Multi-objective community detection for bipartite graphs

Julia Handl
Alliance Manchester Business School, University of Manchester
julia.handl@manchester.ac.uk

Luis Ospina-Forero
Alliance Manchester Business School, University of Manchester

Tristan Cann
School of Computer Science, University of Exeter

ABSTRACT

Bipartite graphs pose particular challenges for community detection. Here, we explore a multi-objective approach to community detection that simultaneously considers multiple different one-mode projections. Building on existing packages in the pymoo and igraph packages, we show how existing evolutionary approaches to clustering and community detection can be adapted to tackle bipartite community detection, and demonstrate the effectiveness of the approach on problems with a known ground truth. Our results show that multi-objective optimization is useful in this problem, both as a means of reducing biases of the modularity measure and as a means of combining the information obtained from individual one-mode projections.

CCS CONCEPTS

• **Theory of computation** → *Design and analysis of algorithms*; • **Computing methodologies** → Cluster analysis.

KEYWORDS

Clustering; community detection; bipartite graphs; multiobjective machine learning

ACM Reference Format:
Julia Handl, Luis Ospina-Forero, and Tristan Cann. 2022. Multi-objective community detection for bipartite graphs. In *Genetic and Evolutionary Computation Conference (GECCO '22), July 13–17, 2022, Boston, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3321707.3321761

1 INTRODUCTION

Bipartite graphs arise in a wide range of problems, most prominently recommender systems, but methodologies for their analysis are in their infancy compared to the range of methodologies for standard graph structures [**? ?**]. The most common approach to community detection is based on the projection of a bipartite graph structure into the space of its top or bottom nodes (see Figure 1), but the limitations of this approach are well recognised [**? ?**].

Here, we report initial results of the development of a multi-objective framework for this problem that is able to address the limitations of existing projection0based approach, and potentially able to enrich information embedded in the graph structure with additional input from a range of information source A focus in this work has been on the implementation of our approach on top of specialised Python packages for graph analysis (igraph) and multi-objective optimization (pymoo), to allow for easy exchange of and experimentation with individual algorithm components.

GECCO '22, July 13–17, 2022, Boston, USA

ACM ISBN 978-1-4503-6111-8/19/07…$15.00
https://doi.org/10.1145/3321707.3321761

2 RELATED WORK

2.1 What is a bipartite graph

2.2 Projections

2.3 Community detection techniques including modularity

cite EA papers Many-objective optimization for community detection in multi-layer networks

Evolutionary Computation for Community Detection in Networks: A Review

cite projection papers

cite papers on limitations of modularity

3 EVOLVING CLUSTERS

Building on recent work in multi-criterion and multi-view clustering, we explore the use of a multi-objective algorithm to support the integration of the two projections. Conceptually, the betweenness centrality of an edge describes the amount of traffic that flows through an edge. Mathematically, it is given as

$c_B(e) = \sum_{s,t} \frac{\sigma(s,t|e)}{\sigma(s,t)}$

where

$$\sigma(s,t)$$

is the number of shortest paths between vertices $s,t \in V$ and $\sigma(s,t|e) is the number of these paths that passes through edge$e

In other words, betweenness centrality provides an indication of the importance of an edge in bridging communities within a graph. We exploit this principle in the design of our representation. As we are using a minimum spanning tree to bias the search, we would like to ensure that meaningful communities can be identified through the removal of individual links in that tree. We improve the ability of the representation to do so by constructing the MST on a weighted version of our graph instead of its unweighted forms. Specifically, the edge for each weight is given by its betweenness centrality, which aims to ensure that cross-community edges are minimized within the MST.

Definition of modularity Definition of the minimum spanning tree Definition of a disconnected component

3.1 Multi-objective optimizer

Our implementation assumes use of a multi-objective optimizer as implemented in the pymoo package. Here, our evolutionary algorithm of choice is NSGA-II, with a population size of 50 and run for 1000 generations, but different choices could easily be interchanged.

3.2 Representation

We employ the linkage-based adjacency scheme successfully employed in previous work on multi-criterion clustering [**?**] and multi-criterion community detection [**?**].

