Methods in Constrained Community Detection: An Integer Optimization Model and Heuristic Approach for Cohort Creation

Danielle Heymann*, Collin Schwantes†, Viveca Pavon-Harr*, Ian McCulloh*§
*Accenture Federal Services, Washington, DC, USA
†EcoHealth Alliance, Washington, DC, USA
§Whiting School of Engineering, Johns Hopkins University, Laurel, MD, USA
Email: *danielle.heymann@accenturefederal.com

***Abstract*—As a result of the COVID-19 pandemic, many organizations and schools have switched to a virtual environment. Recently, as vaccines have become more readily available, organizations and educational institutions have started shifting from virtual environments to physical office spaces and schools. For the highest level of safety and caution with respect to the containment of COVID-19, the shift to in-person interaction requires a thoughtful approach. With the help of an Integer Programming (IP) Optimization model, it is possible to formulate the objective function and constraints to determine a safe way of returning to the office through cohort development. In addition to our IP formulation, we developed a heuristic approximation method. Starting with an initial contact matrix, these methods aim to reduce additional contacts introduced by subgraphs representing the cohorts. These formulations can be generalized to other applications that benefit from constrained community detection.**

***Index Terms*—Optimization, Integer Programming, Approximation Heuristic, Community Detection**

I. Introduction

There is a small selection of community detection algorithms for which the user can select the amount of clusters to be created (which would correspond to cohorts), and an even smaller amount of which are constrained community detection algorithms.This particular case, incorporating capacity constraints, is largely unexplored. Our integer program models and heuristic propose an approach to constrained community detection, enforcing minimum and maximum community size constraints. In comparison to the integer program models developed, the heuristic introduced is much faster and computationally tractable with a small impact in optimality. This has been found to be useful for informing return-to-work cohorts in response to COVID-19 recovery. Moreover, it has broader application as organizations seek to establish hybrid in-person/remote work schedules for teams.

These methods were initially developed as part of Accenture Federal Service's Applied Intelligence Discovery Lab's simulation engine and graphical user interface that allows for integration of interactions at temporal and spatial scales to understand infection risk from COVID-19. The goal was to create a cohort development module to be incorporated in the office simulation as a supplementary hazard control module. The creation of the module led to the development of two Integer Programming (IP) Optimization model and an approximation heuristic. This combination of methods allows for flexibility of resources and time available, as the integer programs require a solver, and the approximation heuristic is open source in nature. This approach also makes it possible to diagnose approximation to optimality.

*https://www.accenture.com/us-en/services/us-federal-government/artificial-intelligence

II. Literature Review

There are related methods and research to the proposed constrained community detection formulations. These papers do not address all of the specific constraints or requirements of the formulation addressed in this paper, but they share mutual principles and concepts.

In Ganji, Bailey, and Stuckey [4], The authors note that constrained community detection is a very challenging problem to solve because existing approaches are not flexible enough to cover a variety of background information types. It is evident that existing methods limit potential problem scopes by only addressing certain categories of constraints [4]. Many times it is difficult to understand how to integrate constraints within the community detection algorithms. For example, modularity can be maximized across communities, or pairwise constraints can be implemented, but the authors do not specifically address constraining the amount of groups or size of groups in the paper [4]. They note that the research uses constraint programming to solve a variety of constrained cases for optimality, and they share that it is possible to further customize the formulation to reflect different specialized propagators for community detection [4].

In Zhong and Ghosh [18], the authors address balanced model-based clustering, a method of generating clusters of balanced sizes, through an iterative two-step optimization

process. The framework consists of iterative model re-estimation and sample re-assignment, with implementation of a balance-constrained approach for sample assignment that leverages a bipartitioning heuristic for reduced computational complexity [18].

Zachary W. [17] presents Zachary's Karate Club Model, an information flow model for contact and fission in small groups, which is based on the Ford-Fulkerson labeling algorithm for maximum flow. The model is based on fission in the community network [17]. The model uses a social network approach and Ford-Fulkerson's labeling algorithm to predict the fission of the community, which results in subgroups that have more internal stability than the group as a whole [17].

McCulloh and Savas [11] compare several network-based community detection approaches to include Newman, Louvain, and k-Truss. A benefit of the k-truss is that it incorporates a hierarchical clustering property that allows for the construction of hierarchical reduced graphs [11].

Girvan and Newman [6] developed a method to detect highly connected communities, built around the idea of using centrality indices to find community boundaries. [6]. Newman [13] further proposes an alternative algorithm to determine community structure on the basis of modularity.

Le, Levina, and Vershynin [10] propose an approach to maximize a function of a network adjacency matrix over discrete labels by projecting the set of labels onto a low-dimension space so that the feasible set is much smaller. [10].

