Lab 3-1
The cluster-median problem

Tutorial for Optimization

DMKM Course

Xinyu WANG

wangxinyu.xenia@gmial.com

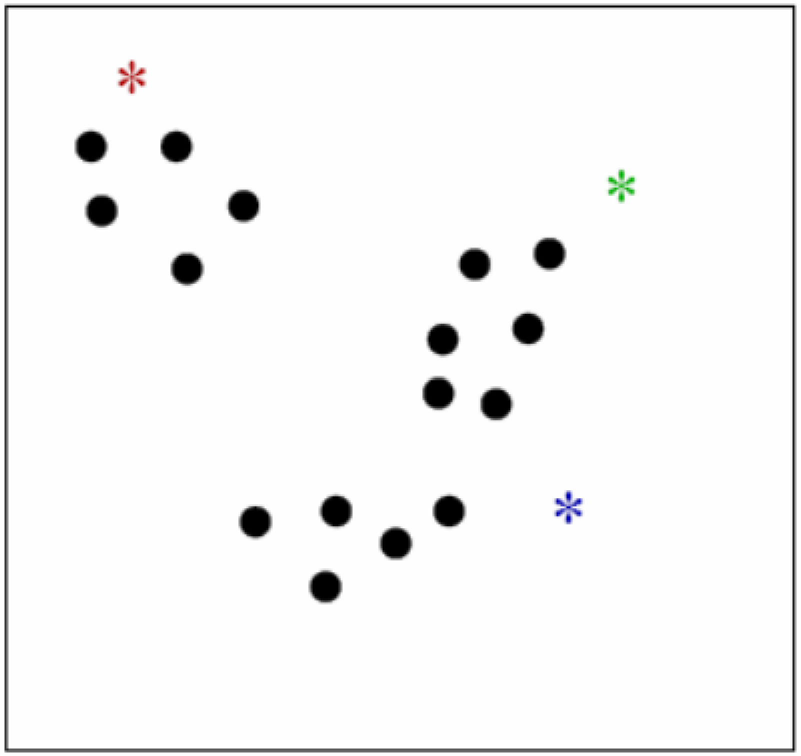
28-Nov-2013

Basic algorithms of Machine Learning

- Supervised learning: algorithms are trained on **labeled** examples, i.e., the desired outputs are known. Common traits: output set is know before training. You can always label a new input with one label in the set.
  - Neural networks.
  - Support vector machine
- Unsupervised learning: algorithms operate on **unlabeled** examples. Given a dataset, you only know how they are grouped, but you don't know the labels of the clusters.
  - Clustering
    - K-means
      - **K-median**

K-means

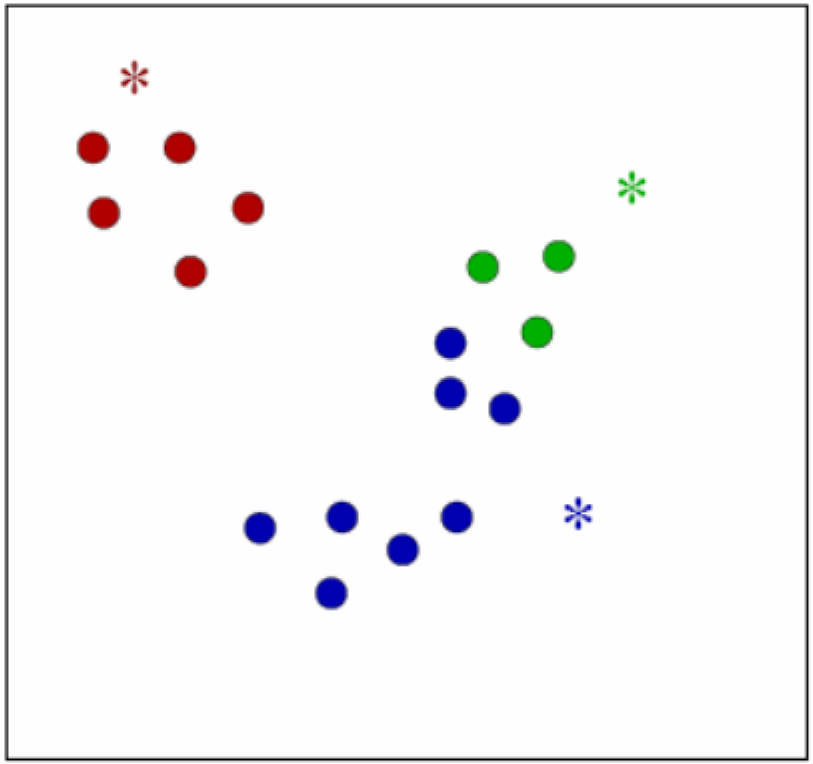
- K = # of clusters (given); one "mean" per cluster
- Initialize means (by picking k samples at random)
- Iterate:
  - (1) assign each point to nearest mean
  - (2) move "mean" to center of its cluster



