A clustering problem begins with N items of data.

It wants to organize the data into K clusters.

To do so, it needs to determine K means.

Then each item of data will automatically organize itself by joining the cluster with the closest mean (easy for the computer to handle).

Once this is done, the entire clustering can be described by the K mean values (compression of information).

Any new item can automatically be added to the clusters by finding the nearest mean (easy for the computer to update).

The K-means algorithm has some practical uses.

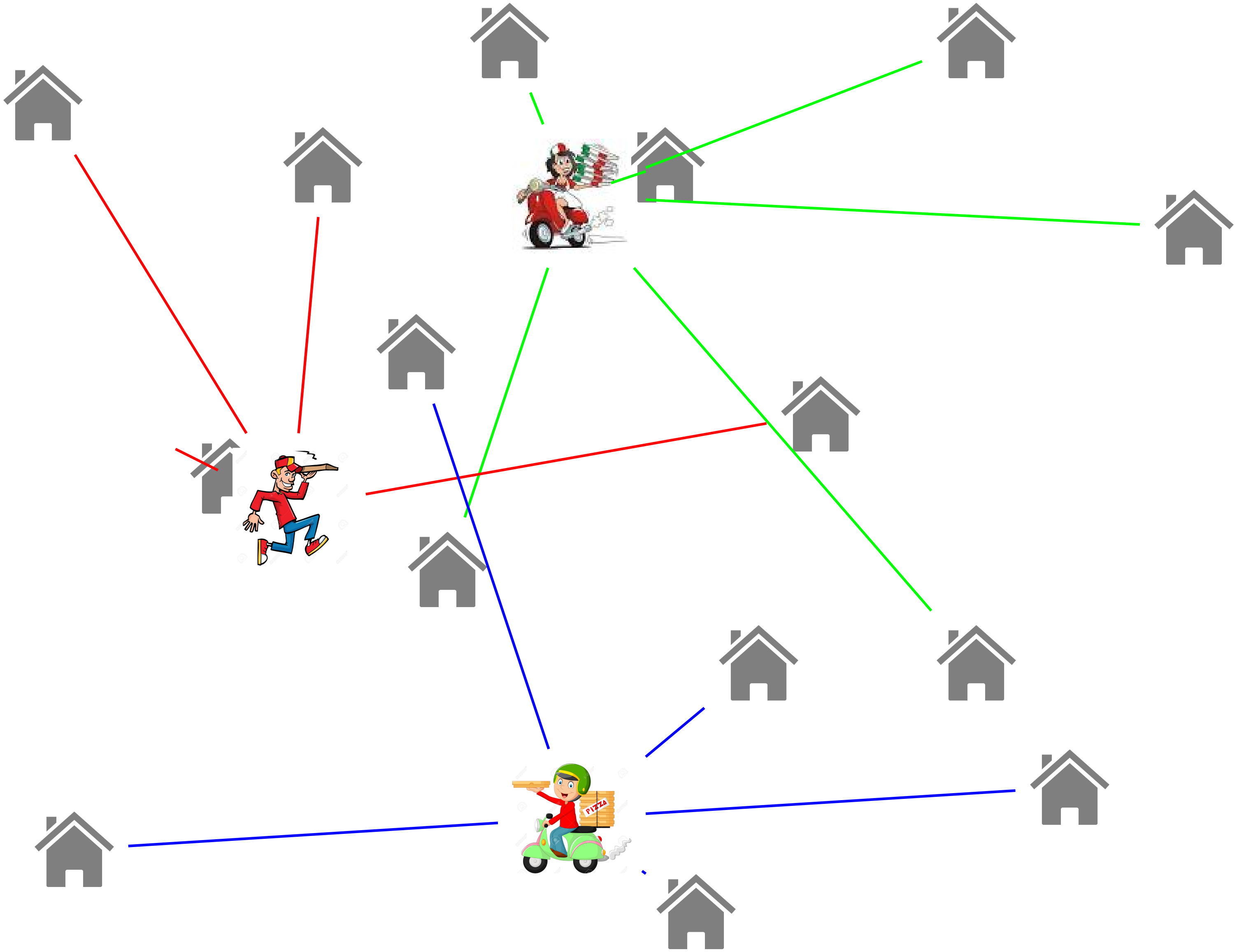
Suppose that the things we called data items were actually customers of some sort, and that each customer needed to have access to a service center.

We may already have $\mathbf{K}$ service centers built, in which case we could study whether customers go to the nearest center, and whether the centers are well placed.

Or, if we are planning on building $\mathbf{K}$ service centers, the K-means algorithm would suggest where they should go.

As an example of this application, let us consider a problem in which the houses of customers are scattered over a model town