3.3 Objective

Our approach separately maximizes the modularity of each projection. The projections of the bipartite graph structures, and their modularity, are calculated using implementations available in the pymoo package. Minimization of the number of communities is added as a third objective, to compensate modularity's tendency to overestimate the number of communities.

3.4 Initialization

The adjacency-based representation is initialised using the minimum spanning tree, an approach designed to bias the search space that has proven successful in previous work on cluster analysis. More specifically, the minimum spanning tree (MST) is derived using edge betweenness as a proxy for edge weights. The resulting MST is then converted to a link-based encoding, whith value i at position j indicating that j is linked to its ith neighbour. A value of 0 serves as a proxy that indicates a self-loop, i.e. a lack of a further link, and a choice of 0 can thus serve to break up a solution into multiple disconnected components. To generate diversity within the original population, the Fast Greedy heuristic is used to obtain fifty separate partitions with k ranging from 1 to 50, in steps of 1. Each individual in the original population is adjusted to be consistent with one of these partitions: specifically, we identify any edges contained in the MST and linking separate communities in the Fast Greedy solution. There are removed from the solution, i.e. replaced by a self-loop, using the proxy value 0.

3.5 Variation operators

A simple graph-based mutation is used that allows each node to be connected to one of its neighbours in the original bipartite graph structure or to form a self-loop, which corresponds to the deletion of the associated edge in the minimum spanning tree. Pymoo's polynomial mutation operator int_pm, with default mutation probability is used. As all edges in the original graph are unweighted, probability of mutation is equal across all possible edges.

Uniform crossover has been shown to be appropriate for this representation, and is used with probability $p = 0.3$.

4 EXPERIMENTS

The experiments reported in this manuscript are designed as a proof-of-principle to explore the potential of a multi-objective approach in the context of simple synthetic benchmark sets.

4.1 Benchmark data

A simple generating model, based on a stochastic block model, is used:

- A bipartite graph structure with L vertices assigned to the lower part of the graph and U vertices assigned to the upper part of the graph.
- A two-class structure supported by the lower and upper vertices of the graph, with percentage M_L of lower nodes belonging to class 0, and $M_L - 1$ to class 1, and percentage M_U of upper nodes belonging to class 0, and $M_U - 1$ to class 1.
- E edges that obey the bipartite structure of the graph, preferentially connect within-community vertices (with probability P), with equal probability for both classes, but are otherwise randomly distributed. No duplicate edges are allowed and edges in the original bipartite graph are unweighted.

Our experiments focus on performance on the giant component of the graph only. For each choice of parameters, 30 different instances are generated, and results are reported across these.

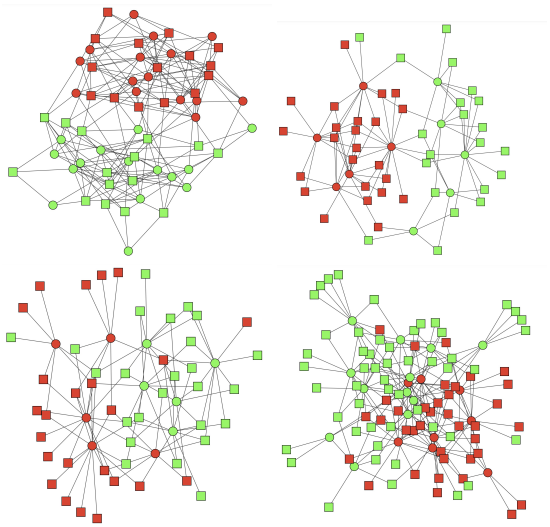

Figure 1: Example test instances with different ratios of top and bottom nodes, between-community connection and class imbalance. Top and bottom nodes are differentiated by shape and the ground truth is highlighted using color coding.

4.2 Contestant techniques

We compare to several generic community detection techniques available in the igraph package. This includes greedy optimization of modularity (Fast Greedy FG, [**?**]), a random walk method (Walking Trap WT, [**?**]) and a hierarchical algorithm employing edge betweenness (Edge Betweeness methods CE [**?**]). Out of these, only Fast Greedy employs modularity directly during the optimization. All three methods return hierarchies of communities and analysis focuses on the best solution contained in the set determined by $k \in [1, 15]$.