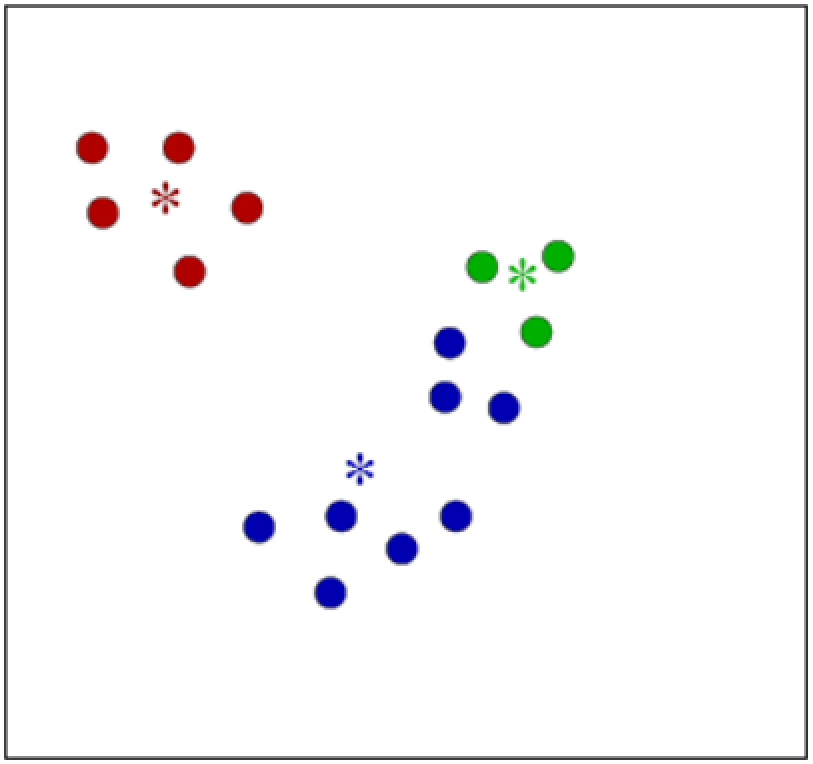
K = 3
three colored points are initialized means

*** Mean is not a real point, it is only a value.**

K-means



(1) assign each point to nearest mean. The points that are close to nearest mean are marked with the same color as the mean.



(2) move "mean" to center of its cluster. This is updating means.

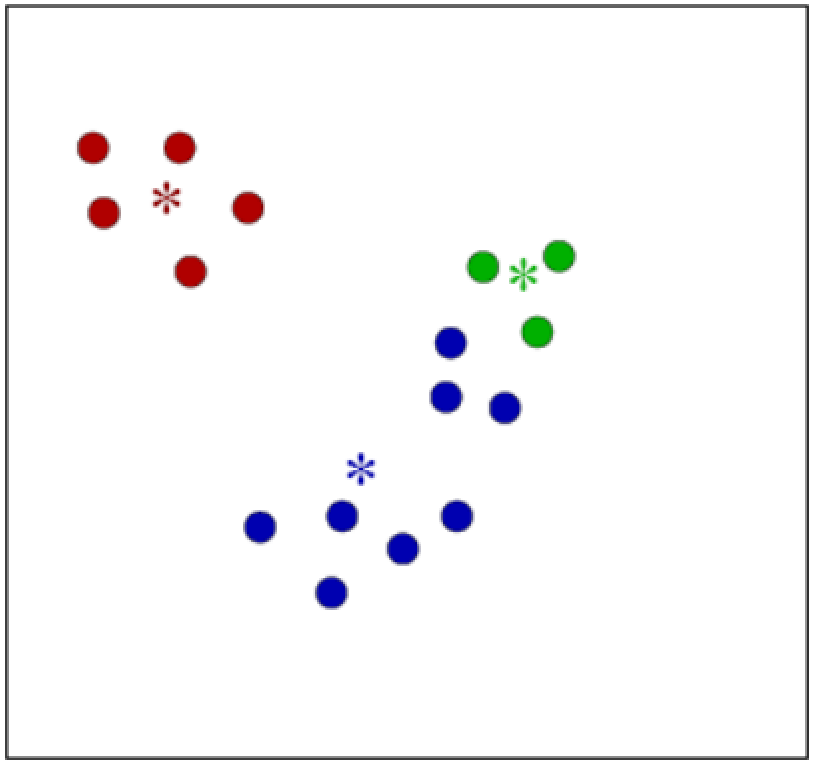
K-means

The objective function:

$$\min_{\{\boldsymbol{\mu}_1,\cdots,\boldsymbol{\mu}_k\}} \sum_{h=1} \sum_{\mathbf{x}\in\mathcal{X}_h} \|\mathbf{x}-\boldsymbol{\mu}_h\|^2$$

It is to minimize the Euclidean distance between the mean and all the other points in one cluster.

h , is the number of clusters, here $h = 3$
μ_h, is the mean value of cluster h
x, is one point in cluster h



After a number of iterations, convergence is reached where means keep stable.

