Exploration of Community Detection Using Max-Min Modularity

Cassandra Spath[1]

[1] *University of Colorado Boulder*

There are several problems with modularity. One of these is that only existing edges are factored into the calculations. This makes it so that two very different networks might have the same modularity. One proposed method for improving this is using max-min modularity. Does this max-min modularity improve over the results of modularity? If so, what cases are causing the improvement? We will examine three small real-world graphs for results including Zachary's Karate Club, Sawmill Strikes Communication, and Mexican Politicians. These three networks were chosen since they are small enough to analyze and also have communities associated with them that were used as the ground truth. There is a clear improvement in max-min modularity for these networks compared to modularity. This is primarily because of nodes with few connections and communities made up of two smaller clique-like structures.

I Introduction

There are many datasets that can be represented as networks with entities as vertices and relationships as edges. Social networks are becoming increasingly prevalent through the use of technology. In these networks, the nodes typically are people or users and the edges are communications online or in-person. These social networks often have community structures based on location, age, race, and many other factors. However, in many networks, especially larger networks, these communities are not known. This has brought about the field of community mining. In which the goal is to locate the communities in the network.

Once these communities are found, there are many applications depending on the network type. Suggesting links is used across platforms such as suggesting new friends on social media. Improving search results often is based on communities for retail websites.

As an ongoing field of research, the are many data mining algorithms. These include supervised classification, association rule mining, and clustering analysis. However, many of these algorithms are trying to identify the model given an IID sample [1]. One emerging challenge is finding community structure in a heterogeneous data set with multiple types of nodes representing different types of entities.

We examined the proposed max-min modularity. This factors in the structure of the network and a set of user defined related pairs while performing community detection. It does this in a similar way to modularity, by comparing both the number of edges and the number of unrelated pairs in a community to the expected values in a random network. This measure was incorporated into a hierarchical clustering algorithm that greedily optimizes max-min modularity while detection communities in the network.

We tested the community detection algorithm by comparing its results to a similar algorithm that uses modularity Q instead of max-min modularity. For testing, we used three small real-world networks, Zachary's Karate Club, Sawmill Strikes Communication, and Mexican Politicians. These were chosen since they are small enough to analyze, and they have communities associated with them that are considered the ground truth. This allowed us to compare the max-min modularity against modularity to see the differences, as well as calculate the adjusted rand index and improvement.

Finally, we examine which cases were causing the most improvement for max-min modularity compared to modularity. These were all looked at more closely to determine what allowed for this improvement. Additionally, there are some cases where max-min modularity was still having problems compared to the ground truth communities. These were also examined to see if any improvements could be made here.

II Methods

Modularity

Modularity Q is a measure used to quantify the groups of the nodes in a network [2]. It does this by comparing the actual number of edges in a community to the expected number of edges. By doing this, modularity is comparing the network to a randomized network with the same vertices and degrees. This is used to determine if the edges in the network are focused in a community or have a more random structure.

Given an Adjacency matrix A, degree sequence v for the vertices, and labeling of vertices in communities x, define m as the number of edges or $sum(A)/2$ and the Kronecker delta function $\delta(x_i, x_j)$ as 1 if x_i is in the same community as x_j and 0 otherwise. The modularity for an undirected network is defined as:

$$Q = \frac{1}{2m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{2m}) \delta(x_i, x_j)$$

The sum only factors in ij pairs that are in the same community since the Kronecker delta function is multi-

plied by each term. The value in the adjacency matrix indicates if the pair has an edge. The fractional value $\frac{k_i k_j}{2m}$ is the probability of the two nodes having an edge in a random graph. The sum is divided by $2m$ to keep the values between -1 and 1.

The modularity of a random graph would be near zero regardless of the communities. This is because for each community, there will approximately the same number of edges as expected. Thus, the sum will have some negative terms and some positive terms which will sum to be close to zero. If the communities fit the graph well, the score will be positive since there will be more edges within the community than expected. This will cause more positive terms in the sum and result in a higher score. However, if the communities are a bad fit for the graph, the score will be negative. There will be very few edges within the community resulting in more negative values in the sum and a lower score.