III. Method

A. Integer Optimization Model

An Integer Programming (IP) optimization model is the first method to generate a solution to creating cohorts which minimize additional contacts outside of the initial contact network. When a cohort is created, it is assumed that every individual is now in contact with the members of their cohort. Accordingly, in the context of COVID-19, to reduce potential for spread of disease, the objective of the IP is to create cohorts which limit the creation of additional new connections between individuals who have not been in regular contact. We first developed the IP method due to the accuracy and power of the model and solver combination. Two formulations of IP models are presented, each with its own strengths in terms of efficiency for handling different network sizes of varying connectivity. We developed the models in Python [15] and use the NetworkX package [1], gurobipy package, and Gurobi solver, a powerful mathematical optimization solver [6]. Developing the models with gurobipy allows for integrality constraints and complex conditional statements within the programming of the constraints for extensive flexibility and customization. To solve this problem, the solver uses a combination of Branch-and-Bound algorithm, cutting planes, and heuristics. As the models are programmed as a network and has multi-indexed decision variables, an aggressive presolve is specified to tighten the formulation and reduce the size of the problem ahead of the Branch-and-Bound algorithm [9]. Although this formulation of the IP takes longer to solve for optimality than the developed heuristic approximations, it is a reliable diagnostic measure to analyze the accuracy of the heuristic approach.

B. Constrained Community Detection Heuristic

A heuristic approximation method, which can be categorized to constrained community detection, is the second method to generate a solution to optimal cohort creation within the scope of the problem. We developed this method in R and use the igraph package [3]. Our heuristic uses an iterative approach while keeping constraints intact. While the heuristic executes significantly quicker than the IP optimization model, it is to be noted that the heuristic is an approximation method and does not always achieve optimality. There are further ways to improve the heuristic as the current method does not perform iterative changes in parallel and so reduces the potential for unexplored improvement to the objective value. The paper will delve into this idea in the analysis section of the research.

IV. Integer Programming Optimization Model

A. Formulation

The formulation as an integer program depends on a collection of input variables, parameters, decision variables, constraints, and objective function. We developed and implemented two model formulations, the first with two sets of decision variables that are based off of the incidence function, the second with one set of decision variables that are based on the principles of correlation clustering.

1) Input Variables and Parameters: Let graph $G = (V, E)$ describe the network of individuals who work in the same office. In testing the model, the test graphs are generated using $G(n, p)$ Erdos-Renyi random graphs to represent graph G, with the number of nodes, n, and the probability of edge creation, p. In the real world, an edge would be created between two individuals in the graph if they have interacted in person over a defined period. Accordingly, the input graph, G, would represent the contact network of a set of individuals. In the context of the COVID-19 cohort case, the contact network is a matrix that captures the individuals who are in regular contact with one another. The aim of this cohort model is to reduce unnecessary additional edges introduced to the network after establishing cohorts.

After establishing the input contact network graph structure, model parameters can be defined. The parameter, $total\ cohorts$, represents the user defined number of cohorts in which the set of individuals in the contact network will be partitioned to. For example, if the parameter is set such that $total\ cohorts = 3$, three cohorts will be created that could come in on separate days or separate times of the day. While the cohort group assignments are generated, the

scheduling aspects of the day and time combination is flexible and not addressed in the formulation. The next parameter, $min\ cohort\ size$, is the user defined minimum cohort size. The parameter can be set in a way to ensure a minimum level of social interaction is achieved. The next parameter, $max\ cohort\ size$, is the user defined maximum cohort size. It can also be adjusted to satisfy capacity constraints and social distancing and gathering policy requirements. Different geographical locations will have to consider local gathering restrictions, which can be used to set this parameter.

Next, the adjacency matrix of G, A_G, is generated, where a_{ij} is an element of A_G. If $a_{ij} = 1$, then there is an edge between v_i and v_j. Otherwise, $a_{ij} = 0$. The cost matrix for this formulation is defined as C_G, where if c_{ij} is an element of C_G. If $a_{ij} = 0$, then $c_{ij} = 1$. Otherwise, $c_{ij} = 0$. This cost matrix will be incorporated in the objective function so that extra edges introduced to the solution cohorts will incur a cost.

B. IP Formulation 1: Incidence Function Mapping

The first IP formulation is developed in a way that uses the definition of the incidence function as part of how the decision variables are constrained so that the architecture of the model resembles a network [7]. The two decision variables represent edges and nodes and are constrained in a way that enforces incidence function mapping.

1) Decision Variables: For simplicity, in this formulation the set of nodes that represent individuals in the network will be referred to as N. The set of cohorts will be referred to as C. There are two sets of decision variables for this formulation.