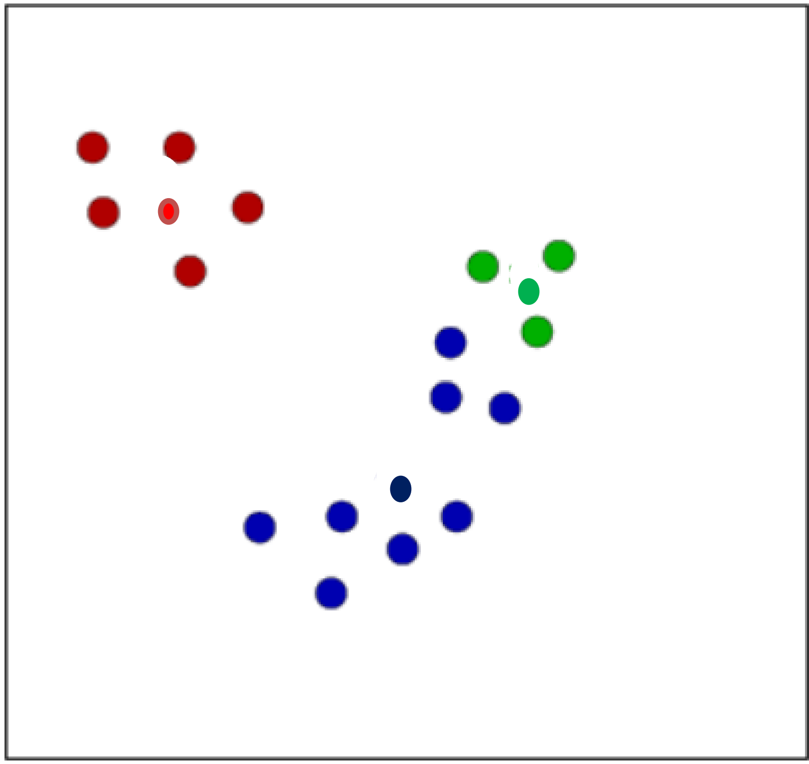
K-median

Almost the same as K-mean clustering,
just here median is used instead of mean value.

Thinking: how to get the median of the mean value
of a list of integers?

In K-median, median is an actual point.

First, the number of cluster is given,
then medians are randomly initialize for all the clusters.
By looking for the minimal distance between a median
and the other points in one cluster, it will finally reach
a convergence, where medians keep stable.



In your Lab description, r is the median, I represents a cluster, i, j are two points in one cluster, d_{ij} represents the Euclidean distance between point i and point j. d_{ir} is the Euclidean distance between point i and median r.
When convergence is reached, it will have:

$$\sum_{i \in \mathcal{I}} d_{ir} = \min_{j \in \mathcal{I}} \sum_{i \in \mathcal{I}} d_{ij}$$

An integer optimization problem

To formalize K-median as an integer optimization problem, we need to think it in another way:

let's suppose in our dataset, there are m examples, so m points if we can visualize them. Initially, we can simply think of these m points as m clusters, therefore there are m medians, the median actually is the point itself. Suppose there are only k clusters are needed, then $(m - k)$ clusters are empty.

Denote: $i, j = 1, \dots, m$; i, j represents two points
As there are m clusters at beginning. Let j be the median of its own cluster, we can have:

$$x_{ij} = \begin{cases} 1 & \text{if element } i \text{ belongs to cluster-} j, \\ 0 & \text{otherwise.} \end{cases}$$

As point j belong to cluster-j, so $x_{jj} = 1$ and $\sum_j^m x_{jj} = k$, as there should be k clusters.

An integer optimization problem

$$\begin{aligned} \min \quad & \sum_{i=1}^{m}\sum_{j=1}^{m} d_{ij}x_{ij} \\ \text{subject to} \quad & \sum_{j=1}^{m} x_{ij} = 1 \quad i = 1,\ldots,m \\ & \sum_{j=1}^{m} x_{jj} = k \\ & \boxed{x_{jj} \geq x_{ij} \quad i,j = 1,\ldots,m} \\ & x_{ij} \in \{0,1\} \end{aligned}$$

m points (i), m clusters (j), d_{ij} is the distance between i and j. Objective: to minimize the distance between median and the points in one cluster.

