ASSIGNING SCOUTS TO OPTIMAL PATROLS

PHIL SNYDER, IAN BRUCE, AND JULIO MARCO PINEDA

Abstract. The Scoutmaster wants to find an arrangement of 13 new scouts into two patrols. The cohesiveness and overall quality experience of a patrol is severely affected by adverse relationship and can be improved by existing preferences. Thus, by using anonymous data from scouts and parents, a brute force strategy was employed to find the optimal arrangement of scouts that avoids severe conflict and maximizes existing preferences. An arrangement of 6 and 7 scouts on one patrol respectively was found that satisfies the desired conditions.

Problem Description

Every year the Scoutmaster must assign new scouts to groups called patrols. In these patrols, scouts would plan and perform different scouting activities together in which they can form close friends that grow together. However, in recent years, the number of scouts have been increasing at a rate such that it is no longer logistically viable to place every new scout in a single patrol; multiple patrols are needed to organize activities better and to form proper bonds and community between the new recruits. Furthermore, the new scouts are in the fifth or sixth grade who have not fully developed their interpersonal skills. Many can be temperamental and awkward when dealing with adversity, and even some devastating conflicts occur where not only the two fighting scouts create animosity between each other, the rest of the troop alienates the destructive pair. If enough of these adverse interactions occur within a troop, many scouts would decide to quit and lose out in a great experience and community in the Boy Scouts.

The data the scoutmaster collects about adverse pairings and preferences are either privately given by the scouts themselves or by their parents. The possibly destructive interactions are given by the parents of the scouts.Previously, the Scoutmaster was able to determine the perfect patrols manually due to having smaller number of scouts. However, the number of scouts have been increasing yearly to the point that solving this grouping problem cannot be done by hand in a reasonable time. Therefore, we can help our community partner find a better means to solve his troop organization problem. Our goal for this project is to assign boy scouts in different patrols of appropriate size (of 6-8 scouts) such that we maximize the retention rate of the scouts in the program by avoiding severe conflicts while maximizing positive relationships between scouts.

We would like to answer the direct problem our community partner provided of arranging 15 scouts into two troops that avoids severe conflicts and promotes the positive relationships between the scouts. Furthermore, we want to determine if we can build a model to predict these patrol arrangements for any number of scouts so that the Scoutmaster can use this model for future years. Other than improving the overall experience of the scouts and alleviating the burden of the Scoutmaster,

this model can possibly be applied to other situations where arranging a number of people of groups where interpersonal dynamics is necessary and important.

Simplifications

Ultimately, our goal is to find groups for the scouts that will allow them to grow and be happy in the troop. If we really wanted to fully understand how a group can foster growth in an individual, we would want to consider the astronomical number of aspects about the personalities of each member and how those influence each of the other members given those other members astronomically complex personalities. Sociological and psychological literature would need to be studied extensively to learn about these aspects, and experts in the field would most likely be required to collect relevant data for each child, taking several weeks to conduct interviews. At the of the day, given our limited knowledge in this area, we settled with considering a subset of the sociological data between the scouts. We asked each scout which other scouts they liked and which ones they disliked. Now, this is obviously a giant simplification, but the difficulty of gathering more complex information and compiling said information would be too great for the expertise level of anyone involved in solving this problem. Also, this process of gathering data would have to be repeated each year for each new batch of new scouts, and would therefore be a burden on the Scoutmaster to collect a lot of data.

Another simplification is to assume that the quality of a group can be determined by the quality of the relationships between the unique pairs of the group - in other words, the whole is equal to the sum of the parts. For example, consider the case where scouts A, B and C individually like each other when they are alone with one other scout, but dont like being in a group together. This epiphenomenon wont be considered; in context of our model, the pairing of these scouts would be heavily favored.

Mathematical Model

To model the relationships between scouts, we use a mathematical graph. We let the scouts be vertices in our graph. A directed edge between Scout A and Scout B has a weight proportional to how much Scout A likes Scout B. We would then like to find a partition of the vertices in the graph that minimizes an objective function given constraints on the sizes of subsets generated by our partition. This results in a complete graph on 13 vertices having edge weights 0, 1, or -1, corresponding to indifference, approval, or disapproval, respectively. Scoutmaster Bruce specifically would like two patrols from the 13 scouts - one of size 6, the other of size 7. The total number of partitions given these constrants is $\binom{13}{6} = \binom{13}{7} = 1716$. This number is reasonable enough that we may calculate the total loss of every valid grouping and select the partition with the lowest score.