The first decision variable to be introduced, X_{ptc}, is the binary variable which determines the creation of an edge between person p, teammate t, in cohort c.

$$X_{ptc} = \begin{cases} 1 & an\ edge\ exists\ between\ p\ and\ t\ in\ c \\ 0 & otherwise \end{cases}, \quad (1)$$

$$\forall\ \ p \in N,\ t \in N,\ c \in C$$

Next, the decision variable, Y_{pc}, is the binary variable which determines the assignment of person p to cohort c.

$$Y_{pc} = \begin{cases} 1 & person\ p\ is\ assigned\ to\ cohort\ c \\ 0 & otherwise \end{cases}, \quad (2)$$

$$\forall p \in N,\ c \in C$$

2) Constraints: The model is subject to the following constraints:

Cohort Unique Assignment Constraint
A person p can only be assigned to one cohort:

$$\sum_{c \in C} Y_{pc} = 1, \quad \forall\ \ p \in N \quad (3)$$

Conditional Constraints on X_{ptc} and Y_{pc}
Upper and Lower Bounds to establish relationship between edge X_{ptc} and node Y_{pc} existence: The upper bound constraint below forces X_{ptc} to zero, unless both Y_{pc} and Y_{tc} have values of 1. We still require a lower bound to ensure that X_{ptc} will take the value of 1 when both Y_{pc} and Y_{tc} have values of 1.

$$X_{ptc} \leq \frac{Y_{pc} + Y_{tc}}{2}, \quad \forall\ \ p \in N,\ t \in N,\ c \in C \quad (4)$$

The lower bound constraint below forces X_{ptc} to 1 if both Y_{pc} and Y_{tc} have values of 1. Paired with the upper bound constraint, we now have a well-defined relationship between X_{ptc} and Y_{pc} that can be interpreted as a network of nodes and edges in a defined space.

$$X_{ptc} \geq Y_{pc} + Y_{tc}, \quad \forall\ \ p \in N,\ t \in N,\ c \in C \quad (5)$$

Minimum Cohort Size Constraint
Each cohort must achieve the minimum cohort size.

$$\sum_{p \in P} Y_{pc} \geq min\ cohort\ size, \quad \forall\ \ c \in C \quad (6)$$

Maximum Cohort Size Constraint
Each cohort must be within the maximum cohort size.

$$\sum_{p \in P} Y_{pc} \leq max\ cohort\ size, \quad \forall\ \ c \in C \quad (7)$$

Binary Integer Constraints
The IP model is dependent on X_{ptc} and Y_{pc} taking on binary integer values.

$$X_{ptc} \in \{0, 1\}, \quad \forall\ \ p \in N,\ t \in N,\ c \in C \quad (8)$$

$$Y_{pc} \in \{0, 1\}, \quad \forall\ \ p \in N,\ c \in C \quad (9)$$

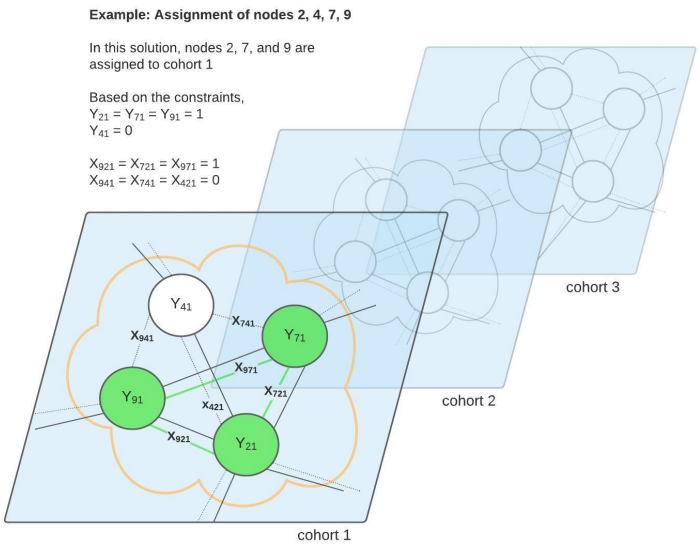

Fig. 1: The example shows the relationship of the decision variables and the incidence function mapping, enforced by the constraints

3) Objective Function: The objective function minimizes the cost associated with introducing new edges, which represent in-person interactions. Only the lower triangular matrix is taken into account to avoid double counting the penalties.

$$minimize \sum_{c \in C} \sum_{t \in P} \sum_{\substack{p \geq T, \\ p \in P}} c_{pt} X_{ptc} \tag{10}$$

C. IP Formulation 2: Correlation Clustering

The formulation uses correlation clustering to establish relationships between cohorts and people. Model constraints enforce the dissociation and association of these entities. In Bansal, Blum, and Chawla (2004), the authors present an approximation method for minimizing disagreements and maximizing agreements for the problem of clustering vertices. The authors also delve into the theory behind the correlation clustering approach that is implemented in the integer program model [2].

