CNI Hackathon – Swabs2Labs: Allocation Track Solution Report

Team Mallocators

Arpan Dasgupta, Kunal Jain, Nikhil Chandak, Shashwat Goel

1 Introduction

1.1 Abstract

A coordinated distribution of COVID samples within a state can increase the testing throughput while maintaining feasible costs in an already stressed economy. Notably, within the dataset provided, the landscape for optimization per day is in the order of crores (INR). We propose a Mixed Integer Programming (MIP) model for inter-district transfers of samples that can provide fast high-quality solutions (within 1% of lower-bound on optimal in 10 minutes). If run for longer, it can provably obtain optimal solutions within the cost delta of cluster selection (which we show is in the order of lakhs, or $\leq 0.2\%$ of total cost) for the provided data.

We further demonstrate that our solution scales to critical situations like a much higher sample-load than capacity (congestion) or a worst-case distribution of samples (no samples in the hub - Bangalore Urban). Further, our model can efficiently handle a large magnitude of samples/capacity, and will also work under a significant increase in number of labs as well.

1.2 Statement Interpretation

There are 30 districts in the state of Karnataka. This number is likely to remain static, though our model can scale to a larger number of such sub-regions. Each district accumulates all it's swabs (samples) at a headquarters (HQ), the coordinates of which are provided in input. These samples are then distributed to 'labs' within the district or elsewhere. With a high penalty (10,000), samples can be kept as backlog at the HQ to be processed the next day. We thus optimize on a day-to-day basis. We assume that given the backlog penalty, the present day's solution becomes independent of other days.

There are 86 labs, one of two 2 types: private (cost per sample processed: 1600) or public (cost per sample processed: 800). The number and type of labs can change (and our model can scale to these), but does so less frequently. Each lab has a fixed per-day capacity, above which only the district where the lab exists can overload upto 100 samples (with a penalty of 5000 per sample overloaded). Labs may have some backlog from previous days, provided in the input. Note that these backlog samples effectively reduce the capacity of the lab for the day, they cannot be transported elsewhere.

We have to optimize at a macro-level, without caring about small details like road infrastructure. With this in mind, transport costs to labs within the district (from the HQ) can be ignored. Similarly, when transferring samples to another district, the euclidean distance to a 'centroid' of a chosen cluster of labs (provided any pair of labs in the cluster is within 40km of each other) can be taken as a fair approximation of transport costs (when multiplied by 1000 per km). Moreover, it is assumed samples leave on a single vehicle from the district, so a district sends samples to a single cluster and the transport cost is independent of the number of samples shifted.

2 Integer Programming Formulation

As the number of districts and labs are small, we are motivated to use exact methods for optimal allocation of swabs. We first propose a MIP formulation for a simpler problem of only intra-district transfers and eventually develop the final model through the introduction of pre-computed clusters.

2.1 Intra District Model

Let us first simplify the problem and assume no outside district transfers (to external labs) are allowed. Further, let d_i be the demand of i^{th} district (ie. the amount of samples generated from the district on the current day) and cap_j be the (testing) capacity of lab j. Let $x_{i,j}$ be the number of samples sent from i^{th} district to j^{th} lab such that the lab j is inside the i^{th} district (since only intra-district transfers are assumed for now).

Let o_j be the number of samples that end up being overloaded in lab j. Then,

$$o_j = min\Big(100,\ max\Big(\ \sum_i x_{i,j}\ - cap_j,\ 0\Big)\ \Big)$$

Since $o_j \in [0, 100]$, so the outer min constraint above can be relaxed as our overall objective also needs to be minimized. While we want $o_j = \sum_i x_{i,j} - cap_j$ when $x_{i,j} \geq cap_j$, we can simply keep $o_j \geq \sum_i x_{i,j} - cap_j$ along with $o_j \geq 0$ due to minimization of objective function again. For similar reasons, we don't need the upper bound on o_j as $\sum_i x_{i,j} \leq cap_j + 100$ needs to be satisfied as well due to capacity and overload constraints.

