These approaches demonstrate how our ideas evolved throughout the two weeks. Each is better the previous, with improved performance and more robust edge case handling ability.

First approach:
Idea:
Run shortest path algorithm from Soda Hall to every TA's home. Meanwhile record how many times a vertex is included on a valid a shortest path, call this h. Add vertices with the higher h, until all homes are connected. Run MST on these vertices and find out the drop-off locations and the final path.

Reasoning:
"h" indicates how "valuable" a vertex is in some sense. For vertices with high h values, since they are on the shortest path to more homes, they've more possibilities to be on the optimal solution.

Second approach:
Idea:
The most natural approach is to use the greedy algorithm by every time to drive to the closest home, mark it, and then repeat the process until all homes are marked. But this approach might be compromised by some particular inputs which exploit the "greedy" nature of this algorithm and yield bad result.
One improvement to this is to find the closest clusters instead of a single home. This way it will reduce the probability of driving a short-sighted greedy path and consider more of the bigger picture. Within each cluster, use MST to figure out the exact path. And finally connect all clusters using cluster shortest path which is the shortest path between two vertices each from a different cluster.

Third approach:
Idea:
This problem is similar to the Traveling Salesman Problem which can be approximated by an MST algorithm. However, this problem differs in that it does not need to go to every vertex in the graph, only homes. We design our approximation algorithm beginning by modifying the Prims MST algorithm. Instead of finding the edge to the closest vertex from the connected MST, it finds the shortest path to the closest home from the connected MST and add only the edges along the path. After all homes are added to the modified MST, we are done for finding all the relevant edges. This approach will still return a tree just like regular Prims Algorithm because it only adds paths from a tree. It also preserves the factor of 2 approximation for the same reason as the original TSP approximation method using MST.

Drop strategies:
Above algorithms all assume driving TAs to their home without letting them walk. Now we add the drop-off strategy to optimize the result. This drop-off strategy works for all three algorithms.

1. Leaf drop-off: If there is only one home at the leaf of a branch, recursively remove the leaf vertex until it's no longer a leaf. That branch vertex is the drop-off location.
2. Long leaf drop-off: If there are multiple homes along a single branch, recursively drop the leaf vertex until it's another home. That home is the drop off location of TAs.
3. Branch drop-off: If the home is right on the branch, just drop-off there.