1) Decision Variables: Once again, the nodes that represent individuals in the network will be referred to as N. The set of cohorts will be referred to as C. There is only one set of decision variables for the formulation.

The decision variable, Z_{ij}, indicates dissociation and is the binary variable where i and j refer to either cohorts or individuals in the network.

$$Z_{ij} = \begin{cases} 1 & i \, and \, j \, are \, in \, different \, clusters \\ 0 & otherwise \end{cases}, \tag{11}$$

$$\forall \;\; i \in N \cup C, \, j \in N \cup C$$

2) Constraints: The model is subject to the following constraints:

Cohort Unique Assignment Constraint
A person j can only be assigned to one cohort:

$$\sum_{i \in C} (1 - Z_{ij}) = 1, \quad \forall \;\; j \in N \tag{12}$$

Pair Identity Constraint
In the Z matrix, it is assumed that $Z_{ij} = Z_{ji}$:

$$Z_{ij} \; = \; Z_{ji}, \quad \forall \;\; i, \, j \in N \tag{13}$$

Cohort Dissociation Identity Constraint
When looking at Z_{ij}, if $i, j \in C$ are considered, it is assumed that each cohort will be dissociated with another cohort:

$$Z_{ij} \; = 1, \quad \forall \;\; i, \, j \in C \tag{14}$$

Correlation Clustering Constraint
By definition of the decision variable, it is assumed that Z_{ij} takes on the value of zero when i and j are associated. Therefore, if looking at any three entities (cohorts/people) the dissociation (taking value of one) of any 2 needs to be greater than the association (taking value zero) of the other 1:

$$Z_{ij} \; \leq Z_{ik} \; + Z_{kj} \; , \quad \forall \;\; i, \, j, k \in N \cup C \tag{15}$$

Minimum Cohort Size Constraint
Each cohort must achieve the minimum cohort size.

$$\sum_{j \in N} (1 - Z_{ij}) \; \geq \; min \, cohort \, size, \quad \forall \;\; i \in C \tag{16}$$

Maximum Cohort Size Constraint
Each cohort must be within the maximum cohort size.

$$\sum_{j \in N} (1 - Z_{ij}) \; \leq \; max \, cohort \, size, \quad \forall \;\; i \in C \tag{17}$$

Binary Integer Constraints
The IP model is dependent on Z_{ij} taking on binary integer values.

$$Z_{ij} \in \{0, 1\}, \quad \forall \;\; i \in N \cup C, \, j \in N \cup C \tag{18}$$

3) Objective Function: Like the previous formulation, this objective function minimizes the cost associated with introducing new edges, which represent in-person interactions, with the same scoring method. Here the entire matrix is considered, and the denominator prevents double counting the penalties.

$$minimize \sum_{i \in N} \sum_{j \in N} \frac{c_{ij}(1 - Z_{ij})}{2} \tag{19}$$

D. Considerations

These integer programs have many variables and constraints depending on the user defined parameters. Branch-and-Bound can address global optimization for discrete problems that are usually NP-hard, such as the cohort optimization problem, but by relaxing the initial integer constraints and introducing fractional solutions, the tree can become very large as the network size increases [5]. Therefore, Branch-and-Bound, cutting planes, and heuristics that the solver uses can ultimately find an optimal solution, but it may be very time consuming [12]. Additionally, a license for the Gurobi solver is required to carry out the optimization [8].

While running test graphs through the IP models, a time limit of 600 seconds was applied. In the results table, an indication of reaching this cutoff time is noted.

E. Examples

In many cases, there is evidence of multiple optimal solutions. In the example below, the two different IP models generate different cohort assignments with the same optimal objective value of 7. It is due to how the solver explores the Branch-and-Bound tree in a different order based on the architecture of the constraints. Once the Branch-and-Bound algorithm found the objective value of 7, it retains the first cohort assignment that generated that value, despite exploring more nodes with the same objective value.