We measure the goodness (or score) of a partition in six different ways, corresponding to six different objective functions, which we dub MinCut, FriendCut, EnemyCut, AwkwardCut, and HybridCut. MinCut is designed to minimize the cut edge weights between partitions (see *Solution of Mathematical Problem*). FriendCut is designed to maximize the number of intra-patrol positive edge weights. EnemyCut is meant to minimize the number of intra-patrol negative edge weights, Awkward cut $(+i)$ is the same as MinCut, but with an additional penalty of i if there are a pair of scouts, Scout A and Scout B, such that Scout A favors Scout

B, but Scout B disfavors Scout A. Hybrid Cut is a convex combination of MinCut and FriendCut, each with equal weight (that is, half the MinCut loss plus half the FriendCut loss). Our three primary objective functions - MinCut, FriendCut, and EnemyCut are given a more rigorous treatment in the next section.

Solution of Mathematical Problem

We represent our directed graph as a 13 by 13 similarity matrix, or, as might be more fitting for the problem, an "affinity" matrix. Like below[1]:

1	0	1	-1	0	0	0	0	0	1	-1	0	0	1
2	0	0	-1	1	1	0	0	0	0	0	0	0	1
3	0	0	0	0	0	-1	0	1	0	0	0	0	0
4	0	0	0	0	0	0	0	0	NA	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	1	1	0	0	1	0	0	0	0	0
7	0	1	0	1	0	0	0	0	0	0	0	0	0
8	0	0	0	0	1	1	0	0	0	0	1	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	1	0	0	0	0	0	0	0	0
11	0	0	0	0	1	0	0	1	0	0	0	1	0
12	0	0	0	0	1	0	0	0	0	1	0	0	1
13	-1	1	0	1	0	0	0	0	-1	0	0	1	0

The NA value represents a hard constraint established by a scout's parent (i.e., under no circumstances are these two to be in the same patrol). We designed a number of objective functions that accept two scouts and calculates the loss associated with such a placement. The loss is defined to be $L(x, y) = \max(-2, x[y] + y[x])$ where $x[y]$ is x's disposition towards y and vise versa. For example, if x dislikes y and y dislikes x, the loss is -2. If x likes y but y dislikes x the loss is 0. If x and y both like each other, the loss is 2. In mixed cases where the first scout is indifferent to another, but the other likes or dislikes the first scout, the loss is +1 or -1 respectively. We are trying to minimize the loss, so in applying our loss function over the entire dataset, our algorithm is equally averse to same-patrol mutual enmity as it is to dividing mutual friendship, half as averse to the indifference of one scout and the stronger inclination of another, and indifferent to polar opposite affinities within the same patrol. There is a question as to whether this is an accurate reflection of how individual scouts would view their own "loss" of any of the possible scenarios. As it turns out, our algorithm arrives at a very reasonable solution regardless.

Results

As mentioned earlier, the number of scouts we must partition is small enough to find an optimal solution by brute force. Our algorithm finds a partition of scouts into patrols of sizes 6 and 7 which is optimal with respect to each objective.

[1]The scouts were enumerated like so: 1. Brandon, 2. Cameron, 3, Christian, 4. Colby, 5. Daniel, 6. Darwin, 7. Evan, 8. Jake, 9. Jordan, 10. Nathan, 11. Patrick, 12. Timmy, 13. Tommy.

MinCut		**FriendCut**		**EnemyCut**	
Patrol 1	Patrol 2	Patrol 1	Patrol 2	Patrol 1	Patrol 2
Brandon	Christian	Brandon	Daniel	Brandon	Christian
Cameron	Daniel	Cameron	Darwin	Cameron	Jake
Colby	Jake	Christian	Jake	Colby	Jordan
Darwin	Jordan	Colby	Jordan	Daniel	Nathan
Evan	Nathan	Evan	Nathan	Darwin	Patrick
Tommy	Patrick	Tommy	Patrick	Evan	Timmy
	Timmy		Timmy	Tommy	
AwkwardCut (+1)		**AwkwardCut (+2)**		**HybridCut**	
Patrol 1	Patrol 2	Patrol 1	Patrol 2	Patrol 1	Patrol 2
Brandon	Christian	Brandon	Cameron	Brandon	Christian
Cameron	Daniel	Christian	Evan	Cameron	Daniel
Colby	Jake	Daniel	Evan	Colby	Darwin
Darwin	Jordan	Darwin	Nathan	Evan	Jake
Evan	Nathan	Jake	Timmy	Timmy	Jordan
Tommy	Patrick	Jordan	Tommy	Tommy	Nathan
	Timmy	Patrick			Patrick

FIGURE 1. The partitions arrived at by minimizing each objective function

For each method we define three scoring metrics:

(1) **MinCut**. This is the traditional metric of a cut on a weighted graph and is the sum of the weights of the edges we would "cut" if we were to sever the edges between distinct patrols. More formally, for some partition I, J $I \neq J$, the MinCut is defined as:

$$MinCut(I, J) := \sum_{e=(i,j), i \in I, j \in J} c(e)$$

We want this to be as low as possible.