PIZZA

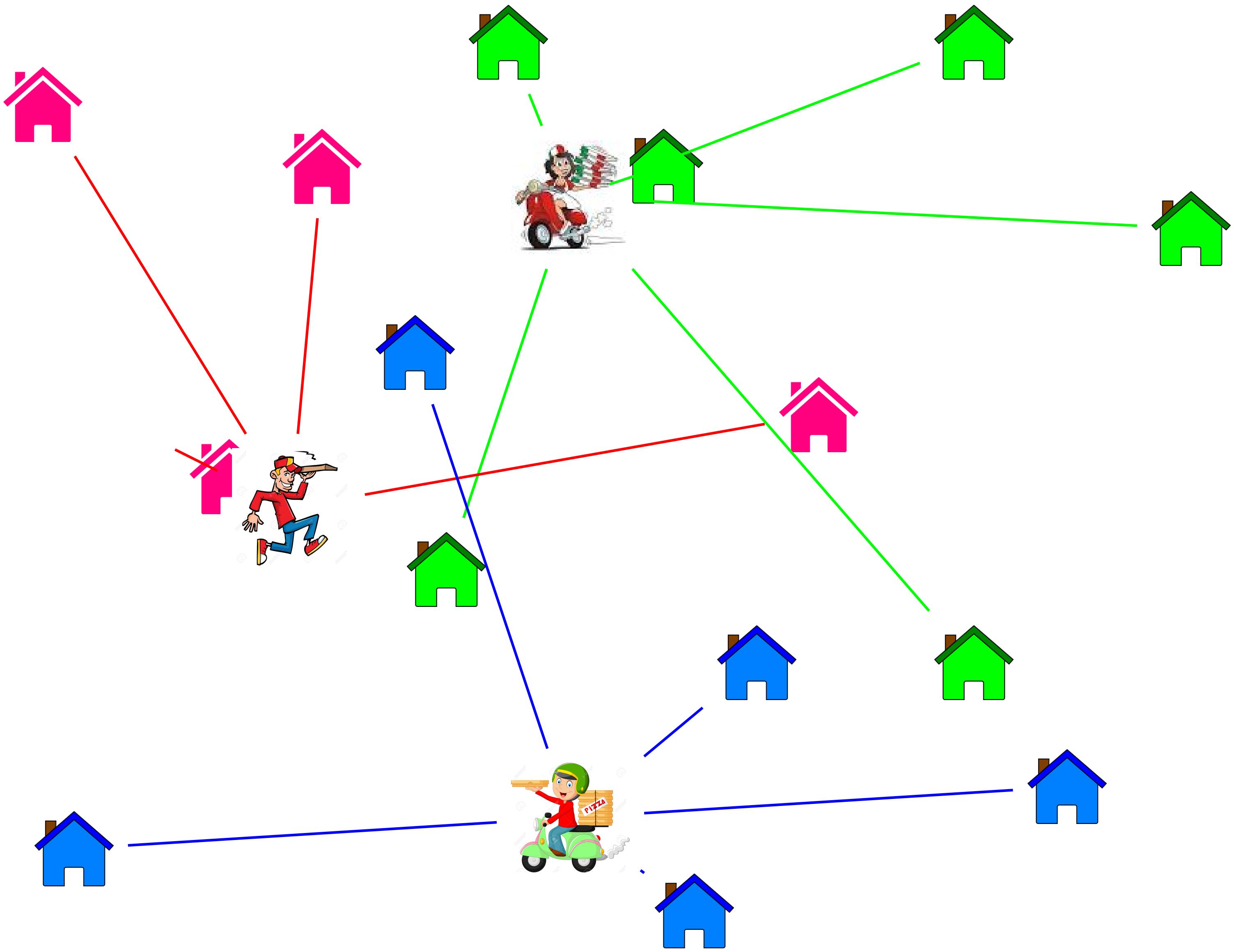


We want to estimate how efficient this delivery system is.

The K-Means algorithm measures the cost of a single delivery in terms of the square of the distance traveled.

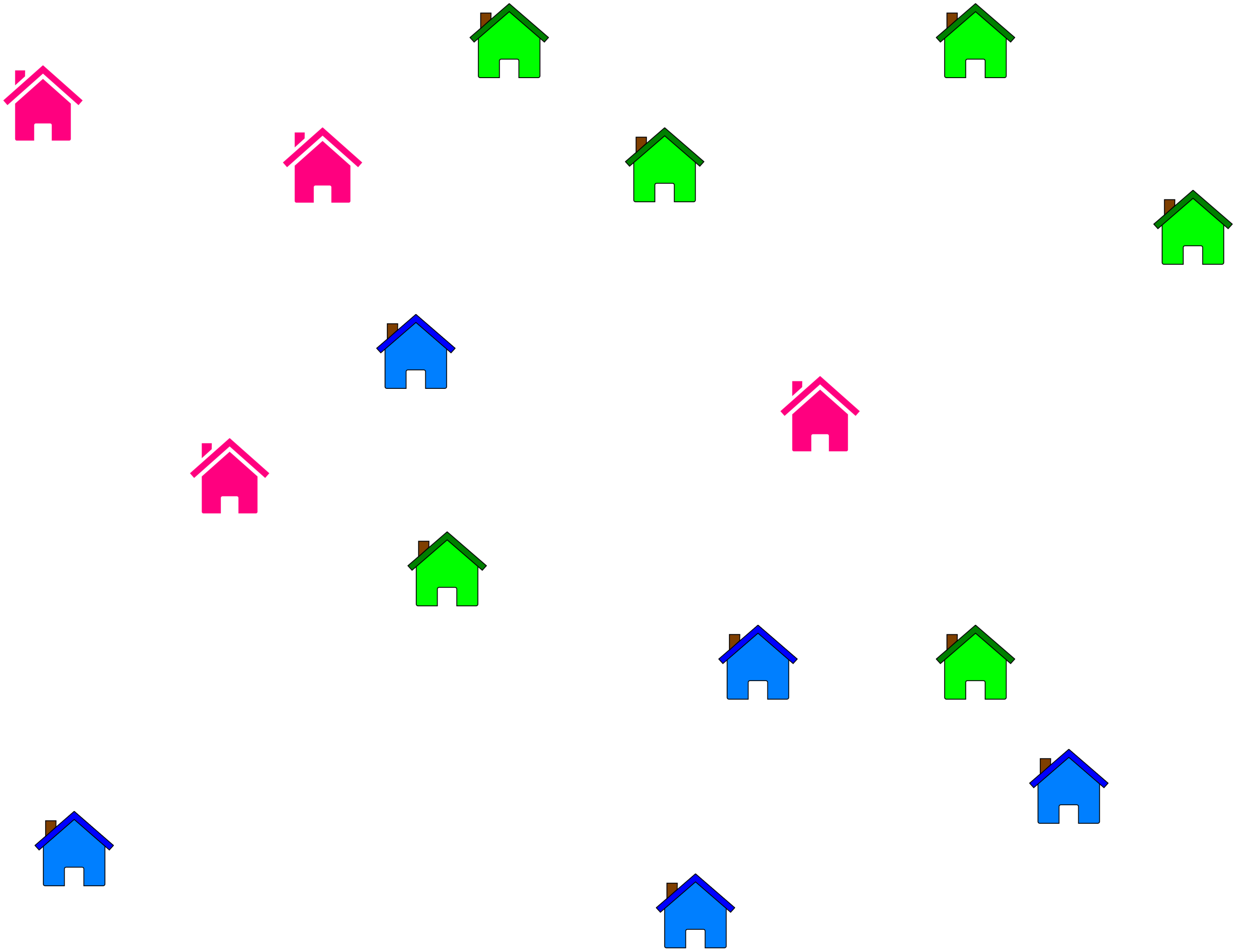
So pizza deliveries of 1 mile, 2 miles or 3 miles would have a K-Means cost of \$1, \$4 or \$9. This crazy increase in cost occurs because the K-Means algorithm is really trying to avoid long travel times.