Additionally, we compare against two methods specialised in handling bipartite graph structures: the first of these is the BRIM algorithm, as implemented in the condor package. This approach returns the optimal modularity partition only, so is at a disadvantage, compared to other approaches, that are compared across a range of possible community numbers. The second approach, ML, is the multi-level method proposed by [], which involves the separate application of the Louvain method on each projection, followed by the identification of a consensus clustering through hierarchical agglomerative clustering on the resulting partitions []. The similarity between pairs of communities is measured as the number of edges that cross those two communities.

Finally, we compare against an alternative version of our own multi-objective framework, optimizing modularity of the full graph and the number of clusters only (i.e. using just two rather than three objectives). This is to highlight contribution arising from the choice of objectives (i.e. the simultaneous optimiation of multiple projections) rather than simply the use of a meta-heuristic optimizer. The resulting algorithms, 2d and 3d, both return an approximation front covering a range of values of k, and analysis focuses on the best solution contained in each front.

4.3 Performance assessment

Performance is assessed through comparison of the detected communities to the available ground truth. Similarity of the partitions is quantified using the Adjusted Rand Index, takes values in the range $[0, 1]$ and is suitable for the evaluation of performance across multiple values of k.

5 RESULTS

Our experiments incrementally analyse the robustness of the different techniques to a range of complicating factors, including edge sparsity, class imbalance, imbalance between lower and upper nodes, and percentage of between-community edges. Our results indicate distinct differences in the algorithms' ability to cope with these aspects, and a robustness of the multi-objective approach to all of these properties.

Figure 2 illustrates the performance of all algorithms on a simple problem with an equal number of lower and upper vertices, equally-sized classes, and 10 per cent between-community edges. It can be see that all algorithms perform well at recovering the ground truth, as may be expected. The BRIM algorithm shows the weakest performance, but it is the only algorithm that returns a single community structure (as compared to a single list of candidate solutions with different numbers of communities), which may contribute to this result.

Figure 3 contrasts this to performance on a graph with reduced connectivity. As the number of edges is reduced from 200 to 100, we can observe a significant drop in the performance of all approaches, with the worst impact observed for the BRIM algorithm. For the majority of instances, the other methods continue to display good performance in retrieving the true class structure.

Figure 4 considers the change in performance as the ratio between top and bottom vertices is varied from 1-1 to 5-1. The performance of the algorithms is not negatively affected by this adjustment. As a matter of fact, contrasting Figure 2 and 3, we can observe

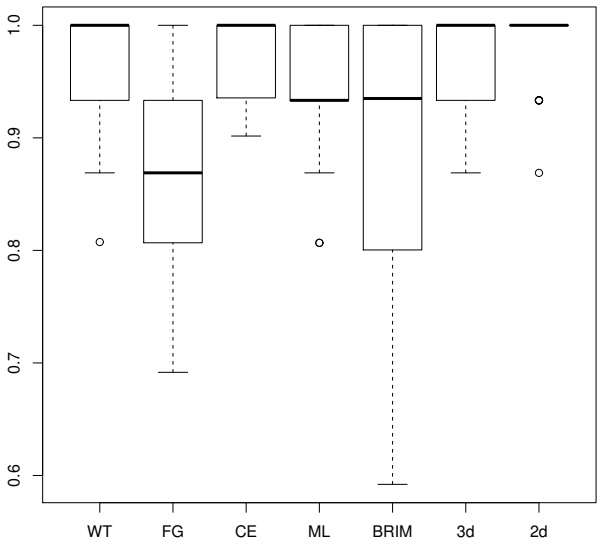

Figure 2: Adjusted Rand Index of best solution for 30 instances of a simple graph structure with a high node degree (200 edges).