One point must belong to one cluster.

k clusters.

A point can only belong to a cluster that exist

You can verify that this expression actually implies m^2 constraints.
A better option is to use:

$$mx_{jj} \geq \sum_{i=1}^{m} x_{ij}, j = 1, \ldots, m$$

For it only has m constrains.

Solve this problem in AMPL

To solve this problem in AMPL:

1. In .mod file, you need to describe the problems in AMPL
2. In .dat file, you need to specify the number of examples, the number of clusters, and a distance matrix

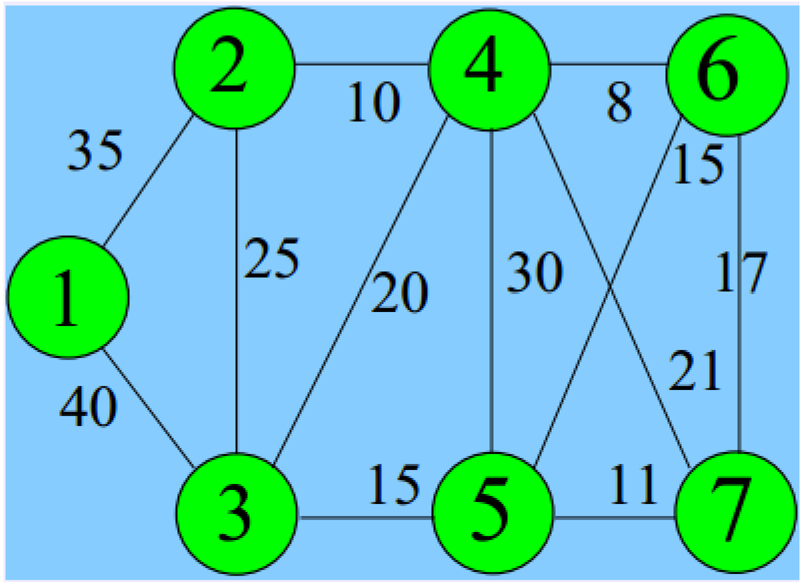
Unlike the previous assignments, there is no data-generating software for this. But you can build a simple dataset, compute the distance matrix D, which is the main thing you would need.

The number of your data examples couldn't be very large, otherwise the number of constraints would probably exceed the limit of student-version AMPL.

CPLEX would be a better option than MINOS to solve this problem, for CPLEX allows more variables and constraints than MINOS.

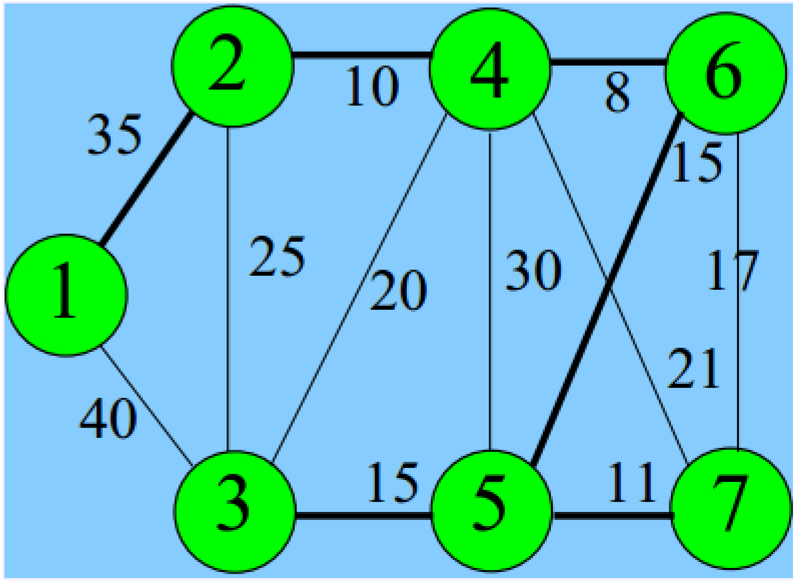
In the end, you would get the minimal total distance as your objective function value, and you would have the value of x_{ij} for $i, j = 1, \ldots, m$, which is expressed as a matrix, from this matrix, you can easily see k medians and k clusters.

Solve it as a minimum spanning tree problem



Undirected network G = (N; A) of n nodes and m arcs.
(i, j) is the same arc as (j, i).
We associate with each arc (i, j) ∈ A as cost c_{ij}.

A spanning tree T of G is a connected acyclic subgraph that spans all the nodes.
A connected graph with n nodes and n-1 arcs is a spanning tree.
The minimum cost spanning tree problem is to find a spanning tree of minimum cost, T^*.