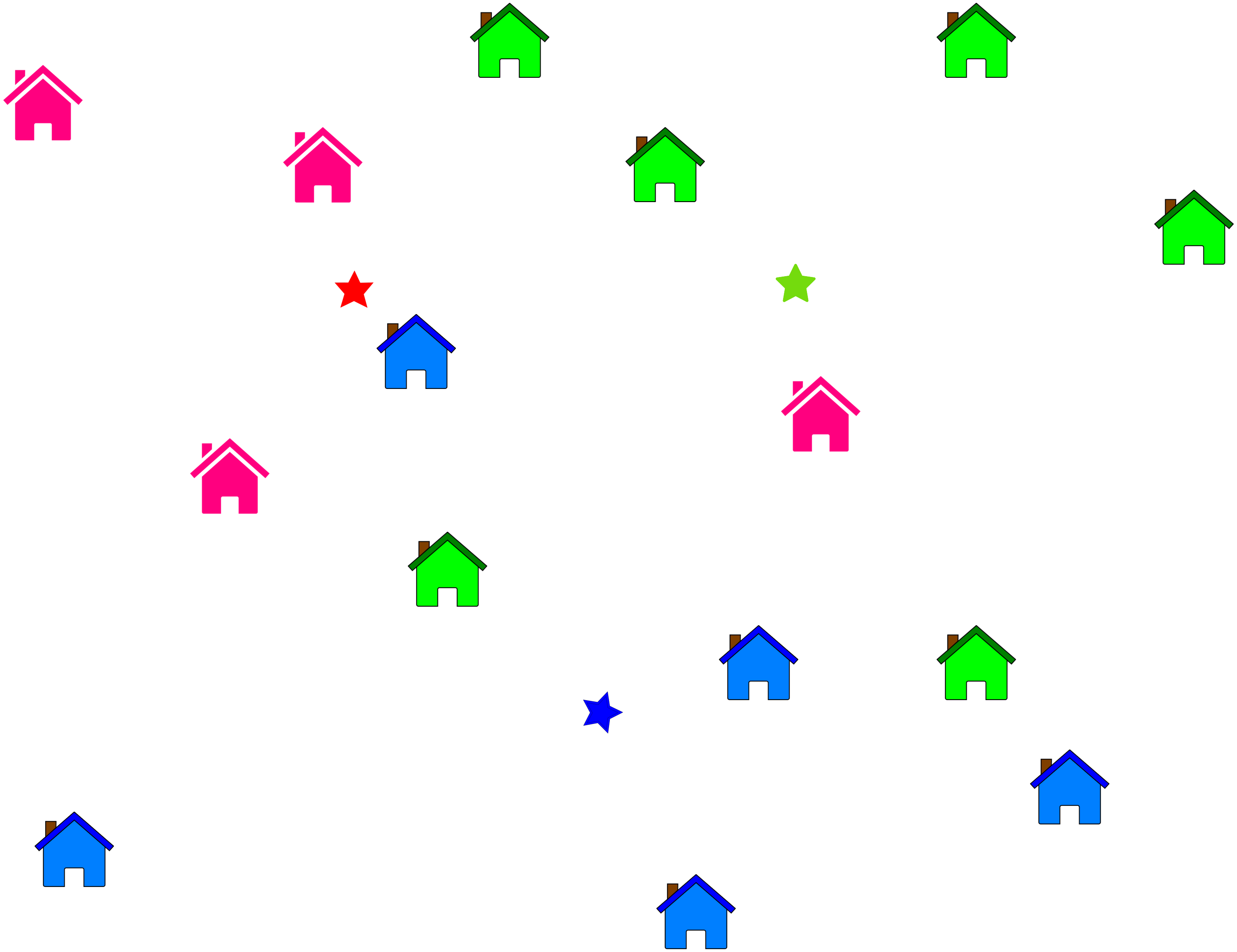
The cost to deliver one pizza to each house, with our initial somewhat random delivery pattern, turns out to be \$2,696.

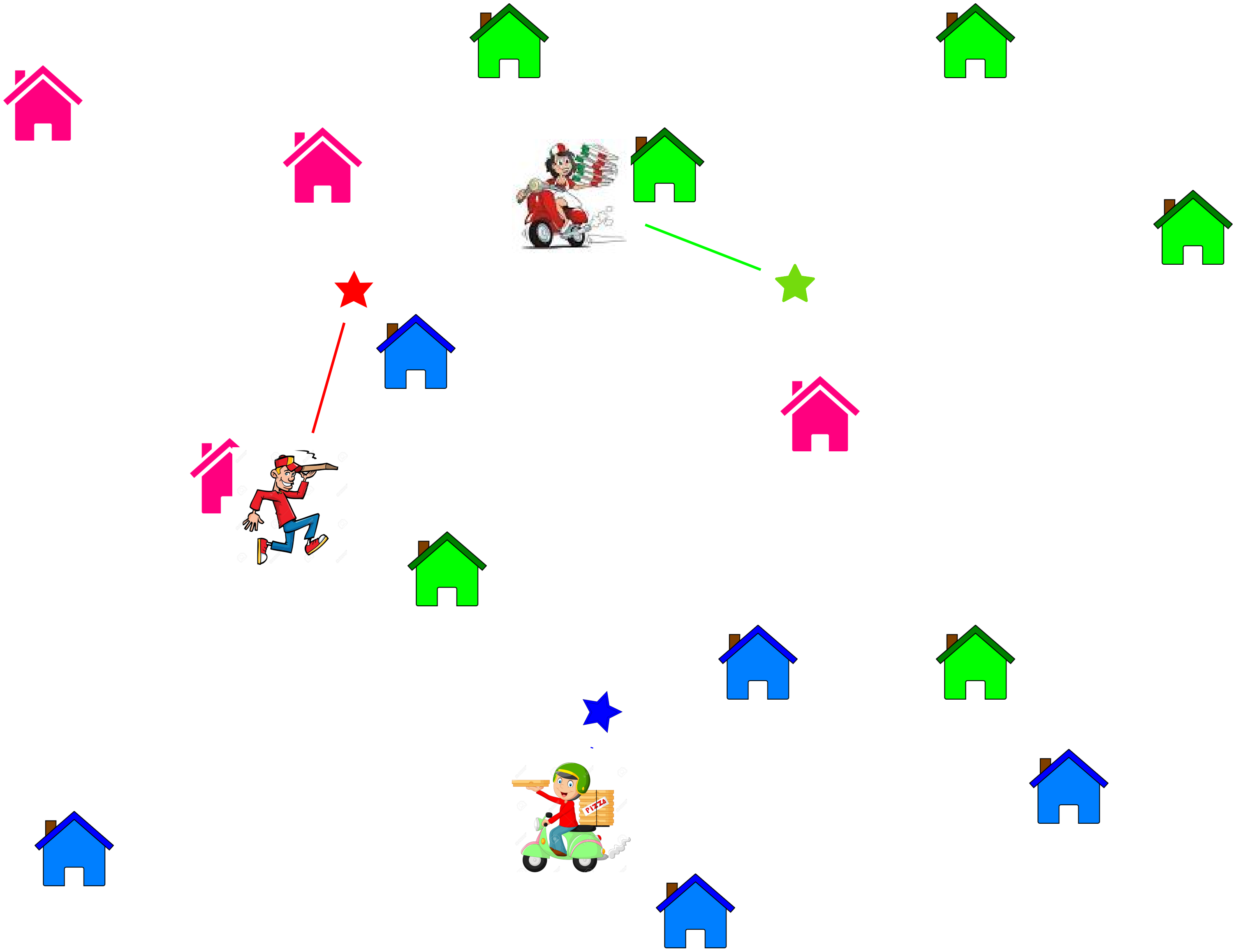

PIZZA

Houses served from the red pizza truck will tend to call the red pizza truck again. So those houses form the “red” cluster, and we also get blue and green clusters.