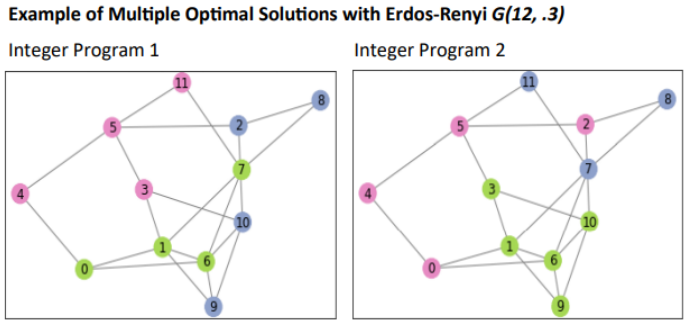

Fig. 2: Different cohort assignments produce same optimal objective value with a different solution set.

F. Analysis

The two proposed integer programming formulations both should hypothetically achieve the optimal objective value if given ample time to run. However, there are certain cases where experimentation reveals that one outperforms the other. Overall, it is evident that higher connectivity within the network reduces the run time for the integer programs. This could be attributed to the abundance of multiple optimal solutions given highly connected communities. When the input network graph is sparser, the number of multiple optimal solutions is less then when the input network graph is denser. It is noted that when the size of the input network graph is scaled larger with higher connectivity between nodes, the integer program formulation based on the incidence function mapping executes quicker than the formulation based on correlation clustering. However, when the same sized graph has lower connectivity between nodes, the formulation based on correlation clustering executes quicker than the formulation based on the incidence function mapping.

V. Constrained Community Detection Heuristic

A. Formulation

The approximation heuristic uses an iterative approach. We developed it in R [14]. The first step involves initialization of the cohorts by segmenting the total nodes nearly evenly (while maintaining integrality) and assigning them based on the those segments to the specified number of cohorts. The arbitrary assignment is the first feasible solution explored and abides by the set minimum and maximum cohort capacity constraints. That is one of the main differences between the integer program and the approximation heuristic; the integer program finds its first solution via a linear relaxation which does not keep integrality intact and so a node may be fractionally part of multiple cohorts, which is not feasible.

Now that there is a data structure populated with each cohort and its set of members, two main functions are executed iteratively until they run consecutively with no improvement to the objective value. The same scoring method is applied to obtain an objective value: each additional edge introduced to the cohort between a set of nodes, which did not exist in the initial contact network, introduces a one point penalty costs. The objective value is the sum of these penalties, so the goal is to minimize this sum.

The two main functions that are part of the iterative approach are a Switch function and a Transfer function. The Switch function does a clean swap of a pair of nodes of different cohorts, but only if the swap improves the objective value. This would mean that if at the beginning of the iteration, if i is initially in $Cohort\ A$ and j is initially in $Cohort\ B$, a swap will be performed if and only if moving i to $Cohort\ B$ and j to $Cohort\ A$ improves the objective value. This is carried out recursively, so that the test pair and test configuration are updated until either all pairs are exhausted or if a stronger objective value is achieved. At either point, the cohort assignments are updated accordingly, and they are passed to the next function, the Transfer function.

The Transfer function operates slightly differently than the Switch function in that capacity constraints need to be enforced, and only one individual is moving out of its cohort to another, again if and only if the move improves the objective value. The first step is to generate lists of both the receive candidate cohorts, and the hand off candidate cohorts, which are cohorts that given capacity constraints can feasibly receive or hand off an individual, respectively. After these lists are created, the heuristic points to the first individual in the first candidate hand off cohort. The individual participates in test and ”travels” to other cohorts until there is either an improvement to the objective value with a new cohort assignment configuration, or if there are no more potential receiving cohorts to test out a new configuration. At this point, if there was an improvement to the objective value, the current cohort assignments are passed through another round of the initialization, Switch, and Transfer functions. If the current individual has not improved the objective value via test transfers to other cohorts, the function is carried out recursively, as the Switch function was carried out, and the test traveling individual and test configurations are updated until either all travelers are exhausted with no improvement to the objective value, or if there was a test configuration that improved the objective value.

In order to exit the iterative process, a cohort configuration must pass through both the Switch function followed by the Transfer function with no improvements to the objective value. The functions achieve a better objective value with this ordering (Switch then Transfer) because the Transfer function constrains the problem when moves occur, and any additional constraint limits the potential for a more optimal solution.

For more detail on the process flows of these functions, in the appendix, Figure 4 shows the high level overview of the heuristic, figure 5 shows the decision made in the switch function, and figure 6 shows the decision made in the transfer function.

B. Considerations

Although the heuristic approach significantly reduces the run-time, there are considerations. The heuristic reaches a local minima, so it is not always optimal. To reach the local minima, the Switch function must go before Transfer function; the transfer function alters current capacity, which further constrains the problem and limits the potential to improve the objective. The updated cohort sizes act as additional restrictions due to less slack in which the nodes can interact during the next update. Therefore, better results are achieved when the Switch function precedes the Transfer function.