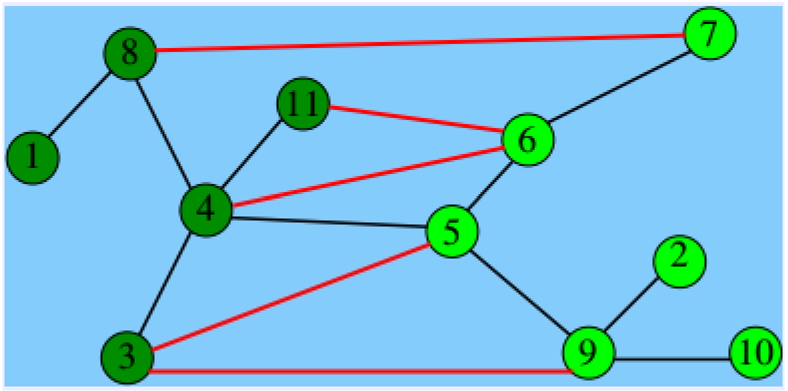


Two ways to find T^*

Cut Optimality Theorem

- Cut optimality conditions:

For every tree arc (i, j) ∈ T*, $c_{ij} \leq c_{kl}$ for every arc (k, l) contained in the cut formed by deleting arc (i, j) from T.

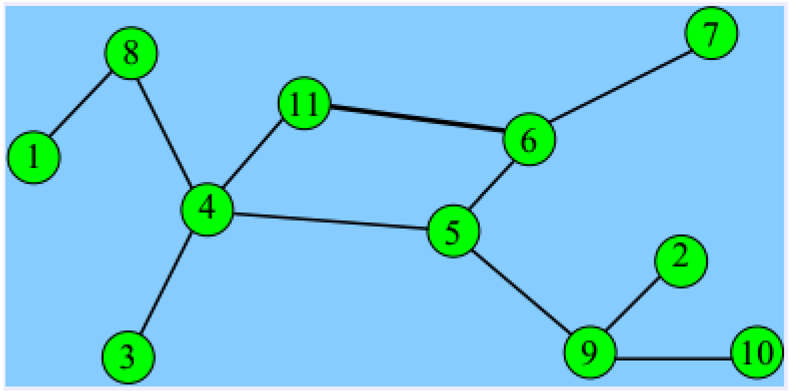


- Prim's Algorithm

Path Optimality Theorem

- Path optimality conditions:

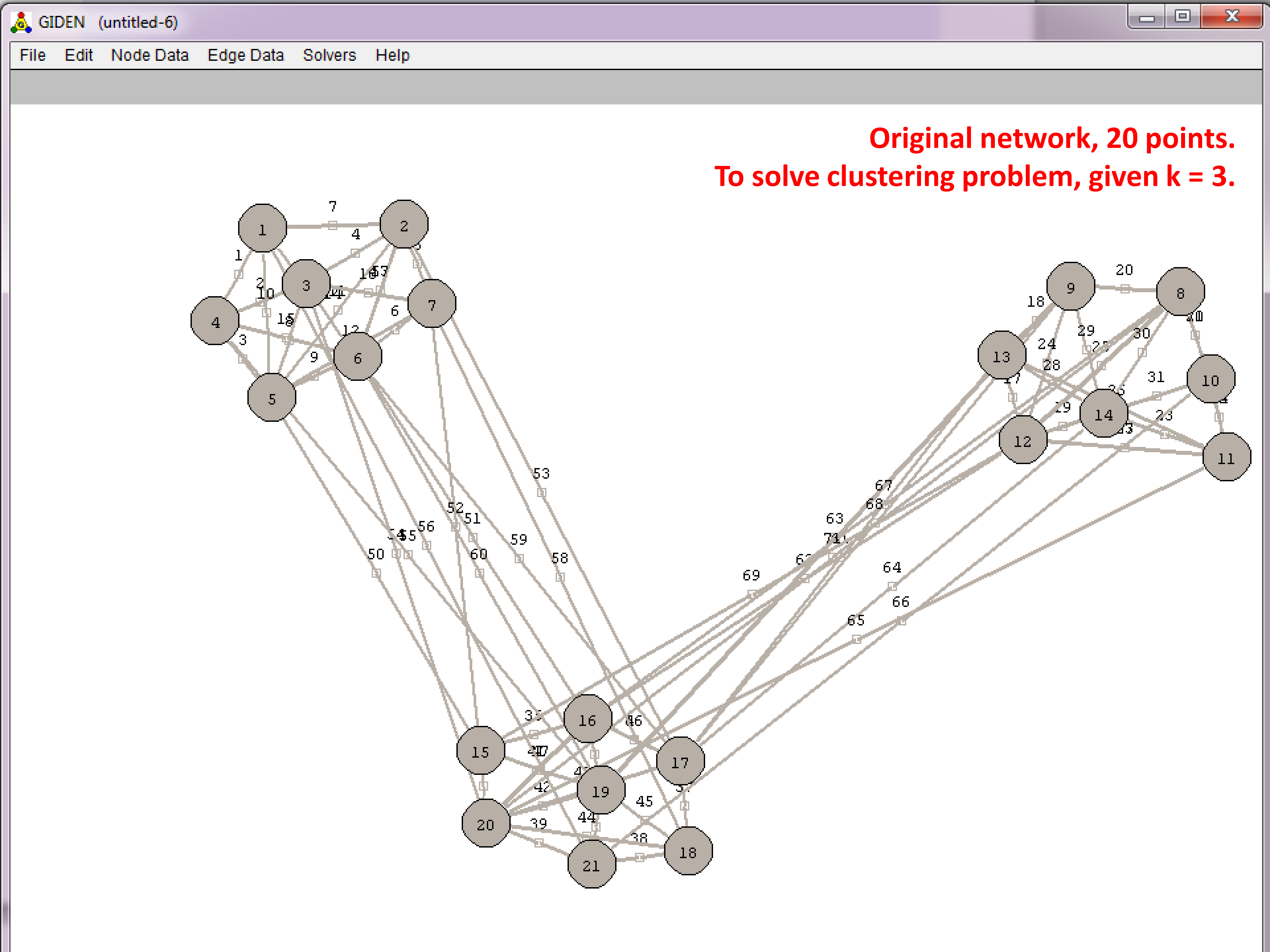
For every non-tree arc (k, l) of G; $c_{ij} \leq c_{kl}$ for every arc (i, j) contained in the path of T connecting nodes k and l.