Thus, the formulation for this simplified model can be stated in the following way:

$$min_{\ x,o}\ \Big(\ \sum_i \sum_j X_j \cdot x_{ij} + \sum_j 5000 \cdot o_j + \sum_i 10000 \cdot (d_i - \sum_j x_{ij})\ \Big)$$

$$\sum_j x_{ij} \leq d_i \qquad \forall\ i \tag{1}$$

$$\sum_i x_{ij} \leq cap_j + 100 \qquad \forall\ j \tag{2}$$

$$\sum_i x_{i,j} - cap_j \leq o_j \qquad \forall\ j \tag{3}$$

$$x_{ij}, o_j \in \mathbb{Z}_{\geq 0}$$

$$X_j \in \{800, 1600\} \text{ depending upon the type of lab } j$$

2.2 Introducing Centroids in Model

While the above model is nice and simple, the actual problem is non-trivial requiring to send samples to external labs while also satisfying the pairwise $40km$ constraint. We can interpret the concept of centroid as an intermediate center between districts and external labs to facilitate distribution of incoming samples among the labs while also saving transportation cost.

Centroids naturally emerge once we have chosen which external labs to send samples. However, we propose a backward formulation and first prepare clusters later on choosing which external labs should a district send its samples through a centroid.

Let us assume a list of clusters L is available such that L_c is a set of labs whose centroid is c and pairwise distance among the labs in L_c is ≤ 40 kms. Also, let $dist(i, j)$ denote the distance from i^{th} district to j^{th} centroid.

2.3 Final Model

With each centroid c acting as an intermediate center where samples are stored before sending to its associated labs, each district can send samples to the labs in its own district or to *one* these clusters but not directly to any labs outside the district.

Let $p_{i,j}$ be the number of samples coming from i^{th} district to j^{th} centroid and let $q_{j,k}$ be the no. of samples going from j^{th} centroid to the k^{th} lab, assuming (j, k) is a valid pair, such as lab k lies in L_j. Also, let $z_{i,j}$ denote the *binary* variable which is 0 if district i sends no samples to centroid j and 1 otherwise. In other words,

$$z_{i,j} = \ min(1, \ p_{i,j}) \ = \begin{cases} 1, & p_{i,j} > 0 \\ 0, & p_{i,j} \leq 0 \end{cases}$$

As there is no restriction on batch size, so a district will send all the samples to a centroid *at once*. If we pre-caclulate the distance between each such pair of district i and centroid j, then $1000 \cdot z_{i,j} \cdot dist(i, j)$ shall perfectly model the objective function taking into account the cost of transportation if district i is sending a batch, and no cost if no sample is being sent by district i to centroid j.

Let $\lambda =$ very large number. As $z_{i,j}$ is an integer, its function can be modelled in the follwing way:

$$\lambda \cdot z_{i,j} \geq p_{i,j} \tag{6}$$

While a district can send its samples to multiple external labs, in this formulation it can only send to any one of the centroids (at max) since centroids by definition act as distribution centers for multiple labs. Thus, another constraint for districts sending samples to centroids is

$$\sum_j z_{ij} \leq 1 \tag{7}$$

When a centroid receives samples from a district, it needs to distribute them to each of its labs. In other words, each of its lab must receive at least 1 sample from the district since keeping one of the lab empty shall change the location of the centroid to which the samples would be sent (and thus, changing the cost of transportation). As $z_{i,j}$ denotes whether district i is sending any samples to centroid j or not, $\sum_i z_{i,j}$ would count the number of district from where samples are coming to centroid j. Since each lab of a centroid should receive at least 1 sample from all the districts which are sending samples to its centroid, thus the following constraint should also be satisfied:

$$q_{kj} \ \geq \ \sum_i z_{ij} \qquad \forall \text{ valid } (i, j, k) \text{ triplets} \tag{8}$$

Hence, the final Mixed-Integer Program in its entirety is stated on the next page.

$$min\Big(\sum_j X_j \cdot \Big(\sum_i x_{ij} + \sum_k q_{kj}\Big) + \sum_j 5000 \cdot o_j + \sum_{i,c} 1000 \cdot dist(i,c) \cdot z_{ic} + \sum_i 10000 \cdot \Big(d_i - \sum_j x_{ij} - \sum_c p_{ic}\Big)\Big)$$

