we used the combination of indegree and outdegree, as when only using either of them there was no distinction possible between any of the ASes: all ended up in the same shell. The main difference between this mass metric and CC size, is that the K-shell is able to bring more resolution to the scoring of the providers of leaf ASes. Where the other metrics only take into account the outgoing edges of an AS, this K-shell metric can show the difference between an AS which is heavily connected through both its providers and customers, and an AS which has the same amount of customers but less providers. The disadvantage of taking into account all edges is that it may attribute a higher mass to an AS with many providers than the provider of many ASes. This is therefore less desirable when using a ranking for economical importance, but more specific research is required to investigate how much the above illustrated example actually occurs in the real ASN.

5.2 Ranking metric correlations

The three clusters in Figure 3 show that through the gravitational approach itself, rankings with multiple properties can be computed. The *K-shell G* seems to be least correlated to any other ranking, which makes sense from our monotonicity observation.

Closeness G and *CC size G* are highly correlated with the conventional customer cone, whilst both having a higher monotonicity. This means they produce a ranking similar to the customer cone one, whilst having a higher resolution, which can be seen as an improvement.

The correlation between *Outdegree G+* and *Betweenness G+* could show that when an AS has many direct customers, they will also provide more shortest links between ASes.

Finally, there are mixed G and G+ ranking metrics correlated: *Betweenness G*, *Outdegree G*, *K-shell G+*, *CC size G+* and *Closeness G+*. A possible explanation for this is that they all adhere to the principles of the Outdegree mass metric discussed in section 4.2. For the G-value metrics among these, all leaf ASes are assigned the mass of 0, which leads to their providers getting a G-value of 0. All G+ metrics also follow this pattern as leaf ASes score a

Ranking metric: Rank	Outdegree G+	Customer Cone	K-shell G	Closeness G+	Betweenness G
1	Level 3 Parent, LLC (3356)	Level 3 Parent, LLC (3356)	Level 3 Parent, LLC (3356)	Telia Company AB (1299)	Telemar Norte Leste S.A. (7738)
2	Telia Company AB (1299)	Telia Company AB (1299)	Cogent Communications (174)	Level 3 Parent, LLC (3356)	ITS Telecomunicacoes (28186)
3	PCCW Global, Inc. (3491)	Cogent Communications (174)	Telia Company AB (1299)	Cogent Communications (174)	Psychz Networks (40676)
4	GTT Communications Inc. (3257)	GTT Communications Inc. (3257)	GTT Communications Inc. (3257)	GTT Communications Inc. (3257)	UPX Technologies (52863)
5	Sprint (1239)	NTT America, Inc. (2914)	NTT America, Inc. (2914)	Tata Communications (America), Inc. (6453)	Vocus Communications (4826)
6	MCI Communications Services, Inc. d/b/a Verizon Business (701)	Tata Communications (America), Inc.(6453)	Tata Communications (America), Inc. (6453)	NTT America, Inc.(2914)	China Mobile International Limited (58453)
7	Cogent Communications (174)	MCI Communications Services, Inc. d/b/a Verizon Business (701)	PCCW Global, Inc. (3491)	PCCW Global, Inc. (3491)	China Telecom Next Generation Carrier Network (4809)
8	Tata Communications (America), Inc. (6453)	PCCW Global, Inc. (3491)	Hurricane Electric LLC (6939)	Telecom Italia Sparkle S.p.A. (6762)	Internexa Brasil Operadora de Telecomunicacoes (262589
9	Telecom Italia Sparkle S.p.A. (6762)	Sprint (1239)	Telecom Italia Sparkle S.p.A. (6762)	Sprint (1239)	M247 Ltd (9009)
10	NTT America, Inc. (2914)	Hurricane Electric LLC (6939)	Level 3 Parent, LLC (3549)	Hurricane Electric LLC (6939)	Reliance Globalcom Limited (15412)

Table 5. The company behind every AS within the top 10 of five different ranking metrics, followed by the AS id

G-value of 0, and therefore their providers score 0 in the G+ metrics. The difference between the Betweenness G and Outdegree G metric versus their G+ counterparts, is that in the G+ variant even the providers of the providers of the leaf ASes get a score of 0 through this reasoning.

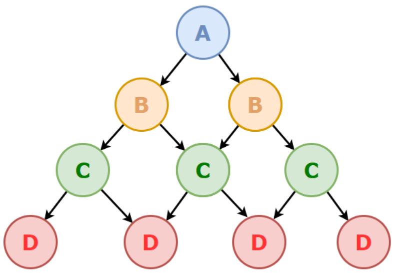

Figure 4. An example ASN

5.3 Top ranked AS characteristics

The top 10 ASes are quite alike for four of the five ranking metrics, only the *Betweenness G* stands out. This can be explained by the characteristics of this metric and the way the ASN is structured. See Figure 4 for an example ASN.