Since there are still edges between communities and within communities, scores near 1 and -1 are quite rare. The average modularity is 0.3 to 0.7 [3].

Downsides of Modularity

Although effective in community detection, this measure of modularity has several shortcomings. The entire network must be provided to calculate modularity. This makes the measure more difficult on larger networks. For some, like the Worldwide Web, getting this structure is extremely difficult. Additionally, this makes computation and storage much more difficult since the network can't be simplified. One solution that has come about from this problem is local modularity [4]. This measures the local community structure rather than the global community structure.

Another issue with modularity is its resolution limit. Using it to detect community structure has failed to identify smaller communities [5]. One main method used to address this issue is recursive algorithms [6].

Finally, the measure for modularity only factors in existing edges [7]. Edges that are missing from the graph are not factored into the metric. Thus, two networks with different structures can have the modularity score.

The two networks in FIG 1 have the modularity score when grouped as shown. Although the red community is the same in both networks, the blue community varies. The left network has 6 nodes in the blue community with 8 edges between them. Whereas the right graph has 9 nodes and 8 edges between them. The left blue community has more of the possible edges between nodes compared to the right community. Since the modularity score doesn't factor in missing edges, these very different graphs will have the same score. The left graph would result in the communities in FIG 1 when maximizing modularity. However, the right graph would break the blue community into two communities, nodes 1-4 and nodes 5-8. Regardless, the two communities fit the graph on the left much better than the graph on the right. Thus, these two networks should have different modularity scores since they have a different structure.

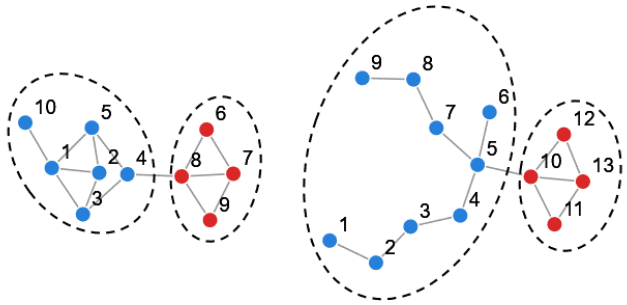

FIG. 1. Two networks with same modularity score when grouped into two communities by colors denoted.

Max-Min Modularity

The goal of max-min modularity is to address the problem that modularity doesn't factor in edges that don't exist in the graph [8]. This will be done by modifying modularity to include a calculation with the edges that don't exist within a community as well as those that do. However, considering all the edges that don't exist in a community would not help the performance since this would assume that communities are cliques. Thus, we will define related pairs as pairs of vertices that are not connected but could still be in the same community. Conversely, unrelated pairs are those that are not connected but should not be in the same community. The goal is to maximize the number of edges within the community while also minimizing the number of unrelated pairs. This will be done using max-min modularity.

The maximizing step is to maximize the number of edges in the community compared to the expected number of edges in a community for a random graph. This is the same as modularity as defined above. The minimizing set is to minimize the number of unrelated pairs in a community compared to the expected number of related pairs for a random graph. This will be defined as a new min modularity. Thus, when there are many edges within the community and few unrelated pairs, we will have a high max-min modularity, and we have found good communities for the network.

Given a set of user-defined related pairs U, an adjacency matrix A for the undirected graph G, degree sequence v for the vertices, and labeling of vertices in communities x. Define m as the number of edges or $sum(A)/2$, n as the number of vertices, and the Kronecker delta function $\delta(x_i, x_j)$. As discussed above the

maximizing factor is the same as the modularity:

$$Q_{max} = \frac{1}{2m}\sum_{ij}(A_{ij} - \frac{k_i k_j}{2m})\delta(x_i, x_j)$$

.