Naturally, these are not ideal arrangements. Some houses are going to get cold pizza, when they could have ordered from another, closer, pizza truck.



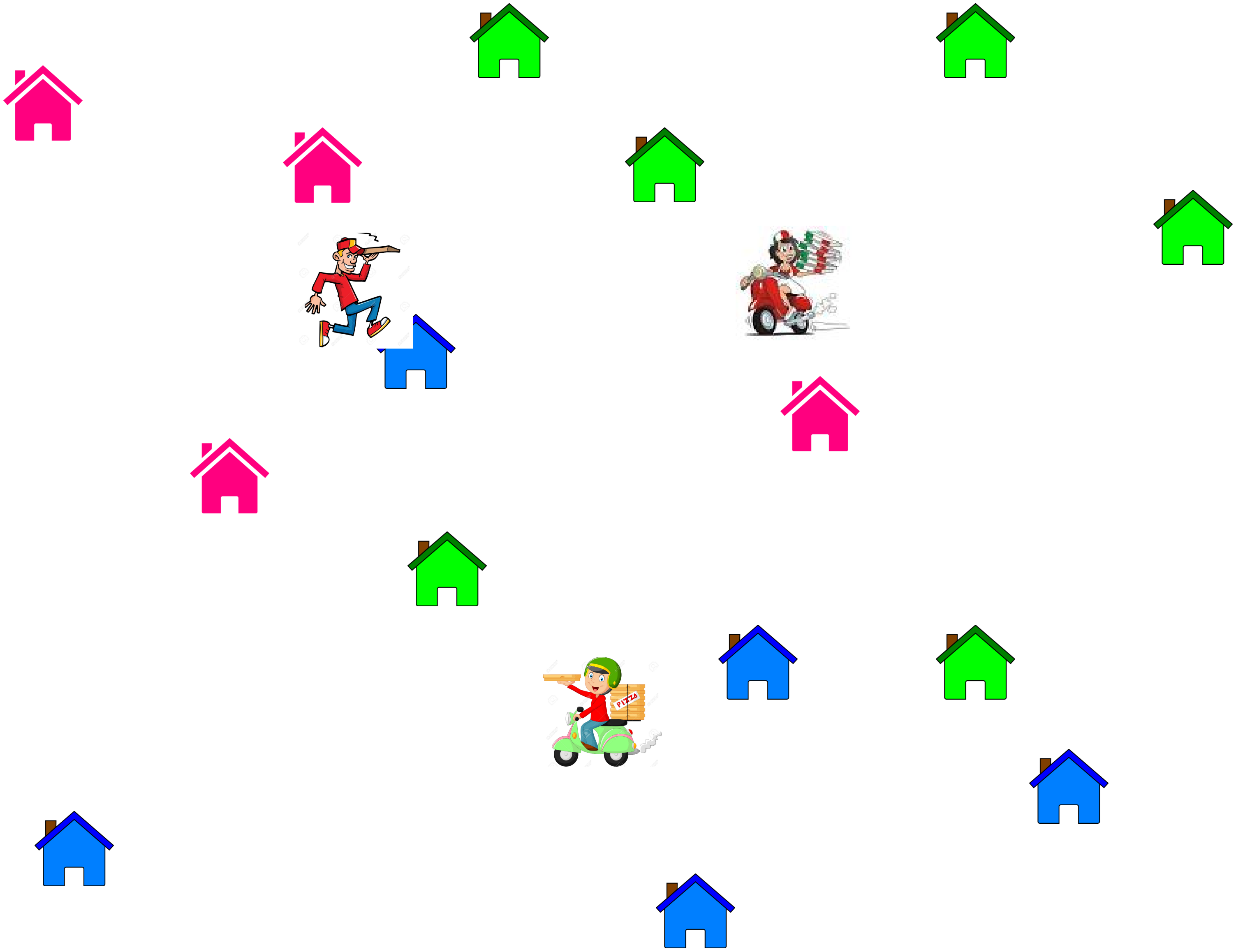



PIZZA

On Friday, the pizza trucks realize that the houses have been divided up into red, green and blue delivery zones.

On Saturday, each truck realizes that, if those are going to be their customers, they might as well move their truck to the center of their delivery zone, which will cut down delivery time.

On Sunday, the trucks move to their new locations.