The betweenness value of a node is computed by the shortest paths which flow through them, whilst they are neither the source nor destination-node. This means that both A and D will have a betweenness value and therefore weight of 0. As the gravitational value for an AS is constructed via a multiplication with their mass, this will result in a score of 0 by default for nodes A and D. In the ASN this means that ASes without providers as well as ASes without customers are ranked at the bottom. Node C will also have a G-value of zero, since this is constructed by D's mass of 0. In the end, only B will have a non-zero G-value. This demonstrates another reason for the low monotonicity value of *Betweenness G*, and suggests that this metric ranks the ASes customer to provider-less clique ASes highest.

6. CONCLUSION AND FUTURE WORK

6.1 Conclusion

Gravitational ranking algorithm

The gravitational approach allows for a novel method of ranking ASes within the ASN, which is possible to compute via a linear complexity algorithm due to the DAG type structure of the network.

Monotonicity

Applying the gravitational ranking approach to the ASN allows for metrics with a higher monotonicity compared to the conventional Customer Cone size approach. The *K-shell G-value* has the highest overall monotonicity, whilst the *Closeness G-value* has the highest monotonicity of metrics which only uses an AS's customer cone.

Correlation

By applying different metrics to the gravitational formula, there are different gradations of correlation possible. We have shown three different highly correlated clusters of gravitational ranking metrics.

Top ranked ASes
Whilst most gravitational metrics rank the AS with the largest customer cones highest, the *Betweenness G-value* metric ranks the ASes forming the bridge between the clique and the rest of the ASN highest.

Therefore, the gravitational approach allows for an efficient ranking metric which is highly correlated to the conventional customer cone approach, but with a higher monotonicity. In addition, the gravitational approach allows for multiple ranking metrics less correlated to the customer cone approach.

6.2 Future work

In this research we have demonstrated the efficiency of a linear complexity gravitational ranking algorithm. Besides this algorithm, there also exists a version with quadratic complexity. One of the differences between these two is that the quadratic algorithm is able to calculate the G+ values of nodes in parallel, contrary to the linear version where this is only possible in sequential order. The parallelism allows for running on multiple threads at the same time, which could lead to an even quicker computation time compared to the linear one. The future research we therefore suggest is comparing the computation times of a linear single-threaded gravitational ranking algorithm to its multi-threaded quadratic counterpart.

7. ACKNOWLEDGEMENTS

I would like to thank my supervisors Dr. Zangiabady and Dr. Garcia-Robledo for their guidance during this research. They have helped enormously in both the practical part such as developing the algorithm, as well as the mental part in keeping me motivated during the isolation.

8. REFERENCES

[1] python-igraph manual. https://igraph.org/python/doc/igraph.GraphBase-class.html (visited 19/06/2020).

[2] Albert, R., and Barabási, A.-L. Statistical mechanics of complex networks. *Rev. Mod. Phys. 74* (Jan 2002), 47–97.

[3] AS Rank, Customer Cone, 2020. https://asrank.caida.org/about#customer-cone.

[4] Bae, J., and Kim, S. Identifying and ranking influential spreaders in complex networks by neighborhood coreness. *Physica A 395* (2014), 549–559.

[5] CAIDA: Center for Applied Internet Data Analysis, 2020. https://www.caida.org/home/ (visited 20/06/2020).

[6] CAIDA AS Rank, 2020. https://asrank.caida.org/ (visited 20/06/2020).

[7] The CAIDA UCSD as-relationships - 20160501,20170501,20180501,20190501,20200501. http://data.caida.org/datasets/as-relationships/ (visited 01/06/2020).

[8] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. Introduction to Algorithms, Third Edition. Tech. rep.

[9] Hawkinson, J., and Bates, T. Guidelines for creation, selection, and registration of an Autonomous System (AS), 1996.

[10] Internet Assigned Numbers Authority (IANA), 2020. https://www.iana.org/ (visited 25/06/2020).

[11] Kendall, M. G. The treatment of ties in ranking problems. *Biometrika 33*, 3 (1945), 239–251.

[12] Kitsak, M., Gallos, L. K., Havlin, S., Liljeros, F., Muchnik, L., Stanley, H. E., and Makse, H. A. Identification of influential spreaders in complex networks. *Nature Physics* (2010).

[13] Kokoska, S., and Zwillinger, D. *CRC Standard Probability and Statistics Tables and Formulae, Student Edition*. 2000. Chapter 14.7.

[14] Luckie, M., Uc, C. ., Diego, S., Huffaker, B., Dhamdhere, A., Org, A., and Giotsas, V. AS Relationships, Customer Cones, and Validation.

[15] Ma, L.-L., Ma, C., Zhang, H.-F., and Wang, B.-H. Identifying influential spreaders in complex networks based on gravity formula. *Physica A 451* (2016), 205–212.

[16] Pearson, K. VII. Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London 58*, 347-352 (1895), 240–242.

[17] Regional Internet Registries Statistics - RIR Delegations - World - Autonomous System Number statistics - Sorted by number, 2020. https://www-public.imtbs-tsp.eu/~maigron/RIR_Stats/RIR_Delegations/World/ASN-ByNb.html (visited 06/05/2020).