$$\sum_j x_{ij} + \sum_c p_{ic} \leq d_i \qquad \forall\ i \tag{1}$$

$$\sum_j q_{cj} = \sum_i p_{ic} \qquad \forall\ c \tag{2}$$

$$\sum_k q_{kj} \leq cap_j \qquad \forall\ j \tag{3}$$

$$\sum_i x_{ij} + \sum_k q_{kj} \leq cap_j + 100 \qquad \forall\ j \tag{4}$$

$$\sum_i x_{ij} + \sum_k q_{kj} - cap_j \leq o_j \qquad \forall\ j \tag{5}$$

$$\lambda \cdot z_{i,j} \geq p_{i,j} \qquad \forall\ (i,j) \tag{6}$$

$$\sum_j z_{ij} \leq 1 \qquad \forall\ i \tag{7}$$

$$q_{j,k} \geq \sum_i z_{i,j} \qquad \forall\ (j,k) \tag{8}$$

$$x, p, q, z, o \in \mathbb{Z}_{\geq 0}$$

$$X_j \in \{800, 1600\} \text{ depending upon the type of lab } j$$

2.4 Efficient Computation of Clusters

The formulation shown above is theoretically optimal if the solver is provided with all possible (valid) clusters. Unfortunately, it is not possible to generate all the clusters as they grow exponentially with the number of labs, and the MIP model would take too much time to run. Thus the MIP model is treated as a function of not only the district and lab inputs, but also the clusters provided. For a particular set of clusters to work with, it provides the optimal solution.

We present two broad approaches to compute clusters, and synthesize both of them as our final approach for this dataset. We further narrow down these clusters as all can't be provided as input to MIP, covered in the last subsection.

2.4.1 Clusterability-Graph

We can model a graph from the data whose vertices are labs, with edges being added between any two labs which satisfy the pairwise distance constraint. Let $G_c(V, E)$ be our "clusterability-graph" where V is the set of labs and $e(u, v) \in E$ if $dist(u, v) \leq 40km$. When a district sends it samples to clusters, all of its labs need to be *close* which translates to the cluster being a **clique** in G_c. Computing all maximum cliques, and further running our MIP model on them is infeasible for general graphs because the number of maximal cliques is upperbounded by $3^{n/3}$ [Moon & Moser, 1965]. Despite this, we come up with efficient techniques to generate and select high-quality clusters by exploiting the structure of the problem and given datasets. Notice that we require atleast a clique cover, so that each lab is accessible in some way to a district.

In the given datasets, there are only 86 labs. Bangalore Urban has 34 of these labs and they form a clique in G_c. All subsets of these 34 labs consequently also form a clique. This means that within Bangalore Urban, clique selection can be an important factor in minimizing loss, especially in cases where a lot of external samples come to the district. The other labs are distributed over a large area, and hence the G_c is sparse with only small cliques. These can be computed using backtracking search.

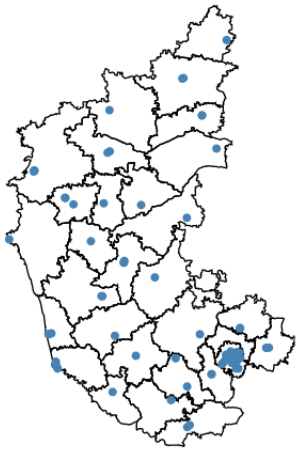


Figure 1: Distribution of labs

2.4.2 Spatial Clusters

As we're dealing with a euclidean planar graph, we can exploit spatial properties. Particularly, one approach to compute a large number of clusters can be to divide the map into a grid of $M = RC$ intersections, where R is the number of rows and C is the number of columns. Then, considering each intersection as a centroid, we create a cluster from labs within $40km$ and add it to a set of clusters S_c. Clearly, $S_c \leq M$ and we can reduce M further based on the spatial properties of the lab distribution. Depending on the dimensions of the map itself and R, C the distance between two intersections can change. Here, Karnataka has dimensions roughly 750km x 400km. With a 400x200 grid, within a 2km delta most clusters would get generated in a fairly efficient way. However, this scales differently to different maps, and does have redundant computation in cases of a sparse distribution of labs (as in the given dataset).

2.4.3 Dataset-specific Combined Approach

Now that we know G_c is sparse and has certain spatial properties (a large clique in Bangalore urban and sparse elsewhere), we can do the following: Run a Depth-first-search on G_c to obtain all connected components. Set aside all labs in Bangalore Urban as a clique and run backtracking search for cliques on the other (small) components.

This approach manages to produce all maximal cliques in the given dataset. Moreover, a large number of smaller cliques (here, all of size ≤ 4) are produced, then some selected are selected as described in 2.4.4, and provided as input to MIP. Even on larger datasets, running backtracking search in each connected component can be far cheaper than the whole graph if similar sparsity is present. In such cases it can be used to produce all cliques of size $\leq k_i$ in connected component i where k_i depends on the size of the components themselves.

2.4.4 Greedy Selection and Random Sampling of clusters for input

As the number of these computed clusters is very large, we reduce the count by sorting clusters in descending order of total capacity. We chose the top ones in a way such that a single lab is not present in too many clusters, and each lab is in at least one cluster (clique cover). Now, hundreds of clusters are selected overall, but many of these would be redundant as the same lab can occur in many clusters.