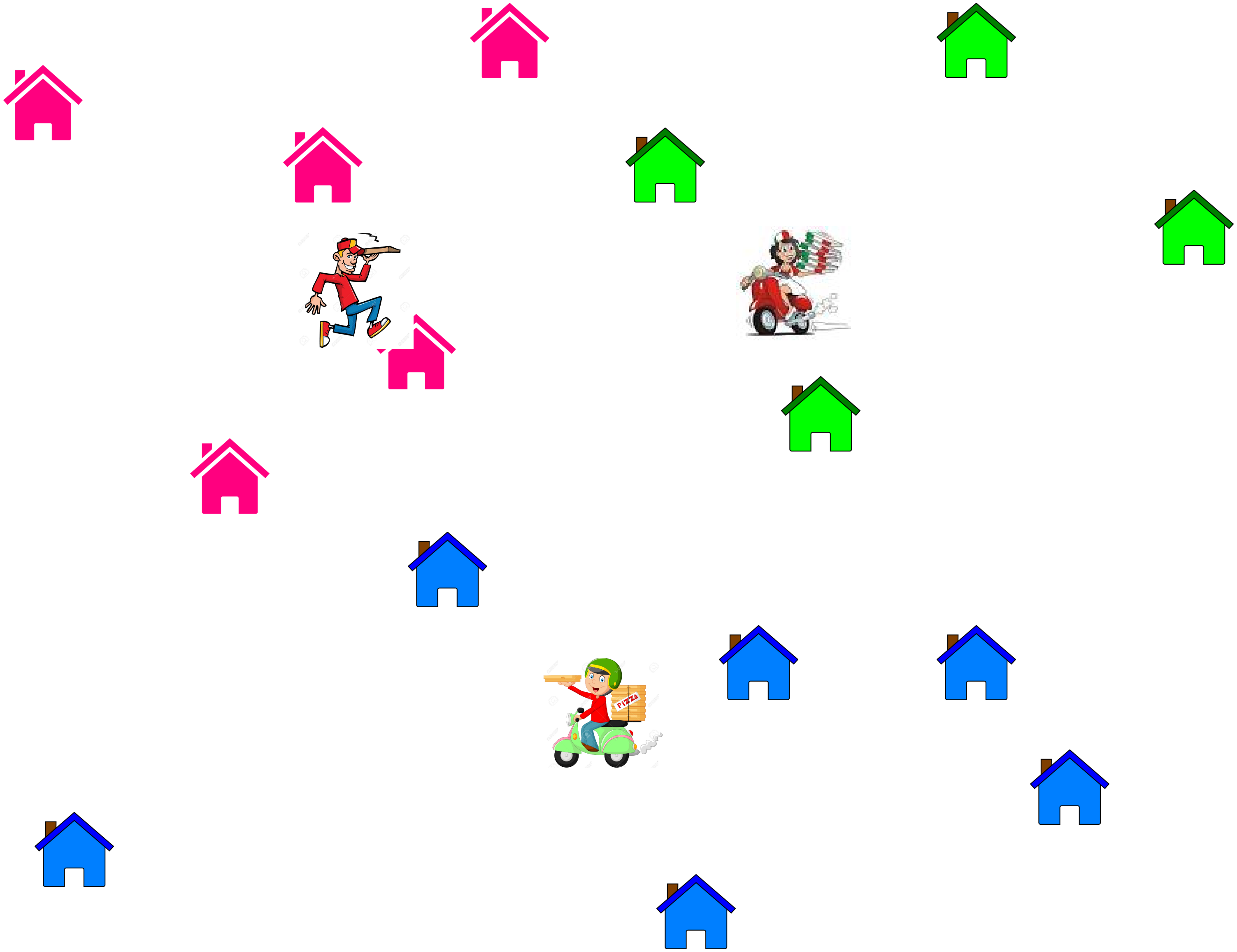

PIZZA

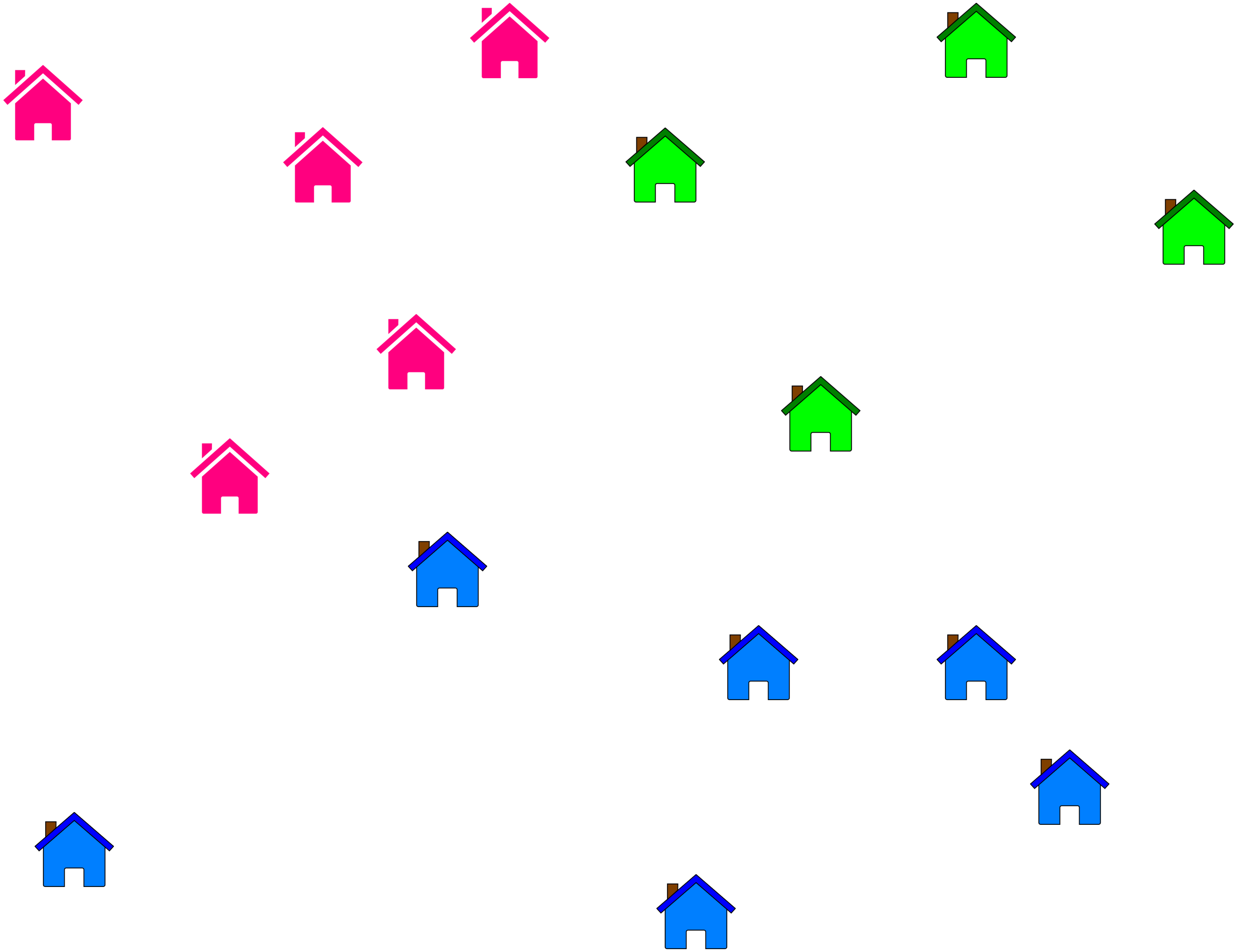
It makes sense to move each pizza truck to the center of the region it is serving.

In this example, the move reduces our K-Means cost to deliver one pizza to each house from $2,696 down to $2,064.

Of course, if we move the trucks, some other things change, which we should pay attention to next!

On Monday, the pizza trucks look at the map, and realize that they should agree that every house should be served by the nearest truck.