- Kruskal's Greedy Algorithm

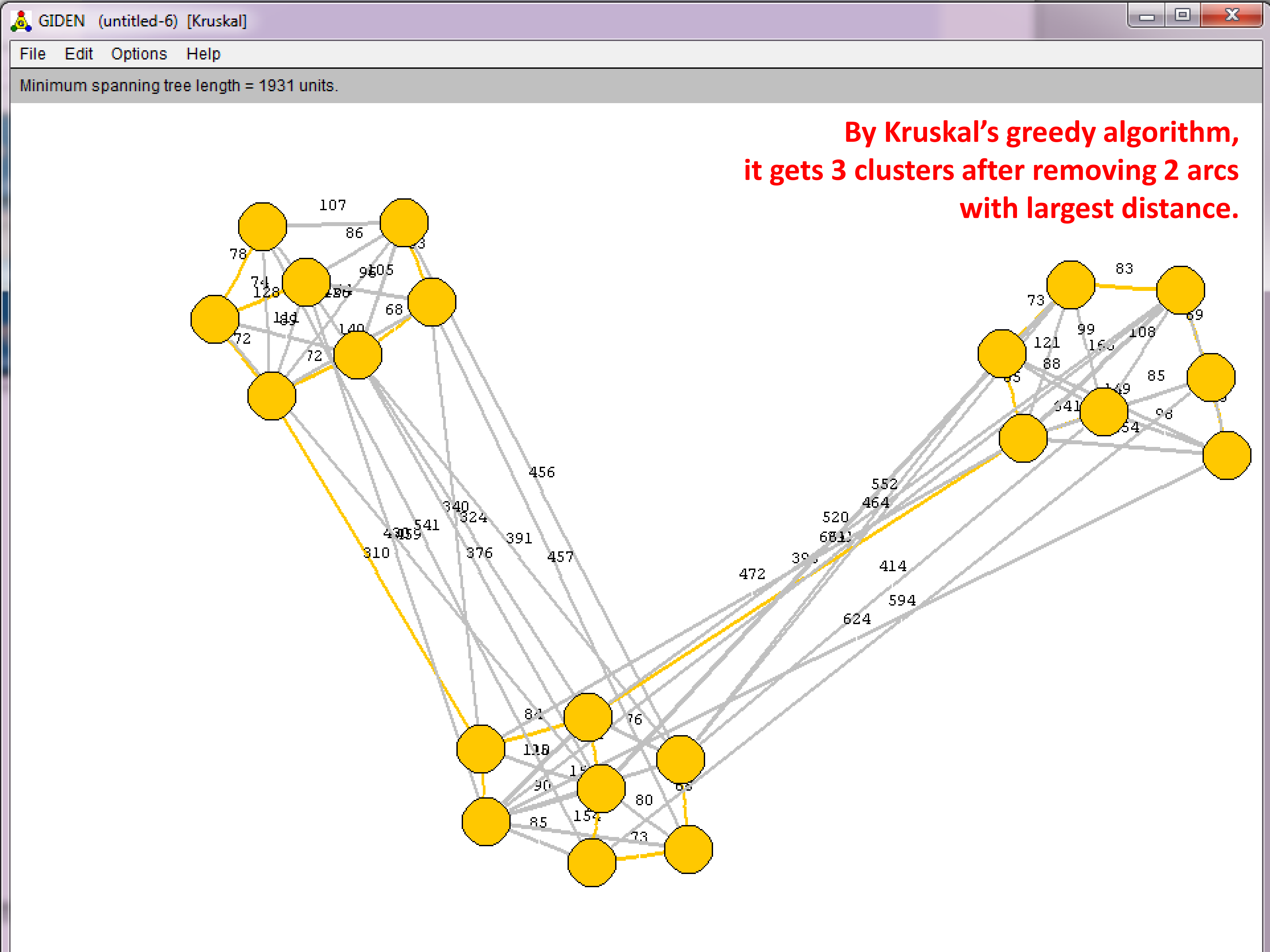
Both are saying the same thing, that every arc in T* must have the least cost.