While the solver can handle such input size, it takes too much time to converge to the provably optimal answer for the input. To ease this, for each district we randomly shuffle these clusters and choose the one which has maximum number of labs uncovered till now, repeating as long as there remains a lab unconnected to this district through some cluster. By sampling randomly, we avoid picking the same cluster continuously whose purpose can be fulfilled by some other clusters thus, achieving uniformity across all instead of congesting among a few.

3 Heuristics

3.1 Greedy Algorithm

Many discrete optimisation problems often have a good solution which can be achieved by some sort of greedy algorithm. We propose one such heuristic below:

Let us perform allocation using simple observations such as -

- Public labs are preferable to fill as compared to private labs.
- Districts with more unallocatd swabs being assigned to labs outside the district might be more favourable.
- Overloading labs inside the district is not favourable as compared to sending them outside the district for testing.
- A district being given backlog is generally the worst case.

Thus, the overall algorithm we have come up with shall assign the swabs as follows -

1. For each district, first assign as many swabs as possible to public labs, then assign as many as possible to private labs.
2. Find districts for which swabs are not allocated and sort them in decreasing order of unallocated swabs.
3. Traverse the districts in this order and for each district :
    - Allocate swabs to the closest possible labs outside the district.
    - If there are remaining swabs, allocate them inside the district by overload these labs.
    - If there are are still unallocated swabs, add them to backlog.

While this greedy algorithm looks simple, it performs well on the sample test data with scores better than the model solution, but worse than that of the MIP solution.

4 Performance Evaluation

4.1 Computing a lower-bound

While optimizing, having a provable lower bound on the optimum cost for each test file can provide an idea on how much scope their is to improve further. Through such bounds, we also prove that our solutions are near-optimal.

4.1.1 Trivial lower bound

One simple method to obtain a loose lower bound on the given datasets is by relaxing transportation and cluster constraints. We perform the following steps as long as samples remain:

1. Try to fill all government labs.
2. Try to fill all private labs.
3. Overload as many government labs as possible.
4. Overload as many private labs, as possible.
5. Backlog samples at district HQ, if required.

While this way of calculating bound may look loose, we demonstrate that our solution comes with 3% of this bound (which is lower than the actual answer) across all datasets.

4.1.2 Tighter bound

Notice that for a given input of cluster, the MIP model provides an optimum solution. Therefore, any difference from the optimum would be due to not providing the right clusters. We can't provide all clusters due to computational constraints, but we can provide an upper-bound on loss due to suboptimal cluster selection, and hence a lower-bound on the optimum cost. This we do by picking cliques based on Section 2.4, running the MIP model with these clusters as input, and then subtract from the output cost: $1000 \cdot$ ("diameter" of cluster) for each cluster that is chosen by the model to send external samples. Here diameter refers to the maximum pairwise distance between any 2 labs in a cluster.

The rationale and proof outline for this is provided in the appendix.