To simplify this, we will define the degree matrix K as follows:

$$K_{ij} = \frac{k_i k_j}{2m}$$

Next, we need to add the minimization of the unrelated pairs. We can easily see that if there are fewer unrelated pairs within a community in the original graph, there are few edges within the community in the compliment graph of related pairs. Thus, a better community for the compliment graph will be indicative of a poorly connected community in the compliment graph. Thus, we can calculate the modularity of this compliment graph. By minimizing the modularity score in the compliment, there is better structure in the original graph.

Define the compliment graph G' based on the adjacency matrix A and related pairs U such that there will be an edge in the new adjacency matrix A', if and only if there is no edge in the original graph, and the vertices are not a related pair. Thus, the new adjacency matrix can be defined as:

$$A'_{ij} = 1 \leftrightarrow A_{ij} = 0 \wedge (i,j) \notin U$$

This will give a new degree sequence k'. Using this compliment graph,the number of edges m' is $(n(n-1) - 2m - 2|U|)/2$. This is the twice number of possible edges $n(n-1)$ minus twice the number of edges in the original graph $2m$ minus twice the number of related pairs $2|U|$. This gives twice the number of edges in the new compliment graph. This will be used to define the min modularity:

$$Q_{min} = \frac{1}{2m'}\sum_{ij}(A'_{ij} - \frac{k'_i k'_j}{2m'})\delta(x_i, x_j)$$

To simplify this, we will define the degree matrix K' as follows:

$$K'_{ij} = \frac{k'_i k'_j}{2m'} = \frac{k'_i k'_j}{n(n-1) - 2m - 2|U|}$$

Now we combine the two modularities such that we are maximizing Q_{max} and minimizing Q_{min}. This can be done with the following formula:

$$\begin{aligned} Q_{mm} &= Q_{max} - Q_{min} \\ &= \sum_{ij}[\frac{1}{2m}(A_{ij} - K_{ij}) - \frac{1}{2m'}(A'_{ij} - K'_{ij})]\delta(x_i, x_j)] \end{aligned}$$

Related Pairs

As discussed above, the max-min modularity uses related pairs to only penalize unrelated pairs in the same community. There are a couple of ways that related pairs can be defined: structurally or using metadata.

When defining related pairs structurally, the edges are the only things considered in the definition. An easy choice is the existence of a shared neighbor. Thus, if nodes a and b are not connected but both connect to some other node c, they would be a related pair. However, if they didn't have this shared neighbor c, then they would be an unrelated pair. As an unrelated pair, if the two nodes are in the same community, they would have an negative value in the max-min modularity.

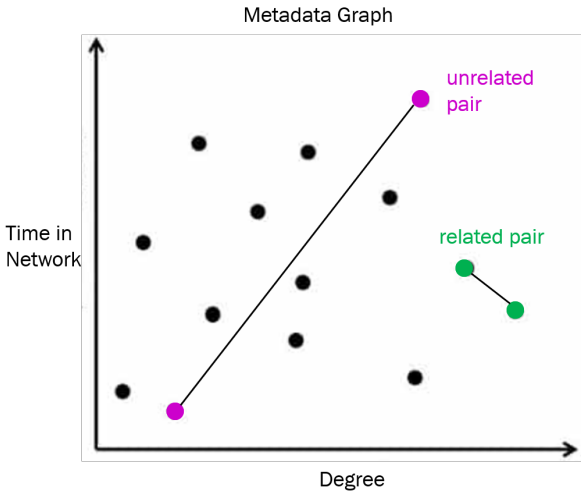

FIG. 2. Plot of metadata for a network with example related and unrelated pairs.

Another way to define the related pairs uses the metadata. As seen in FIG 2, the metadata provided can be plotted. This could include some structural measures such as degree as well as age, race, gender, and other demographics. Then based on the plot, the vertices that are close together such as the green one would be a related pair and those further apart such as the pink would be an unrelated pair. A simple metric such as Euclidean distance could be used to calculate this. The plot would have k dimensions where k is number of metrics being used.

