CS170 Project Design Doc

Ruihan Guan 3033058043, Ke Huang 3032741012, Yin Tang 3032679379

Main Idea:

Our proposed algorithm will evaluate each person and each friendship with a heuristic based on number their mutual friends and impact of rowdy groups they are in, then greedily place as many highly valued friendships in a bus as possible.

Step Breakdown:

1. Assign a non-negative, numeric value to all friendships.
   1) Do a 2-step BFS from every node to calculate the count of mutual friends between each pair of friends. Assign the count as the weight of the edge between these nodes.

   2) Discard rowdy groups with size larger than bus capacity. Then for every rowdy group R, decrease the weight of edge $(u, v)|u, v \in R$ by factor of $f(u, v) = \frac{1}{(|R|-1)^p}$. Let p be 2 for now. This models the fact that larger rowdy groups are less likely to be actually grouped, and thus have less impact. If u, v are not friends, create a new edge (u, v) of weight 0 first.

2. Greedily pick the best valued friendships.
   1) Put all edges into a max PQ. Calculate the average edge weight.

   2) Pop the edge (u, v) with highest weight. Place them in the same bus. Then for each of their mutual friends w_i, put them into the bus in the decreasing order of $t_i = \max((u, w_i), (v, w_i))$, skip if (w_i, x) is negative for any x in the bus already, or stop if $t \leq 0$ or bus is full. Then repeat by adding mutual friends of w_i and u or v with similar logic... repeat until the bus is full.
   Create G' with assigned nodes and incident edges removed. Repeat for the next bus with G'.

Analysis:

The proposed algorithm considers mutual friends and takes into account the "impact" of rowdy groups relative to size. However it doesn't approximate the problem well when large rowdy groups collide with friend circles. To account for this, we might introduce dynamic edge weights, where (u, v) in rowdy group R has pointer to all other edges $(x, y)|x, y \in R$, and change the weights of other edges when added or broken.