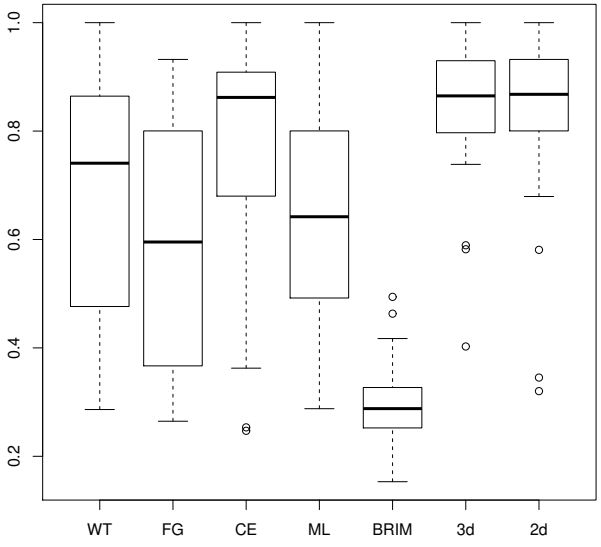

Figure 3: Adjusted Rand Index of best solution for 30 instances of a simple graph structure with a lower node degree (100 edges).

a general increase in performance as a result of this change. The likely reason is the associated increase in the average node degree in the bottom node set, which helps improve overall connectivity of the system.

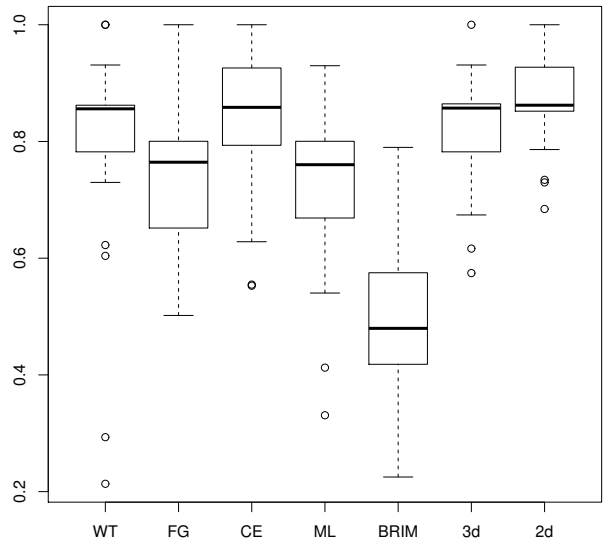

Figure 4: Adjusted Rand Index of best solution for 30 instances of a simple graph structure with 5-1 ratio of top to bottom nodes (other parameters unchanged relative to Figure 3).

Next, we consider an increase in the proportion of between-community edges, raising this from 10 to 20 per cent of edges. Figure 5 highlights that this has the expected negative impact on performance, affecting all algorithms and the multi-level approach, in particular.

Finally, we consider performance changes due to the relative size of the communities. Representative results are shown for generating models involving a mix of complex features, including a larger number of nodes (120 / 600), medium connectivity (200 / 1000 edges), medium level noise (20 per cent between community edges), and various levels of class imbalance (2-3 in both layers). The results indicate a clear advantage of the evolutionary approach in this framework.

To understand the behaviour of the algorithms in more detail, Figure 7 provides a representative example quantifying and contrasting the solution sets generated by the different different algorithms with respect to the optimization of modularity in each projection, optimization of overall modularity, and recovery of the ground truth (as measured by the Adjusted Rand Index), on an instance from the experiment in Figure 6. There are a number of interesting observations to be had.

First, our results highlight known issues with modularity optimization, indicating, in particular, that modularity at the bipartite