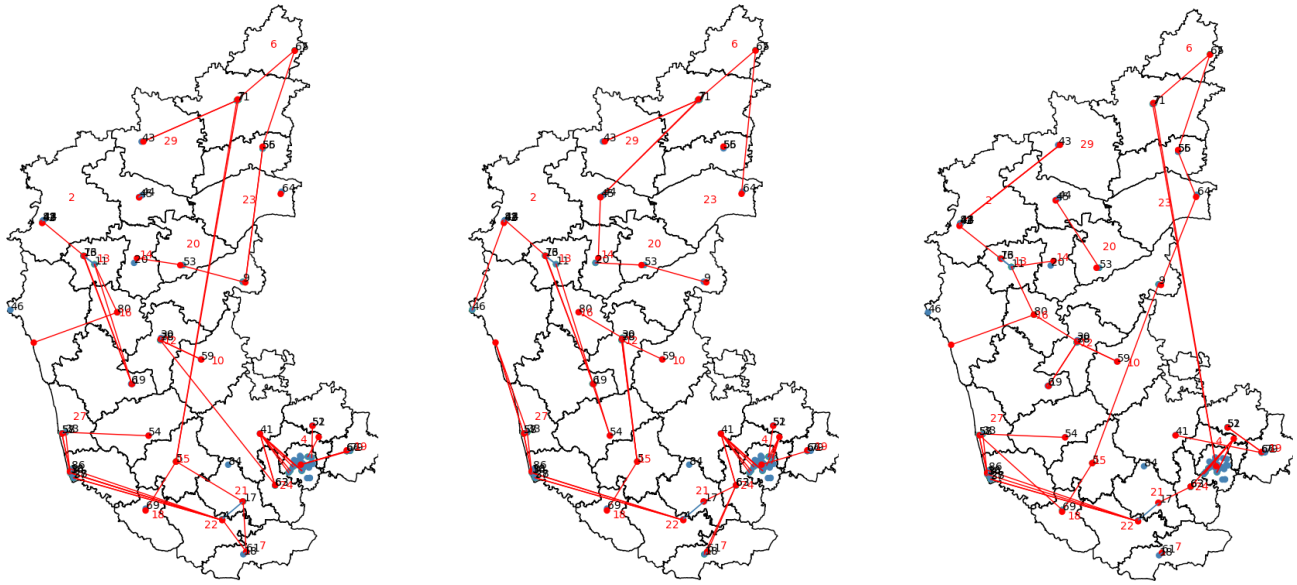
4.2 Benchmarking on Test Data

To solve our MIP formulation, we employ the `PythonMIP` library. We run the solver for 10 minutes with multi-threading (upto 4 threads) on each test file, on a machine having Intel 8^{th} generation i7 octa-core processor and 16GB RAM. As the total cost is in Crores (INR), even 0.1% improvement will be in the order Lakhs of Rupees. Hence, we run the solver for hours to get as better a solution as possible. The best score column is obtained by running the solver for 6 hours on each test file, upon which we calculate how close we have come to the to the **Tight Bound** shown in the **Optimality** column.

Note: We have used the `geodesic` function from `geopy.distance` library to calculate the distance between labs and centroids. A different distance function may yield different scores.

Table 1: Cost comparison of different methods on test data

Test	**Loose Bound**	**Tight Bound**	**Greedy**	**MIP (10 mins)**	**Best Score**	**Optimality**
001	119,005,600	121,169,442	138,782,153	121,942,529	121,403,144	0.193 %
002	139,845,200	142,147,481	162,357,199	142,503,116	142,407,282	0.183 %
003	128,474,800	130,622,301	151,810,345	131,316,815	130,814,729	0.147 %
004	108,314,400	110,041,158	124,178,874	110,211,626	110,180,913	0.127 %
005	135,850,600	138,438,815	157,921,190	139,749,409	138,607,286	0.122 %



(a) Transfers on 002 (b) Transfers on 003 (c) Transfers on 005

Figure 2: Comparison of external allocation in different test files

5 Robustness

5.1 Congestion - Much more samples on a day than capacity

When the samples of a district are within the total capacity of its labs, transfer to outside labs are not required keeping the allocations straightforward. But when the samples are much more than the labs' capacity, not only overload but backlogging happens making external allocation complex. However, our model solution can very well adapt to such scenarios which we verify below.

In our code `MIPsolver.py`, we have a method to scale (increase or decrease) the number of samples, by changing the value of `FACTOR` variable in this function definition – `def loadInput(FACTOR = 1)` which changes the number of samples for each district to `district['samples'] * FACTOR`. We experimented our solution on different values of `FACTOR` $\in (0, 2)$ and in almost all cases, the solver reached within 5% of the loose lower bound in roughly 15 minutes.

5.2 Empty Hubs - Highest capacity area having vacant labs

An unlikely worst-case scenario is Bangalore-Urban having 0 samples while having the largest capacity. If there is congestion in other districts, many will have to send their samples here. Considering this is also the location with a cluster of 34 labs within 40km, the scope for improvement in centroid-selection becomes further pronounced.

We simulated this by taking the first sample file provided, setting samples in Bangalore Urban to 0 and multiplying every other district's sample load by 1.5 (keeping capacities same). The MIP model's output was within 15% of the loose-lower bound for this dataset within 15 minutes. The loose lower bound doesn't include the cost of inter-district transport so it is much lower than the optimal here, which makes 15% quite a decent margin.

5.3 Scaling

Our MIP approach is general and scalable to extreme scenarios as well. While the current number of labs is 86, our formulation can handle upto 500 labs and huge number of samples (roughly 10^7 from each district) by using techniques similar to the ones developed above for approaching the original problem. As the model size shall become humongous in such extreme cases, it is highly recommend to deploy commercial solvers like *Gurobi* to obtain high quality solutions efficiently since they are faster than most other solvers by orders of magnitude.