C. Examples

To understand the heuristic's performance over varying sizes of the input graph and connectivity, 9 Erdos-Renyi random test graphs that were sent through the heuristic. The capacity constraints were scaled consistently as the input graph grew in size. For example, the acceptable capacity ranges for graphs with 12, 24, and 36 nodes were [3,5], [6,10], and [9,15], respectively. By keeping these scaled, the relationship between connectivity, graph size, and run-time is more clearly highlighted. Running these input graphs through the different proposed methods allows for an analysis and understanding of where the methods have limitations and which characteristics of the initial input contact network impact the run-time most.

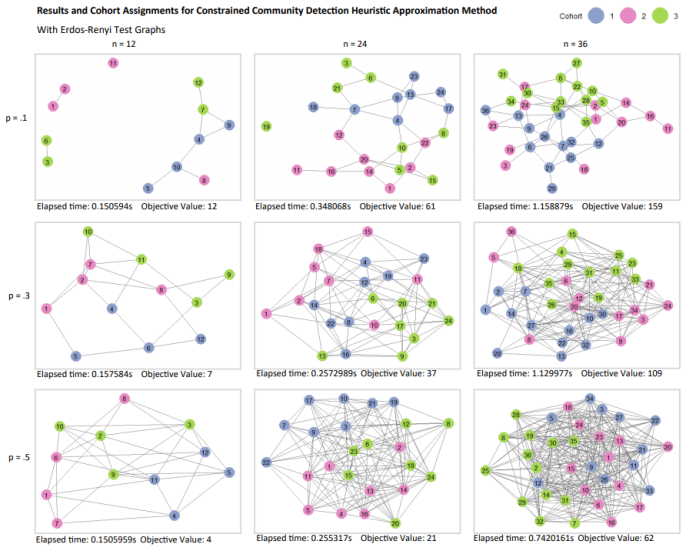

Fig. 3: The grid shows test cohort assignments and performance for the heuristic approximation. The acceptable capacity ranges for graphs with 12, 24, and 36 nodes were [3,5], [6,10], and [9,15], respectively. Erdos-Renyi random test graphs were generated, and the n and p values are indicated on the grid.

VI. Comparison of Methods

A. Analysis and Findings

As mentioned previously, it is observed that the sparser graphs result in longer run-time for all methods due to the smaller amount of multiple optimal solutions. Because the methods all stop once reaching a local or global minima, the graphs with a larger amount of multiple optimal solutions are likely to stop executing iterations of either Branch-and-Bound or the approximation heuristic quicker than the graphs with a smaller amount of multiple optimal solutions. Another note is that the heuristic is significantly quicker in run time than the integer program models. The trade-off between speed and optimality depends on the context of the specific use case and the cost of small changes in the objective value.

Ex.	Erdos-Renyi	IP 1		IP 2		Heuristic	
	$G(n,p)$	Obj. Val.	Time (s)	Obj. Val.	Time (s)	Obj. Val.	Time (s)
1	G(12, .1)	10	0.2499	10	0.4615	12	0.1506
2	G(12, .3)	7	0.1137	7	0.4102	7	0.1576
3	G(12, .5)	2	0.0280	2	0.07086	4	0.1506
4	G(24, .1)	59	67.6019	59	*600	61	0.3481
5	G(24, .3)	34	2.2757	34	74.4558	37	0.2573
6	G(24, .5)	19	0.5345	19	14.2801	21	0.2553
7	G(36, .1)	158	*600	157	*600	159	1.1589
8	G(36, .3)	102	*600	101	*600	109	1.1300
9	G(36, .5)	55	39.3193	56	*600	62	0.7420

*Reached time limit of 600 seconds and terminated solver

B. Future Work

Currently, our heuristic approximation for this problem begins with a feasible yet arbitrary cohort assignment configuration. There are ways to potentially improve the objective value via network flow algorithms such as Ford-Fulkerson, as Zachary's Karate Club does, as a presolve strategy [17], or to leverage k-Truss networks throughout the heuristic [11]. Additionally, other weights can be applied to the cost matrix depending on relationships that a user may want to capture in the network. This would change the objective function. The objective function presented is strictly based on reducing additional connections introduced, which did not exist in the initial input contact network, to the generated cohorts. The objective function can be weighted in a way to capture additional attributes via a weighting function. In the context of COVID-19, vaccination status and individual risk factors could be used to minimize infection risk for vulnerable individuals.