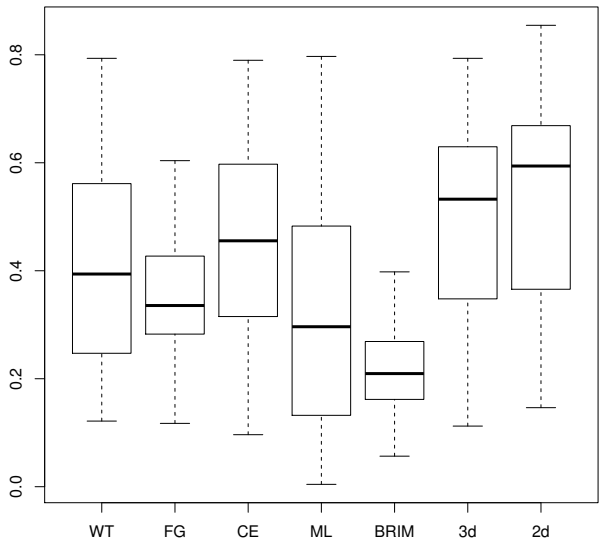

Figure 5: Adjusted Rand Index of best solution for 30 instances of a simple graph structure with increased between-community connectivity (other parameters unchanged relative to Figure 4).

graph level overestimates the number of communities and does not translate into the full optimization of the modularity of each projection. The first issue can be addressed by the optimization of modularity across a range of choices of k and that is the approach taken here for the Fast Greedy approach. The second issue cannot be addressed without adjustments to the modularity measure or use of an alternative formulation. Interestingly, our results highlight that optimization of the modularity of the two projections goes some way towards optimizing overall modularity, seemingly avoiding some of the pitfalls of its direct optimization.

Visualization of the solution sets further highlight a good level of correlation between the objectives used in our multi-objective optimizers. It can be see that, for simple problems, the final approximation sets do not typically exceeding a handful solutions, but the size of approximation sets increases with the complexity of the data. Interestingly, we can see that, even where > 10 solutions are generated, the various trade-offs solutions all translate to reasonable ARI values, highlighting that even the extremes of the front correspond to potentially useful solutions.

Vary noise in community structure Vary imbalance of upper and lower level Vary connectivity Vary community size

6 CONCLUSION AND FUTURE WORK

The work reported in this paper can be seen as a specific case of multi-view learning on graph structures. Our interest here has been the analysis of bipartite graph structures. In that scenario, the projection of a bipartite graph into its top or bottom layer provides

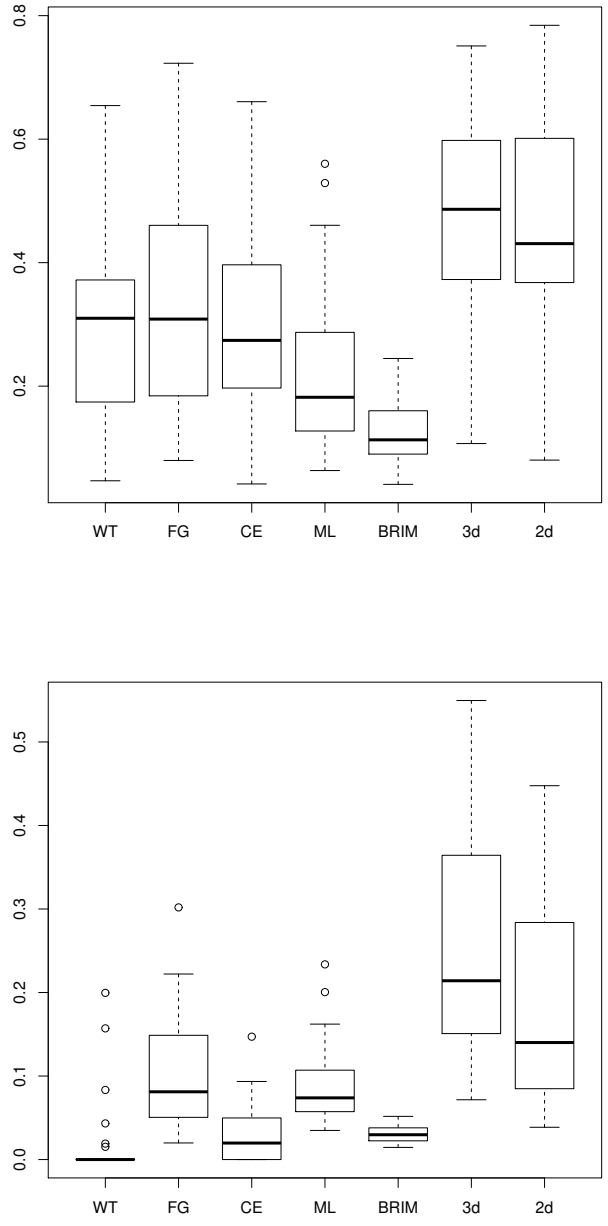

Figure 6: Adjusted Rand Index of best solution for 30 instances of two generating models incorporating a range of challenges.