The most challenging part of this definition for related pairs is determining the cutoffs for distance to define related and unrelated. The fastest way would be to have a value acting as a boundary. Thus, any distance above this boundary would be unrelated and under would be related. While this is likely true for the points further from the boundary, the points near it may be incorrect. Additionally, this value still needs to be chosen. Ideally the points would fall into distinct communities as seen in FIG 3. Algorithms such as k-means clustering would

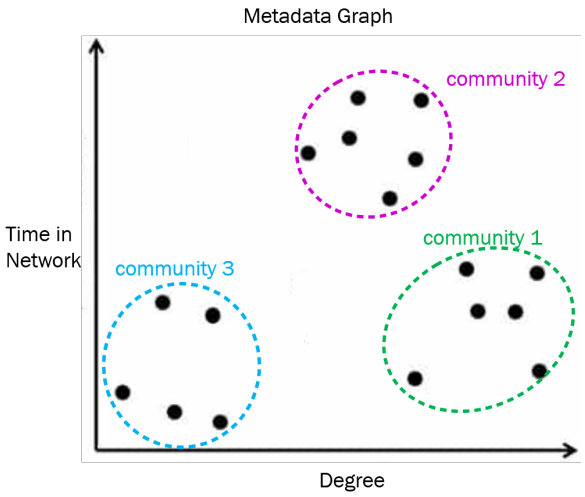

FIG. 3. Plot of metadata with groups based on distances between points.

fine these communities from the plot [9]. However, this requires the number of communities being detected initially which is not a piece of knowledge that we know. A variation of this would be mean-shift clustering [10]. However, this run time is much worse and does not scale well with higher dimensions. While any of these methods could be used, there is a clear trade-off of accuracy and runtime.

Community Detection

To get communities with max-min modularity, we must define a clustering algorithm. The following algorithm is a hierarchical clustering algorithm which greedily optimizes Q_{mm} [11].

Input: An undirected network $G = (V, E)$ where $|V| = n$ and $|E| = m$, an adjacency matrix A, user-defined related pairs set U.

Output: Community assignments for V: $C_1, C_2, \ldots C_d$

1. Start by assuming each node is in an independent community. Thus, node i is in community i. This gives n starting communities.

2. Create a compliment adjacency matrix A' where A'_{ij} is the number of unrelated pairs between communities i and j:

$$A'_{ij} = 1 \leftrightarrow (i,j) \notin U \wedge A_{ij} = 0$$

3. Compute a matrix with ΔQ_{ij} for each existing pair i, j:

$$\Delta Q_{ij} = -\left(\frac{k_i}{2m}^2\right) + \left(\frac{k'_i}{2m'}^2\right)$$

4. While there are more than two communities, select the pair i, j with the largest ΔQ. Merge the two communities together.

   Update the adjacency matrix A by merging row i with j and column i with column j using addition. Delete row and column j.

   Update the adjacency matrix A' the same way.

   Compute the new ΔQ for each existing pair i, j:

$$\Delta Q_{ij} = \frac{A_{ij}}{4m^2} - \frac{k_i * k_j}{2m^2} - \frac{A'_{ij}}{4m'^2} + \frac{k'_i * k'_j}{2m'^2}$$

   Save the communities for this iteration with the min-max modularity associated with it.

5. Determine which of the saved max-min modularities is the largest.

6. Return the communities $C_1, C_2, \ldots C_d$ associated with the best modularity.

This algorithm will create an adjacency matrix in $O(n^2)$ time. Then it will iterate until there is one community remaining. Since it starts with n communities, this will take $O(n)$ iterations. For each iteration, there are merges two in the matrices. With up to $n \times n$ matrix, the each row or column merge will take $O(n)$ time. Since there are a constant number of merges, the total merge time is $O(n)$ for each iteration. Lastly, the ΔQ values are calculated for every existing edge. With m edges, this will take $O(m)$ time each iteration. Thus, each iteration of the algorithm with take $O(n+m)$ time. Since there are $O(n)$ iterations, this algorithm runs in $O(n(n+m))$ time. Worst case scenarios the number of edges m is $O(n^2)$ making the algorithm run in $O(n^3)$. However, if the network is sparse with $O(n)$ edges, then the run time will be $O(n^2)$.