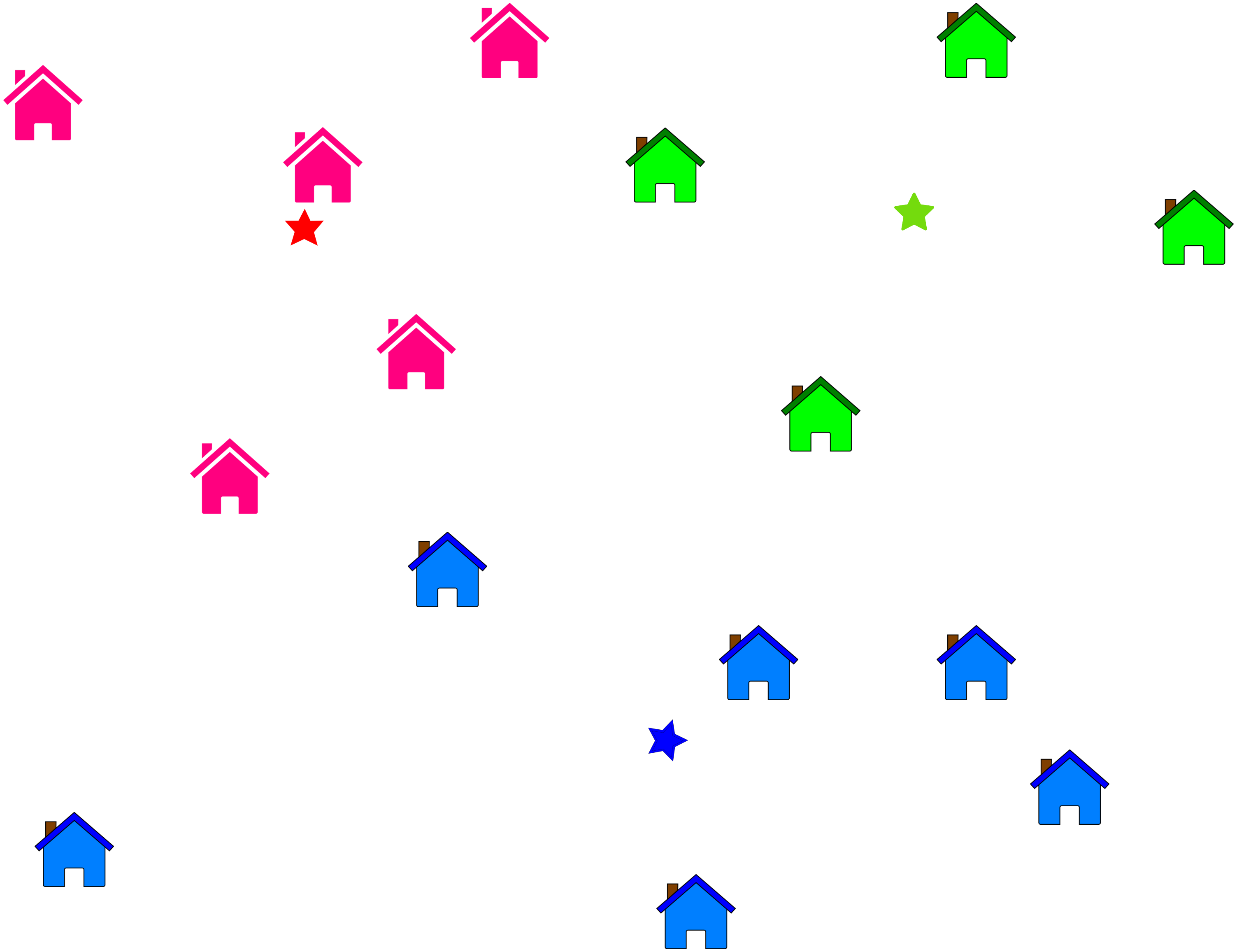

PIZZA

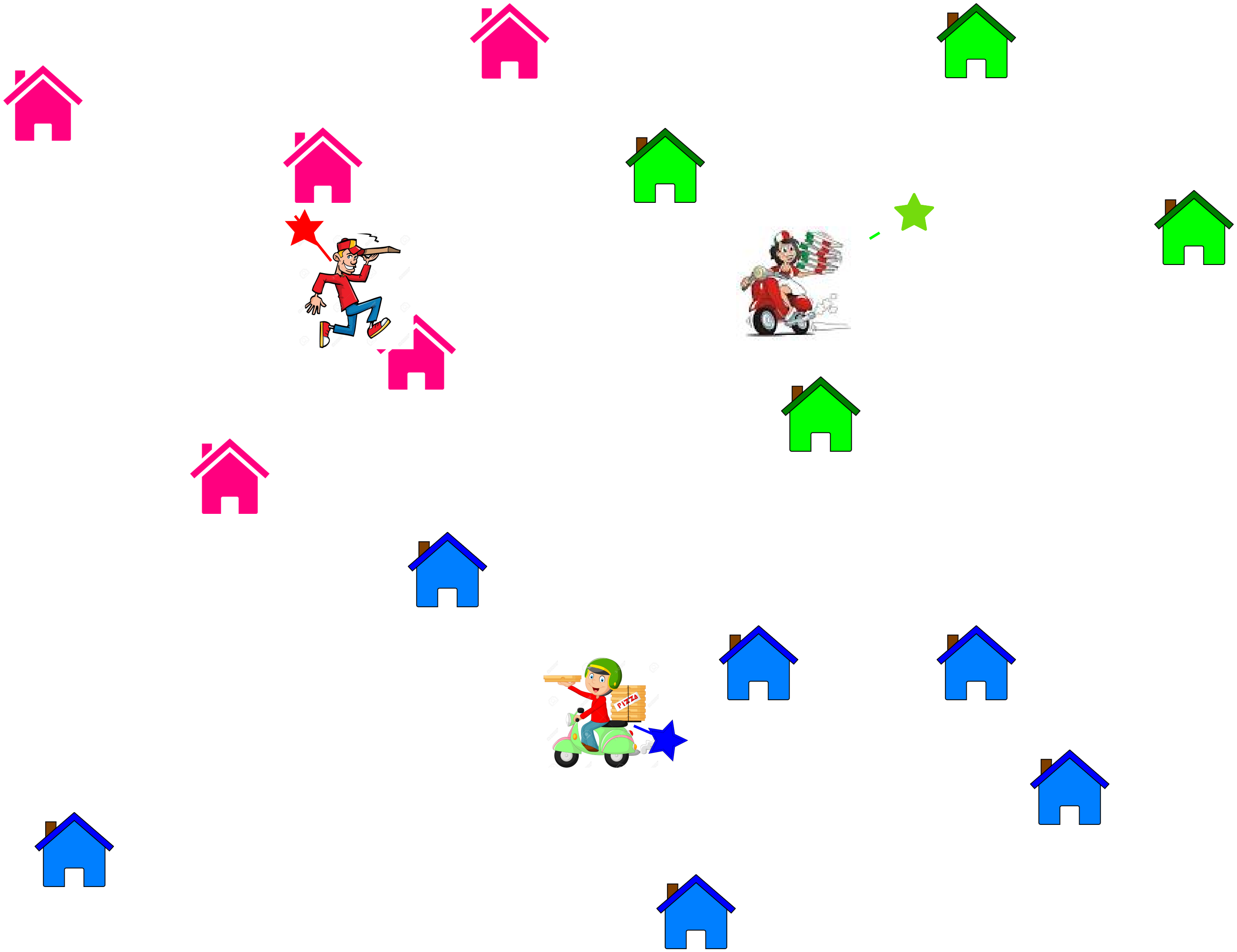


We moved all the trucks to the centers of their regions, and that made sense.