By looking for T*, both ways can find k clusters after removing k-1 largest arcs, but they won't get the median points as k-median algorithm.



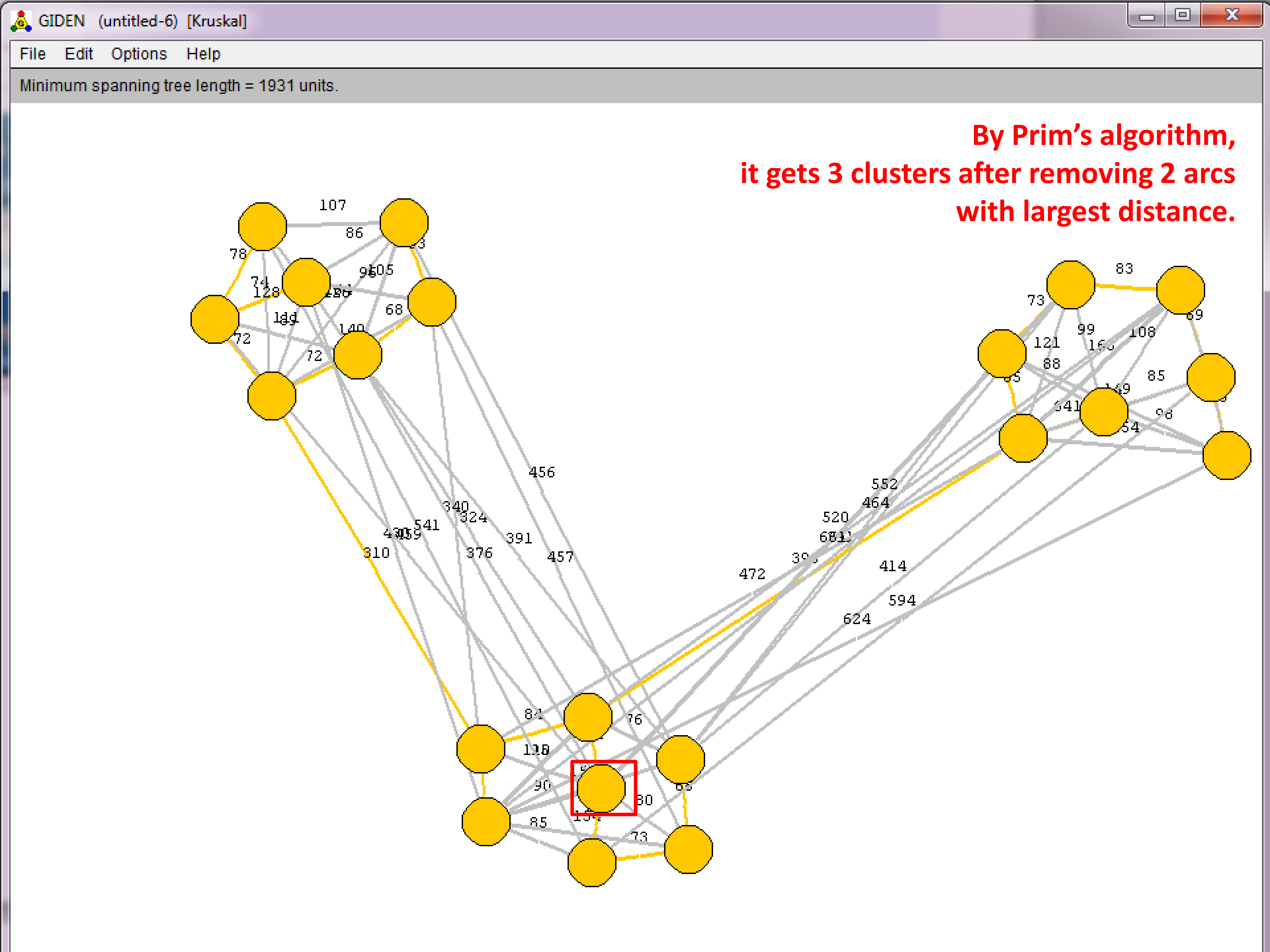
Original network, 20 points.