[18] Shapiro, S. S., and Wilk, M. B. An Analysis of Variance Test for Normality (Complete Samples) An analysis of variance test for normality (complete samples)t. Tech. Rep. 3, 1965.

[19] Tozal, M. E. Autonomous system ranking by topological characteristics: A comparative study. Institute of Electrical and Electronics Engineers Inc.

[20] Wang, Z., Du, C., Fan, J., and Xing, Y. Ranking influential nodes in social networks based on node position and neighborhood. *Neurocomputing 260* (10 2017), 466–477.

APPENDIX

A. CORRELATION CLUSTERMAPS

Figures 5, 6 and 7 demonstrate the clustermaps corresponding to different correlation metrics or ASN components, using the CAIDA AS relationship dataset of 05-2020 [7] as ASN. Lighter colors imply a higher correlation coefficient. The blue contours represents the different correlation clusters. The dotted line highlights the metric which only has a moderate correlation with the other metrics within that cluster.

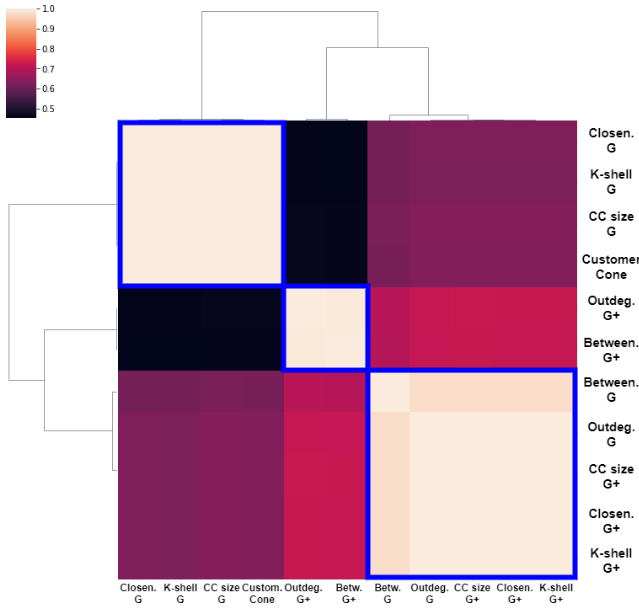

Figure 5. Clustermap of Spearman correlation coefficient, applied to the score lists of various ASN ranking metrics

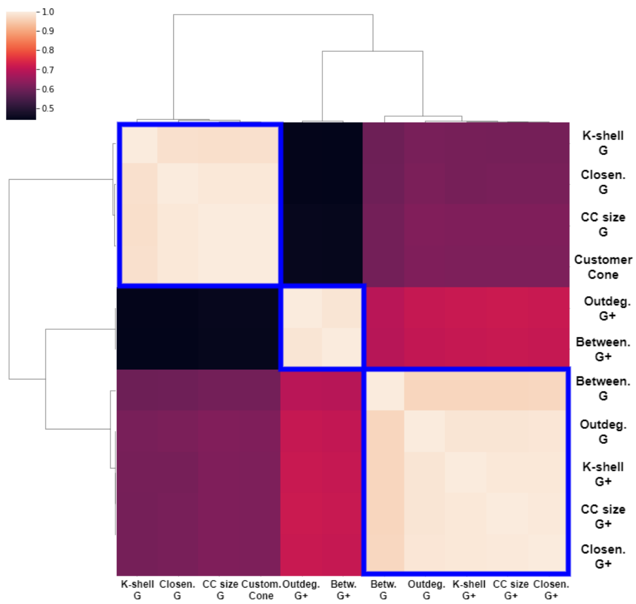

Figure 6. Clustermap of Kendall correlation coefficient, applied to the score lists of various ASN ranking metrics

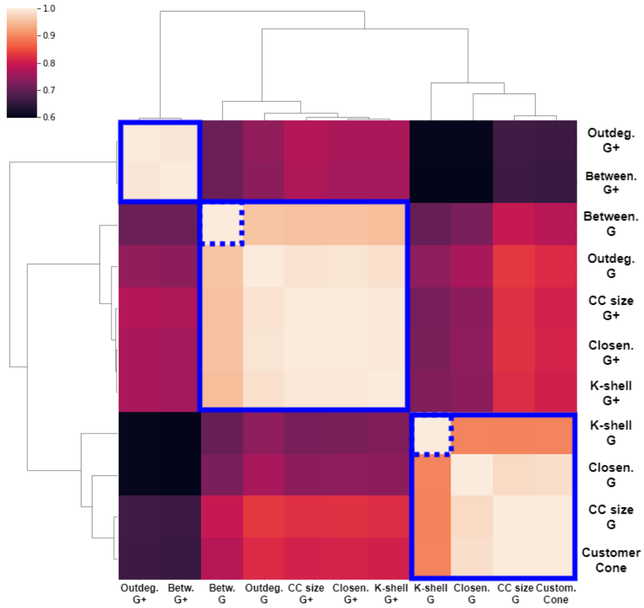

Figure 7. Clustermap of Spearman correlation coefficient, applied to the score lists of various ASN ranking metrics with leaf ASes removed