(2) **FriendCut**. This is the sum of the positive, or friendly edge weights within each patrol. That is, if $I_+ = \{e = (i, j) | i, j \in I, c(e) = 1\}$ is the the set of edges in patrol I with positive weight and $J_+ = \{e = (i, j) | i, j \in J, c(e) = 1\}$ is the set of edges in patrol J with positive weight, then (assuming all positive weights are 1):

$$FriendCut(I, J) := |I_+ \cup J_+|$$

We want this to be as high as possible.

(3) **EnemyCut**. This is similar to FriendCut, except we now measure the number of negative edges going from one scout to another within the same patrol. Let $I_- = \{e = (i, j) | i, j \in I, c(e) = -1\}$ be the the set of edges in patrol I with negative weight and $J_- = \{e = (i, j) | i, j \in J, c(e) = -1\}$ be the set of edges in patrol J with negative weight, then

$$EnemyCut(I, J) := |I_- \cup J_-|$$

We want this to be as low as possible.

The other methods are modifications of these three and are defined in Appendix A.

Method	MinCut	FriendCut	EnemyCut
MinCut	0	17	0
FriendCut	1	18	1
EnemyCut	2	15	0
AwkwardCut (+1)	0	17	0
AwkwardCut (+2)	1	18	2
HybridCut	0	18	1

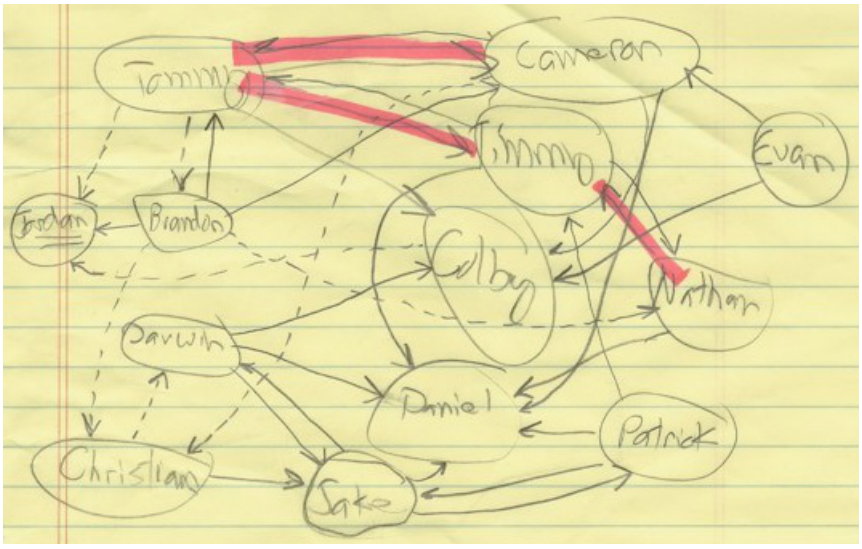
FIGURE 2. The scores according to the three primary metrics.

Of the objective functions we chose to use, it seems like MinCut and HybridCut are the best performing. FriendCut performs almost as well as HybridCut, except for one worse in the EnemyCut objective. EnemyCut does poorly since finding a partition with 0 intra-patrol negative weight edges is easy in this example, and so EnemyCut settles for the first suboptimal (in terms of the other objective functions) solution it finds. AwkwardCut (+1) arrived at the same partition as MinCut (not altogether surprising, but informative, since even with the additional +1 penalty for placing Brandon and Tommy in the same patrol it decided it would be best off to do so anyways). But changing the "awkward" penalty to +2, the same penalty as mutual dislike in EnemyCut, we get a completely different partition.