Now we notice that sometimes a customer is not being served by the closest truck. That doesn't make sense.

If we specify that a customer order should always be filled by the nearest truck, then we have changed our delivery system in a way that reduces our K-Means cost from $2,064 to $1,424.

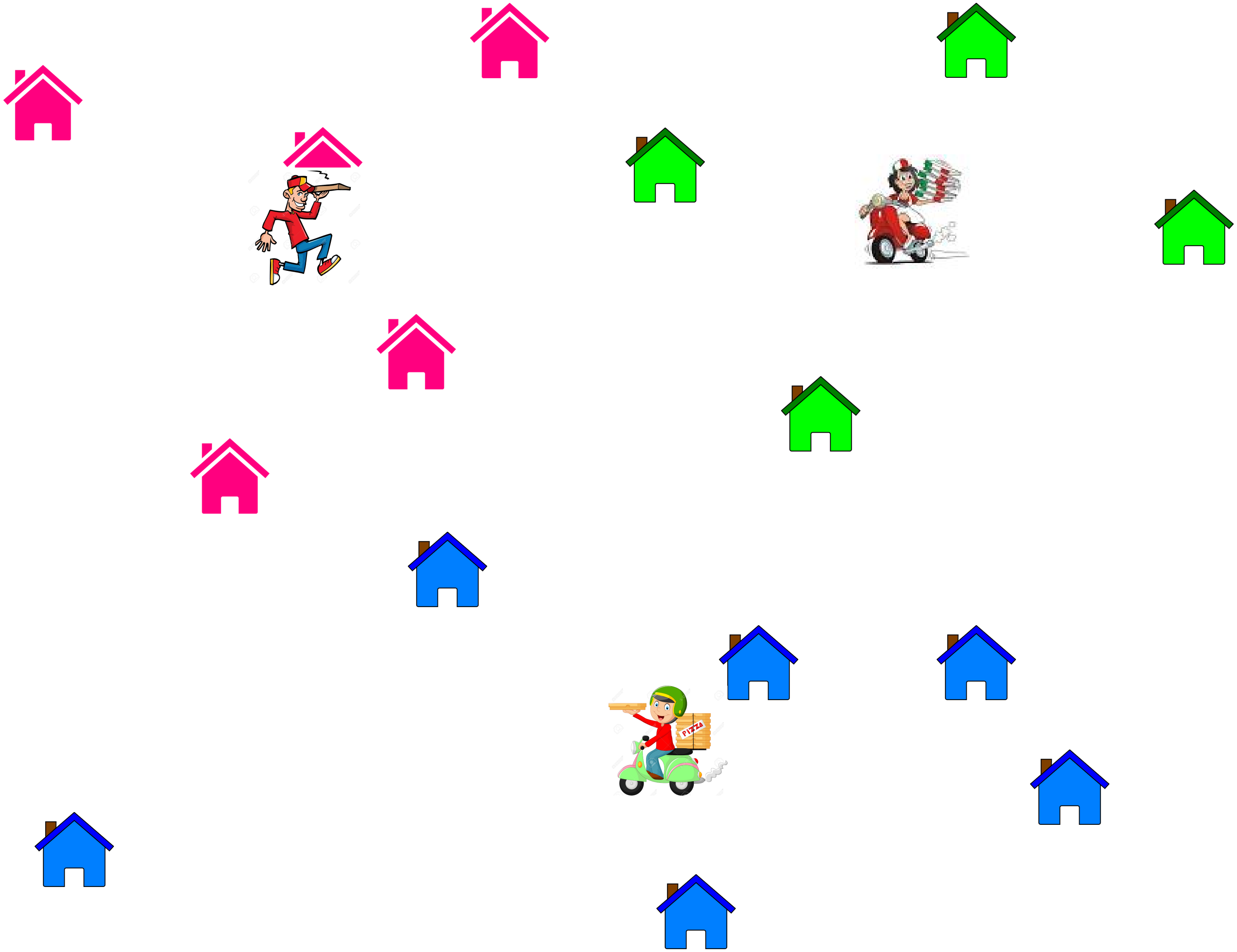


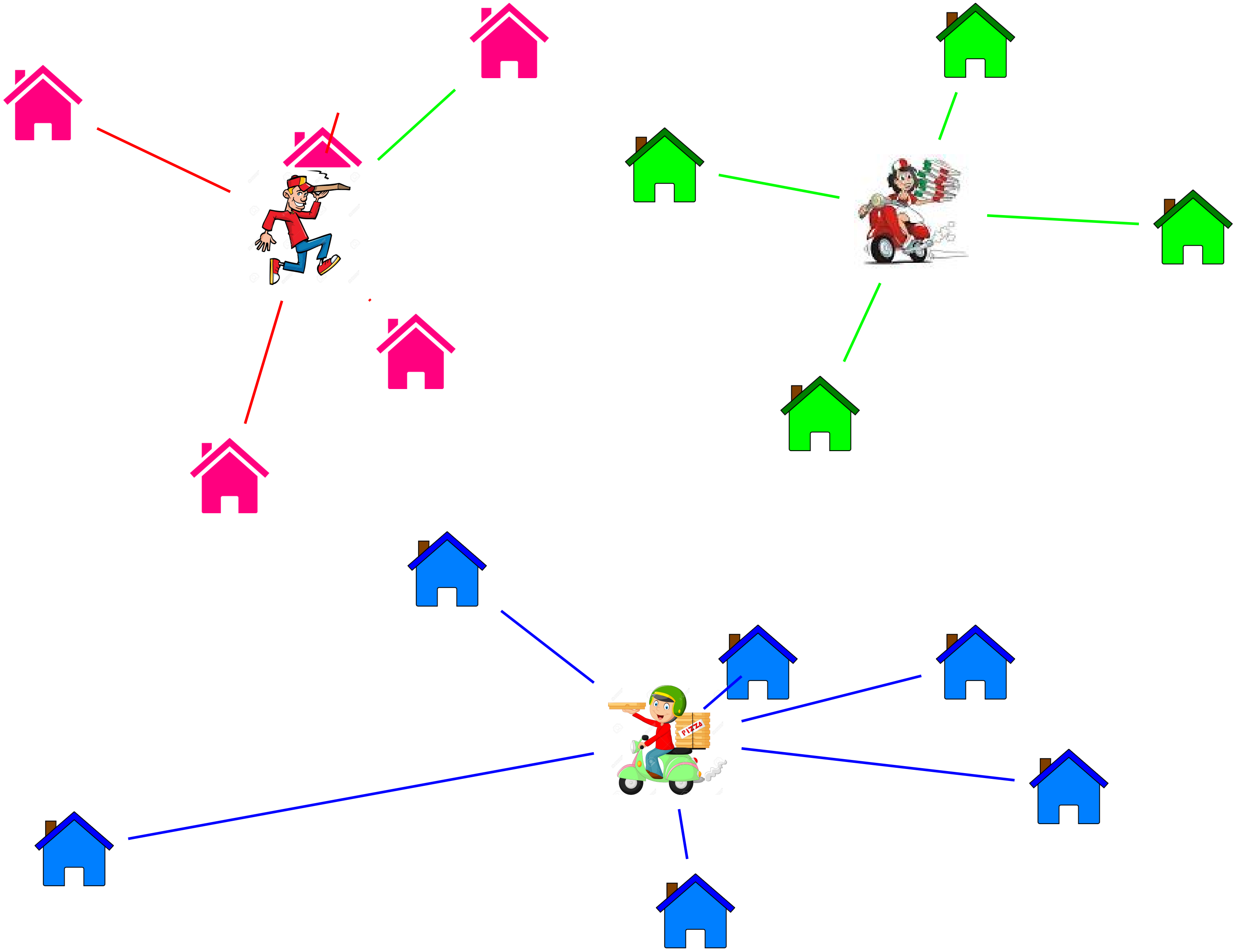

PIZZA

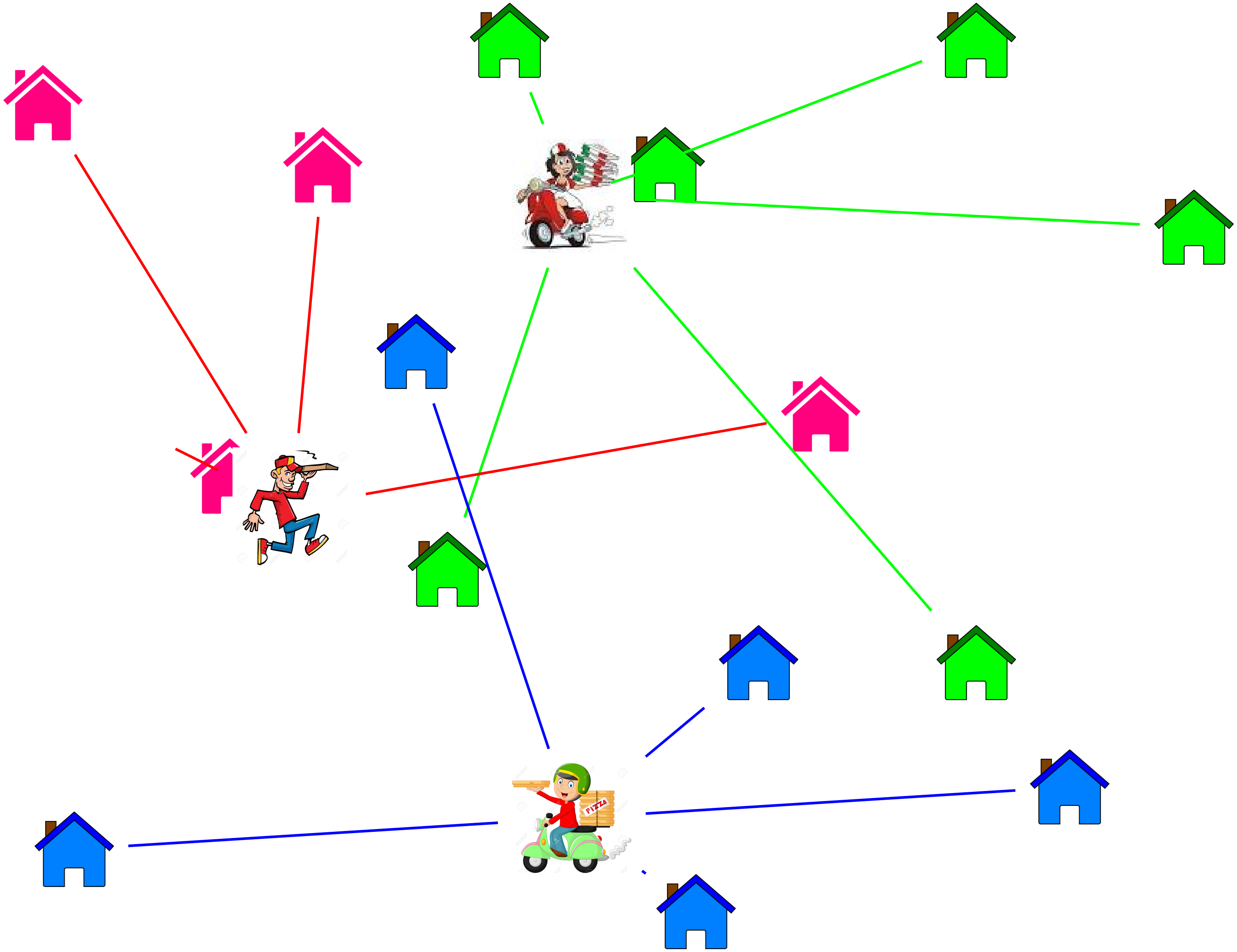
On Wednesday, the trucks realize that, if some customers have switched their delivery truck, then that means that the delivery regions have changed, and so the trucks are no longer in the centers of the new delivery regions.

On Thursday, the trucks have to move again. But this time not so far. That's because the K-means algorithm is getting closer to an ideal solution.

If we make this adjustment, we reduce our K-Means cost of delivering one pizza to each house from $1,424 to $1,278.


PIZZA


PIZZA


PIZZA

Starting from an initial random locations of trucks and delivery regions, with a K-Means cost of \$2,696, we were able to come up with new truck locations and delivery regions that reduced our cost to \$1,278.

We get more confidence in the result by just looking at the final arrangement. It's clear that the K-Means algorithm has done a good job of grouping the houses and relocating the trucks.

We did two steps of the K-Means process; often it's necessary to take many more steps, but for this small example we can see that the cost didn't go down so much, and that the trucks didn't move so much. So at this point we are probably close to having the best possible arrangement of delivery regions and trucks.

We were able to reach this using an automatic procedure that will work whenever we know the location of the customers, the number of trucks (or servers), and can measure distances.