VII. Conclusion

Our methods address an area of constrained community detection with a lack of solutions, yet relevant application, especially in light of the current COVID-19 pandemic. Many organizations have begun to reconnect in person, and the use of a gradual, phased approach can be carried out with a constrained community detection algorithm to help manage the hazard. Current methods address constrained community detection; however, they lack the specific constraints such as minimum size, maximum size, and specified number of groups. The integer programs that we developed are fairly straightforward to implement once the underlying constraints are understood and the cost matrix has been established. Our heuristic method provides a free, lightweight solution for creating cohorts from intrinsic community structure, something that is vitally important as we look to minimize risks during a global pandemic.

VIII. Acknowledgements

Graph Community of Interest at Accenture Federal Services: Heather Patsolic, Marjorie Willner, Evan Williams, Benno Lee, Ben Ortiz, Luke Pascual, Trevor Kent, Matthew Swahn:
For their feedback and ideas, which helped shaped the heuristic approximation method, and for bringing insight on related work to explore

Dr. Anke van Zuylen, Senior Lecturer in Computer Science at Cornell University:
For her perspective of the proposed methods as well as the recommendation and guidance in using a correlation clustering approach for one of the integer program formulations

IX. Appendix

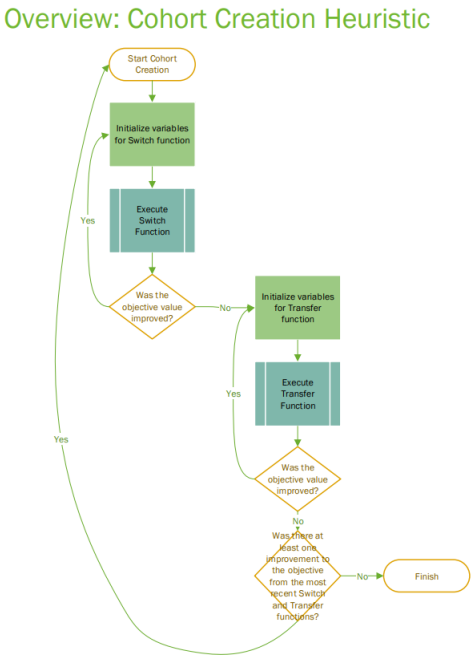

Fig. 4: The flow diagram illustrates the overview of the heuristic.

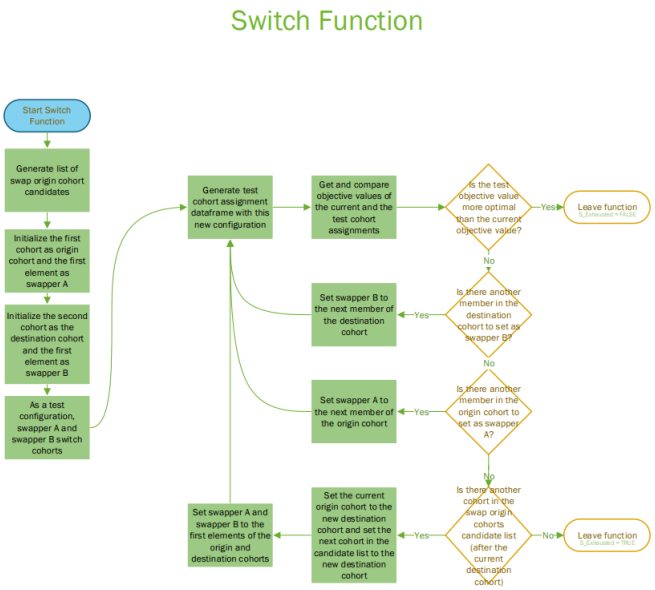

Fig. 5: The flow diagram illustrates the process and decisions that are made to swap elements of cohorts to improve the objective value.