A similar algorithm will be used for modularity. However, the compliment adjacency matrix will be not be used. Additionally, the initial ΔQ_{ij} for each existing pair will be as follows:

$$\Delta Q_{ij} = -\left(\frac{k_i}{2m}^2\right)$$

The iterations will be the same. However, the ΔQ for each existing pair will be computed as follows:

$$\Delta Q_{ij} = \frac{A_{ij}}{4m^2} - \frac{k_i * k_j}{2m^2}$$

This algorithm for modularity has the same behavior but does half as many operations since the computations for modularity are simpler and there is only once matrix being merged. Thus, its runtime will also be $O(n(n+m))$.

	Karate Club	Sawmill Strikes	Mexican Politicians
Ground Truth Communities	2	3	2
Modularity Q Communities	3	4	3
Max-Min Modularity Communities	2	3	3
Modularity Q ARI	0.680	0.664	0.255
Max-Min Modularity ARI	1.00	1.00	0.359
Improvement	47.1%	50.6%	40.7%

FIG. 4. Communities and ARI scores for Modularity Q and max-min modularity as well as the improvement.

III Results

To test min-max modularity, we have run the community detection algorithm using both max-min modularity and modularity for comparison. This was run on several small, real-world networks that have communities that are accepted as ground truth. After running the community detection algorithm, the resulting communities needed to be compared against the ground truth. This accuracy was computed using the Adjusted Rand Index (ARI) [12].

The ARI calculates the similarity between the ground truth communities G and the partition from the community detection P. Define a as the number of vertex pairs in the same community in G and P, b as the number of pairs that are in different communities for G and P, c as the number of pairs that are in the same community in G but different communities in P, and d as the number of pair that are in different communities in G but the same community in P. Using these values, ARI is defined as

$$ARI(G,P) = \frac{2(ab-cd)}{(a+c)(b+c)+(a+d)(b+d)}$$

For two similar partitions, the values for a and b will be larger while the values for c and d will be smaller. Since the communities align, most pairs have the same result in the two partitions whereas few pairs will have different results in the two partitions. Thus, if partitions G and P are the same, then the ARI score will be 1. If communities for P are chosen at random, the ARI will be near 0. The number of communities, ARI scores, and improvement for each network can be seen in FIG 4.

Karate Club

The karate club network is a well-known network that was tested first [13]. It involved the split of the 34 member of the club when the administrator and instructor had a dispute. This resulting in the splitting of the club into two new clubs. The edges were determined by estimating several measures the strength between the members. As seen in FIG 5, there are two ground truth communities representing the new clubs formed by the instructor seen in blue and administrator seen in red.

Since this is a social network, the related pairs are defined by sharing a neighbor. Thus, if two people have a mutual friend they are related for the max-min modularity. Thus, these pairs will not be penalized by max-min modularity whereas unrelated pairs in the same community will be.

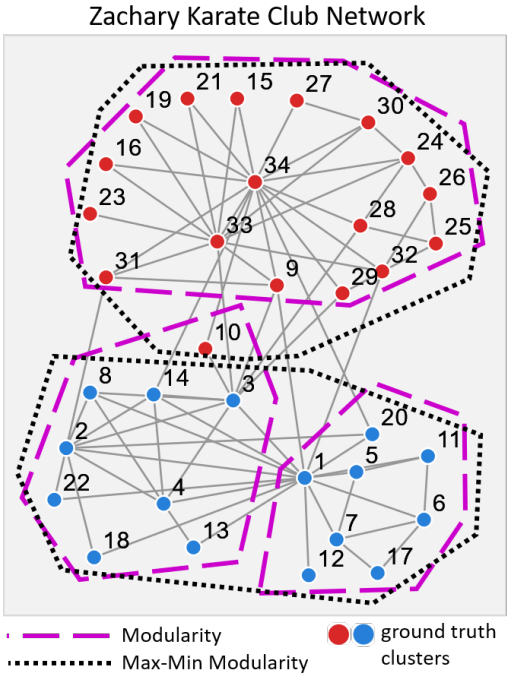

FIG. 5. Karate Club network color coded by ground truth communities. The grouping represent the communities discovered by modularity and max-min modularity.