Appendix - Rationale for tight lower bound

We provide a proof outline for the tight lower bound mentioned in Section 4.2

Observation 1

We compute M = all maximal cliques in the clusterability-graph G_c described above, which is computationally inexpensive for the given test sets using the optimization mentioned in section 2.4.3. Let us remove the constraint ((8) in final MIP model) of sending ≥ 1 sample to each lab in a cluster and input all maximal cliques (clusters) to this modified MIP model. Now all feasible combinations of labs can be selected by an arbitrary district as each combination is a subset of atleast one cluster in M.

Observation 2

Let the cost of the solution so obtained be C_{mip}. It can only be improved by:

1. Picking more precise clusters for the chosen labs. This is because maximal cliques can have redundant labs which don't get any samples and hence shouldn't be considered when computing the centroid.

2. Perhaps pulling the centroids closer to the district(s) sending the samples by sending 1 sample (or a small quantity) to some more labs currently in the cluster. This is an artefact of removing constraint ((8).

Observation 3

Let the diameter $D[i]$ of cluster i be defined as the maximum distance between any 2 labs in the cluster. Let $C[j]$ be the cluster district j sends it's external samples to, and S_j be the set of labs within $C[j]$ that actually get non-zero samples ($S_j \subseteq C[j]$).
Notice that among all clusters containing S_j, $C[j]$ would be the one with a centroid with least distance to district j. This is because the MIP model can otherwise send samples from district j to labs S_j though another cluster that has a closer centroid.

Observation 4

As mentioned in Observation 2 point 2, one way to minimize the distance to the centroid for a given district is by sending a small quantity of samples to closer labs in the cluster. For this a subset of C_j, say T_j would be added to S_j. This is because C_j is maximal and if lab $t \in T_j$ $s \in S_j$ are to be in a valid cluster, $dist(t, s) \leq 40km$ and hence $(t, s) \in E \ \forall \ t \in T_j$ and $s \in S_j$.

Observation 5

Let d be the distance between the new centroid so obtained and the centroid of cluster $C[j]$.
Notice $d \leq D[C[j]]$ as for any 2 subsets in a cluster, the distance between the centroids of the cluster is $\leq$ diameter.

The above statement can be formalized as – *Given a set of points S in euclidean space. For any 2 subsets S1 (centroid C_1), S2 (centroid C_2) of S, prove that dist(C_1, C_2) <= Diameter(S)*
Let us define Q as the set of points belonging to the area bounded by the convex hull of a cluster C (This set X is a convex set).

With this formulation, we show the following *lemmas*:

1. The centroid of any subset in the cluster C will lie in the set Q. This can be proven directly from the definition of the convex set X.

2. The maximum possible distance between any two points in this set X will be when both the points belong to the original set of points in the cluster C. This is true because if the two said points were points on the boundary lines of the convex hull, we could shift this point to either side till we hit a vertex to get a larger distance. Hence, the two points are guaranteed to be corners in X, which belong to C.

As centroid C_1 and centroid C_2 belong to X, their distance cannot be greater than maximal distance between two points in C, which is the diameter. Hence, dist(C_1, C_2) <= Diameter(S)

Hence, for each district, we can only optimize the position of it's centroid by atmost $D[C[j]]$.

Observation 6

Therefore, C_{mip} can only be improved by atmost $1000 * \sum R[C[j]]$ over all districts. Thus $C_{mip} - 1000 * \sum D[C[j]]$ is a lower-bound on the optimum cost.

Dataset based Observation

In the given dataset, we can use the DFS based cluster generation technique described in section 2.4.3 to get all maximal cliques for the given lab distribution. $1000 * \sum D[C[j]]$ comes out to be just 154,631 for the test datasets provided.

Extension to other datasets

For larger but similarly sparse datasets, it might not be tractable to enumerate all maximal cliques. Notice that for observation 1 to hold it is sufficient if the computed cliques include all maximal cliques as a subset. How can that happen? By relaxing the $D[i] \leq 40km$ constraint and adding more edges on top of G_c. Denote such a set of computed cliques as a "Superseder of M". Now, we can get a valid Superseder, by taking each connected component in G_c and assuming edges between every pair of vertices (thus making it a clique). This violates the $D[i] \leq 40km$ constraint, but that doesn't matter for any of our above observations. Now we can similarly computed $1000 * \sum D[C[j]]$ here, and while $D[C[j]] \leq 40km$ not holding makes the bound looser, it's still a fairly tight-bound that incorporates inter-district transfers.