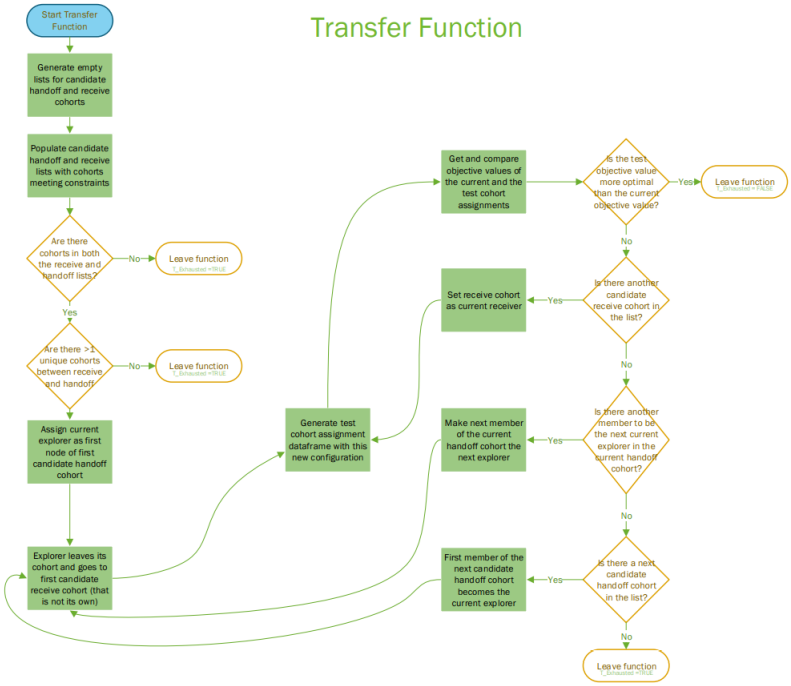

Fig. 6: The flow diagram illustrates the process and decisions that are made to transfer an element of a cohort to another cohort to improve the objective value.

REFERENCES

[1] Aric A. Hagberg, Daniel A. Schult and Pieter J. Swart, "Exploring network structure, dynamics, and function using NetworkX", in Proceedings of the 7th Python in Science Conference (SciPy2008), Gäel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA USA), pp. 11–15, Aug 2008

[2] Bansal, Nikhil, Avrim Blum, and Shuchi Chawla. 2004. "Correlation Clustering." Machine Learning 56 (1-3): 89–113. https://doi.org/10.1023/b:mach.0000033116.57574.95.

[3] Csardi, G., and Nepusz, T. (2006). The igraph software package for complex network research. In InterJournal: Vol. Complex Systems (p. 1695). https://igraph.org

[4] Ganji, Mohadeseh, James Bailey, and Peter J. Stuckey. 2017. "A Declarative Approach to Constrained Community Detection." Lecture Notes in Computer Science, 477–94. https://doi.org/10.1007/978-3-319-66158-2_31.

[5] Gao, Jiyao. 2014. Review of Branch and Bound (BB). https://Optimization.mccormick.northwestern.edu/. 2014. https://optimization.mccormick.northwestern.edu/index.php/Branch_and_bound_(BB).

[6] Girvan, M., and M. E. J. Newman. 2002. "Community Structure in Social and Biological Networks." Proceedings of the National Academy of Sciences 99 (12): 7821–26. https://doi.org/10.1073/pnas.122653799.

[7] "Graphs, Networks, Incidence Matrices." n.d. Accessed July 2, 2021. https://ocw.mit.edu/courses/mathematics/18-06sc-linear-algebra-fall-2011/ax-b-and-the-four-subspaces/graphs-networks-incidence-matrices/MIT18_06SCF11_Ses1.12sum.pdf.

[8] Gurobi Optimization, LLC. 2021. "Gurobi Optimizer Reference Manual." http://www.gurobi.com.

[9] "How Does Presolve Work?" n.d. Gurobi Support Portal. https://support.gurobi.com/hc/en-us/articles/360024738352-How-does-presolve-work-.

[10] Le, Can M., Elizaveta Levina, and Roman Vershynin. 2016. "OPTIMIZATION via LOW-RANK APPROXIMATION for COMMUNITY DETECTION in NETWORKS." The Annals of Statistics 44 (1): 373–400. https://www.jstor.org/stable/43818910.

[11] McCulloh, Ian & Savas, Onur. (2020). k-Truss Network Community Detection. 590-593. 10.1109/ASONAM49781.2020.9381328.

[12] "Mixed-Integer Programming (MIP) - a Primer on the Basics." n.d. Gurobi. https://www.gurobi.com/resource/mip-basics/.

[13] Newman ME. Fast algorithm for detecting community structure in networks. Phys Rev E Stat Nonlin Soft Matter Phys. 2004 Jun;69(6 Pt 2):066133. doi: 10.1103/PhysRevE.69.066133. Epub 2004 Jun 18. PMID: 15244693.

[14] R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.

[15] Van Rossum, G., and Drake, F. L. (2009). Python 3 Reference Manual. Scotts Valley, CA: CreateSpace.

[16] Wikipedia Contributors. 2021. "Zachary's Karate Club." Wikipedia. Wikimedia Foundation. January 14, 2021. https://en.wikipedia.org/wiki/Zachary%27s_karate_club.

[17] Zachary, Wayne W. 1977. "An Information Flow Model for Conflict and Fission in Small Groups." Journal of Anthropological Research 33 (4): 452–73. http://www.jstor.org/stable/3629752.

[18] Zhong, Shi, and Joydeep Ghosh. 2003. "Scalable, Balanced Model-Based Clustering." Proceedings of the 2003 SIAM International Conference on Data Mining, May. https://doi.org/10.1137/1.9781611972733.7.