As seen in FIG 5 and FIG 4, the modularity Q results in 3 communities with an ARI score of 0.680. The ground truth is 2 communities. The max-min modularity Q_{mm} finds these 2 communities exactly resulting in an ARI score of 1. This is a 47.1% improvement. There are 10 edges between these two communities making the maximizing factor relatively effective. However, the main reason the max-min modularity has so much improvement is the related pairs. Vertex 1 is connected to main other vertices. Many of these vertices have only a couple connections with other vertices. Thus, almost every vertex in the blue group become either connected or a related pair. This encourages merging the blue group

into one community rather than keeping it split with the modularity Q. This has a similar effect for vertex 10. This vertex is associate with a subset of blue vertices for modularity Q. However, since it is connected to vertex 34, it has many related pairs in the red group since vertex 34 is connected to almost every vertex in the red group. It has fewer related pairs in the blue group which causes it to be correctly associated with the red community for the max-min modularity.

Sawmill Strikes

The sawmill strikes communication network was the next one used [14]. This network examines the communication of the 24 employees at a sawmill after a strike. The edges indicate that two employees discussed the strike with each other frequently. This network has 3 ground truth communities seen in FIG 6. These communities are primarily based on age and the language they speak. The young Spanish speaking community in blue, it almost completely disconnected with only one edge with the young English speaking community in green. This group has a few connections with the older English speaking group in red. Once again, the related pairs are defined by sharing a neighbor since this is another communication network.

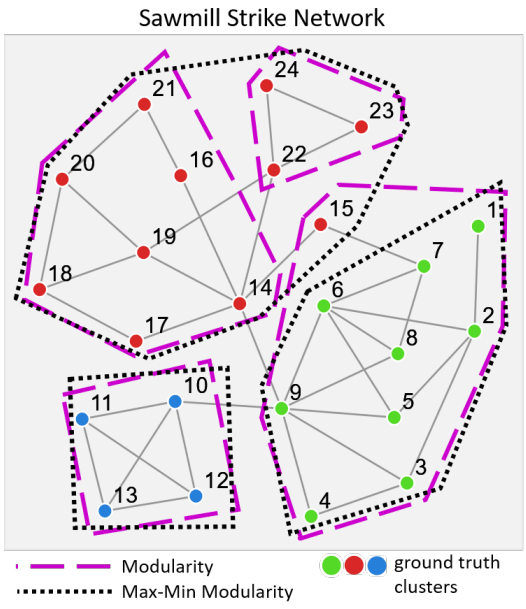

FIG. 6. Sawmill Strikes Communication network color coded by ground truth communities. The grouping represent the communities discovered by modularity and max-min modularity.

As seen in FIG 6, the groups are fairly disconnected making community detection easier for most algorithms. The modularity Q finds 4 communities here. The blue is correct and the green are all together with one red vertex. However, the red is split into two groups. Since the smaller group of 3 has only on vertex with 2 connections, it's easy to see how this would happen. Modularity Q has an ARI score of 0.664 whereas max-min modularity has an ARI score of 1.00 since it gets the ground truth communities. This is an improvement of 50.6%.

The blue community is easy to detect since it is a clique with only one vertex with one edge outsize the community. The green community has a good number of connections and related pairs. Since there are only a few unrelated pairs within this community, Qmm has no problem detecting it. The last community is the red community which was broken up with modularity Q. However, with modularity Q_{mm} vertex 15 has many related pairs with red vertices since it is connected to 14. In contrast it only has a few related pairs with green vertices. Thus, the minimizing factor will be smaller when 15 is group with the other red vertices. Additionally, vertices 22, 23, and 24 each have several related pairs with red vertices. This causes this group to be joined with the other red vertices into one community using Q_{mm}.