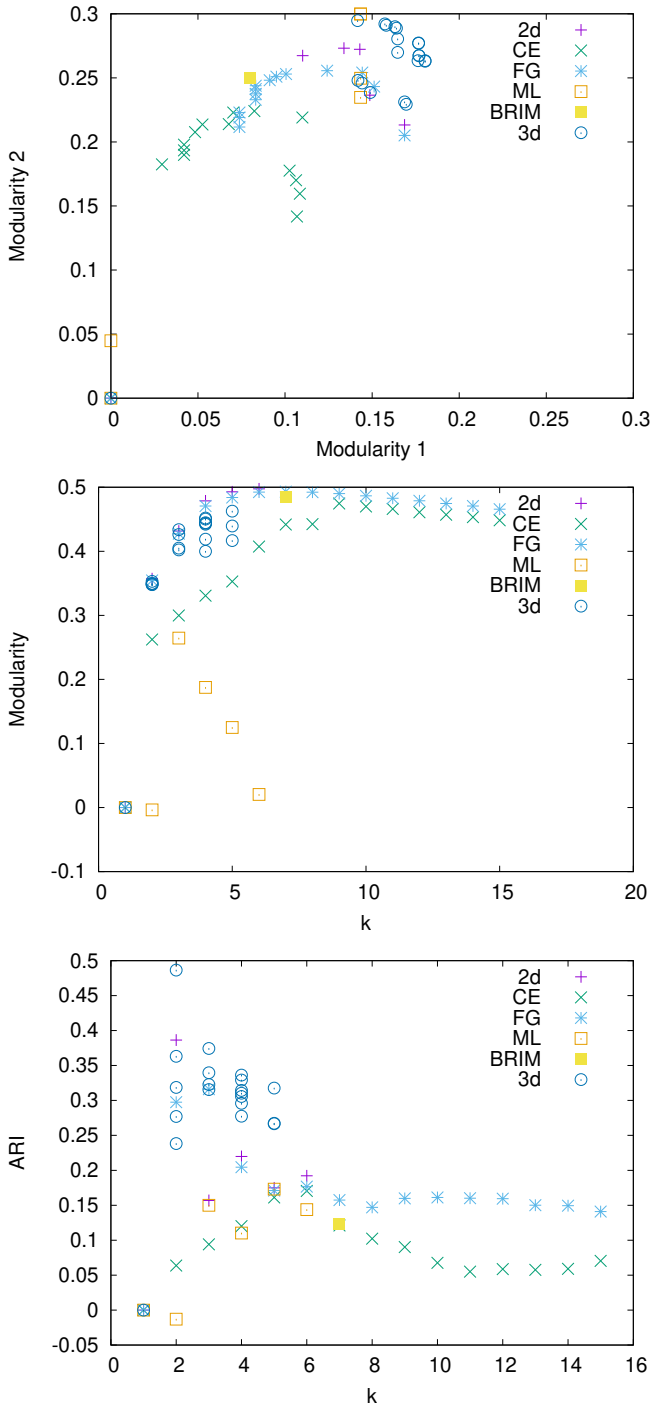

Figure 7: Indicative behaviour of different algorithms when viewed in different objective spaces.

two readily available views of the system that are amenable to objective functions devised for the optimization of univariate graph structures.

It is worth highlighting that the possible uses of our implementation is not limited to bipartite graph structures or projection-based objectives. The key assumption made by the approach is the availability of a single graph structure that is relevant to all objectives, as it is used to inform the adjacency-based representation of individual solutions, and thereby bias the search. If such a graph structure is

available, additional objectives could easily be added to integrate node-specific features or prior knowledge of class membership for specific nodes.

ACKNOWLEDGMENTS

REFERENCES