Let's take a more critical look at the optimal partition arrived at via MinCut:

TABLE 1

Patrol 1	Patrol 2
Brandon, Cameron, Colby, Darwin, Evan, Tommy	Christian, Daniel, Jake, Jordan, Nathan, Patrick, Timmy



(1) Tommy and Timmy both like each other and are in separate patrols. But if we move Timmy to Patrol 1 ($P1$), we break up the mutual friendship of Timmy and Nathan in $P2$. Suppose we moved both Timmy and Nathan to $P1$ and Brandon (who dislikes Nathan) to $P2$. But then Brandon loses two friends (Tommy and Cameron), gains one (Jordan), and now neither Nathan nor Timmy are in the same patrol as Daniel, whom they both like. Although, Brandon and Tommy find themselves in the awkward situation of having complete opposite sentiments towards each other. Perhaps this would be a worthwhile trade after all (and a larger, negative weight needs to be placed upon disfavor-favor relationships).
(2) As mentioned in the previous point, Tommy dislikes Brandon. Trading Tommy in $P1$ for Jordan in $P2$ seems promising, But we have overlooked the fact that Colby and Jordan are our NA pair, and cannot be in the same patrol under any circumstances. Trading Colby for Jordan is another option, but Colby will be missed by Darwin, Evan, Cameron and Tommy (In other words, the whole of P1 minus Brandon).

On the whole, though, there are no obvious improvements that can be made to our algorithm's partition. We find that this is an assignment that could have reasonably been arrived at by Scoutmaster Bruce (granted the group dynamics have not changed since the scouts were surveyed). Furthermore, the solution took only 10 seconds to arrive at on a single core machine.

Improvements

While our algorithm demonstrably works well on smaller groups of 13 people, we do not expect our brute force solution to be tractable on larger groups wherein greater than 2 partitions are required. As a possible scenario, consider a company of 150 people to be divided into fixed-size teams of 10 individuals each. There are $150!/(10!)^{15}$ possible partitions, a number proportional to 10^{164} - meaning we could instead use that computational time to enumerate the number of atoms in the universe a tredecillion (10^{78}) times. Knowing this, perhaps it wont surprise the reader that this particular graph partitioning problem in general graphs is NP-complete - though there do exist approximation algorithms.[2]As scout enrollment increases, it becomes necessary to consider scenarios where we have both a large number of scouts in need of assignment and multiple patrols to choose from. Our algorithm could be extended to use brute force to find a globally optimal solution when it is deemed computationally feasible and to use an approximating algorithm to find a locally optimal solution otherwise.

Conclusions

We initially struggled to decide on determining an appropriate objective function to partition the scouts to different groups. Follow up interviews with Scoutmaster Gene Bruce allowed us to decide on this function. We learned to rely on his expertise and intuition rather than trying to blindly decide on which types of relationships should be fostered and avoided to weight our objective function. Thus, we learned necessary communication skills and consideration of our community partner's needs.

[2]B. W. Kernighan, S. Lin, *An Efficient Heuristic Procedure for Partitioning Graphs.* Bell System Technical Journal. **49** (1970), 291–307. doi: 10.1002/j.1538-7305.1970.tb01770.x

With a feasibility check of how many solutions possible once the mathematical model was decided using graphs, a brute force strategy was the most accessible method to arrive at a solution. Thus we were able to develop a program that can partition scouts using this strategy. We learned to utilize tools we have at hand right away, and then we can consider other methods later on to reduce the computational time of our algorithm.

Acknowledgments

We would like to thank Scoutmaster Gene Bruce for providing us the necessary data and being extremely patient and understanding when discussing the background and problem. We would also like to thank Professor Sarah Billey for the mentorship and guidance she provided throughout this project, and our TA Austin Tran for his comments and suggestions.

Verification Statement

(Once we receive our verification statement, we will place it here.)

Appendix A

The other objective functions

(1) **AwkwardCut (+i)**. Same in all respects to MinCut, except for an additional penalty of i if there are two scouts, Scout A and Scout B, in the same patrol such that Scout A likes Scout B and Scout B dislikes Scout A.

$$AwkwardCut(I, J, i) := \left[\sum_{e=(k,j), k \in I, j \in J} c(e) \right] + i * |\{e_+ = (k, j), e_- = (j, k) : e_+, e_- \in I, c(e_+) = 1, c(e_-) = -1\}|$$

(2) **HybridCut**. A convex combination of MinCut and FriendCut, each with equal weight (that is, half the MinCut loss plus half the FriendCut loss).

$$HybridCut(I, J) := \frac{1}{2} MinCut(I, J) + \frac{1}{2} FriendCut(I, J)$$

References

1. B. W. Kernighan, S. Lin, *An Efficient Heuristic Procedure for Partitioning Graphs.* Bell System Technical Journal. **49** (1970), 291–307. doi: 10.1002/j.1538-7305.1970.tb01770.x