Mexican Politicians

The last network tested was the Mexican politicians network [14]. This network has 34 politicians. Each edge is indicative of a social tie between the two politicians. There are two ground truth communities seen in FIG 7. The civilians are in blue while the military personal are in red. These communities are more connected with many edges between the community. This makes community detection more difficult since they are less distinct. This network was a social network so the related pairs were defined by sharing a neighbor.

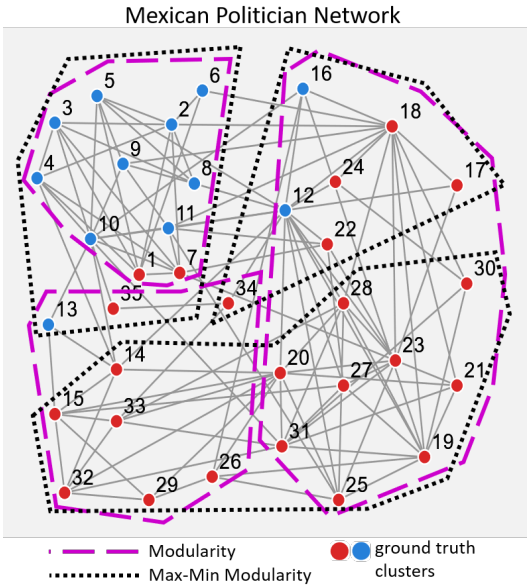

FIG. 7. Mexican Politician network color coded by ground truth communities. The grouping represent the communities discovered by modularity and max-min modularity.

As seen FIG 7, there are 2 ground truth communi-

ties. Both modularity and max-min modularity find 3 communities. Modularity Q results in an ARI score of 0.255. Max-min modularity has an ARI score of 0.359 which is a 40.7% improvement. One of the most notable improvements is that it has grouped the large set of red nodes at the bottom in one community. Since many of these nodes have connections to 20, 26, or 31, many of these nodes are then related pairs. This allows them to be connected into one larger community rather than split in two different communities.

Max-min modularity has also moved 13 from a red community into one with most of the blues. This is because it is connected with node 10 which has connections across most the blue community. This encourages 13 to be in this community since it has fewer unrelated pairs than with the bottom red community we previously discussed. However, it moved node 35 which is red into this community for the same reason.

The last community for max-min modularity is composed of 2 blue nodes and 5 red nodes in the top right. These nodes are very well connected with no unrelated pairs. Many of these nodes have many connections with blue and red nodes in the other communities. This makes it such that there are more unrelated pairs between these nodes and the nodes in the communities they should be in. Due to this, it minimizing modularity if these nodes are kept in one community rather than moved out to the other two communities.

One of the main downfalls for both modularities is nodes 1 and 7. These red nodes are in a primarily blue community. The main reason for this is that the nodes have more connections with blue nodes. Thus, for the modularity Q they clearly belong in this community. Additionally, for the max-min modularity, there are almost no unrelated pairs between them and the nodes in the blue community. However, for the red community, they have many more unrelated pairs due their few connections with this community and the fact that the community is larger. The community structure for this network is not very apparent structurally. This makes any form of community detection using structural characteristics difficult.

IV Discussion

After looking at these three graph, it's clear that max-min modularity improves upon modularity. Although it still had trouble when the community structure was less clear, max-min modularity did very well for less connected networks that modularity struggled with. There were two distinct cases which max-min modularity helped.

The first case is a node with few connections. This node have only a couple edges, often with more edges going to a community that is not its community. With modularity, this node is grouped with the second community with which it has more connections but doesn't belong in. Adding the related and unrelated pairs helped make associations for this node that did not have many connections with their community. If the node was connected to one or more of the central nodes from the community it belongs in, then it would have many related pairs within its community. If it has more connections to the fringe nodes of the other community, the number of related pairs with the members of the other community will be lower. This will encourage moving it to the correct community which we saw in each of the networks. However, if the connections to the other community are also central nodes, then the number of related pairs with this incorrect community is much higher. Thus, the node may still be associated with the wrong community depending on the sizes and community structure of the two communities. However, if the node doesn't connect to any of the central nodes in its community and has many connections with another community, especially the central nodes, then this node is likely to be grouped with the incorrect community. This is a challenge for any algorithm when doing community detection structurally.

The second case is a community that has two smaller clique-like structures with some edges between them. This was seen in the sawmill strike and Mexican politician networks. Since this two sub-communities are clique-like, modularity will group them into two communities during the hierarchical clustering algorithm. However, merging the communities doesn't improve the modularity, thus, this is not chosen leaving the one community divided. When adding related pairs, max-min modularity greatly improves these results. Since there are connections between them and the two sub-communities have clique-like structure, almost all of these nodes will be either connected or related pairs. This means that it increase max-min modularity to merge them together into one larger community during the hierarchical clustering algorithm. This is one of the primary ways that max-min modularity improves on the number of communities in a graph where modularity is incorrect. Although this was only seen for cases with sub communities within one community, theoretically this would for for more sub communities. It would be worth looking for a graph that has this structure to see the performance of max-min modularity.

Thus, by incorporating related pairs, max-min modularity is predicting if two nodes should have an edge between them. This encourages more merges before the score will drop-off causing fewer communities. This causes great improvement of modularity Q which often results in too many communities.

V Future Work

During this project, I began working on using metadata to define the related pairs for max-min modularity. Some of the biggest struggles that I ran into were how to dis-

tinguish related and unrelated pairs. Due to this, I didn't have much success with using the metadata.

Moving forward, I propose starting with a simple boundary define as function of the minimum and maximum distance or starting with related pairs defined structurally. From there community detection can be done with max-min modularity. At this point, take the found number of communities and run k-means clustering with this many communities. Perform community detection again using the new related pairs. This process could be done once. Alternatively, the k-means clustering and community detection could be repeated for a specified number of iterations or until the community detection algorithm ends with the same number of communities used for k-means clustering.

Additionally, I would want to test the metadata on larger datasets. I believe some of the issues that I had were due to the smaller number of nodes in the dataset. With more nodes, there are more points in the graph. This would likely help when running k-means clustering on the data.

Acknowledgments

I thank Professor Dan Larremore for his feedback and support on this project and throughout the semester.

References

[1] L. Getoor and C. P. Diehl, "Link mining: a survey," *Acm Sigkdd Explorations Newsletter*, vol. 7, no. 2, pp. 3–12, 2005.

[2] M. E. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical review E*, vol. 69, no. 2, p. 026113, 2004.

[3] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[4] A. Clauset, "Finding local community structure in networks," *Physical review E*, vol. 72, no. 2, p. 026132, 2005.

[5] S. Fortunato and M. Barthelemy, "Resolution limit in community detection," *Proceedings of the national academy of sciences*, vol. 104, no. 1, pp. 36–41, 2007.

[6] J. Ruan and W. Zhang, "Identifying network communities with a high resolution," *Physical Review E*, vol. 77, no. 1, p. 016104, 2008.

[7] J. Scripps, P.-N. Tan, and A.-H. Esfahanian, "Exploration of link structure and community-based node roles in network analysis," in *Seventh IEEE international conference on data mining (ICDM 2007)*, pp. 649–654, IEEE, 2007.

[8] J. Chen, O. R. Zaïane, and R. Goebel, "Detecting communities in social networks using max-min modularity," in *Proceedings of the 2009 SIAM international conference on data mining*, pp. 978–989, SIAM, 2009.

[9] J. A. Hartigan and M. A. Wong, "Algorithm as 136: A k-means clustering algorithm," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1979.

[10] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[11] C. Spath, "csci5352." `https://github.com/Cass-S/csci5352`.

[12] K. Y. Yip, D. W. Cheung, and M. K. Ng, "Harp: A practical projected clustering algorithm," *IEEE Transactions on knowledge and data engineering*, vol. 16, no. 11, pp. 1387–1397, 2004.

[13] W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of anthropological research*, vol. 33, no. 4, pp. 452–473, 1977.

[14] Pajek. `http://vlado.fmf.uni-lj.si/pub/networks/pajek/`.