To solve clustering problem, given k = 3.



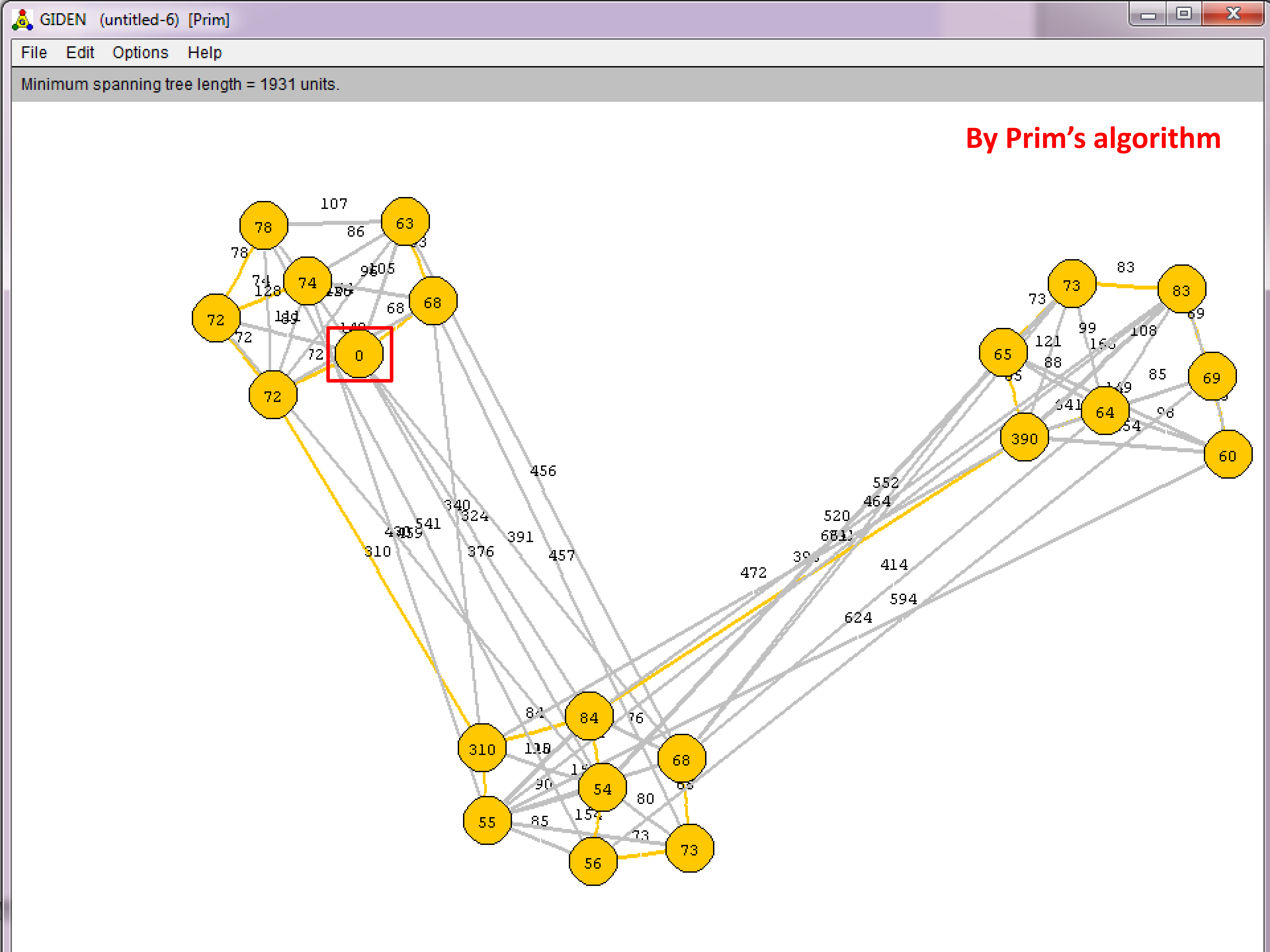
Minimum spanning tree length = 1931 units.

By Kruskal's greedy algorithm, it gets 3 clusters after removing 2 arcs with largest distance.



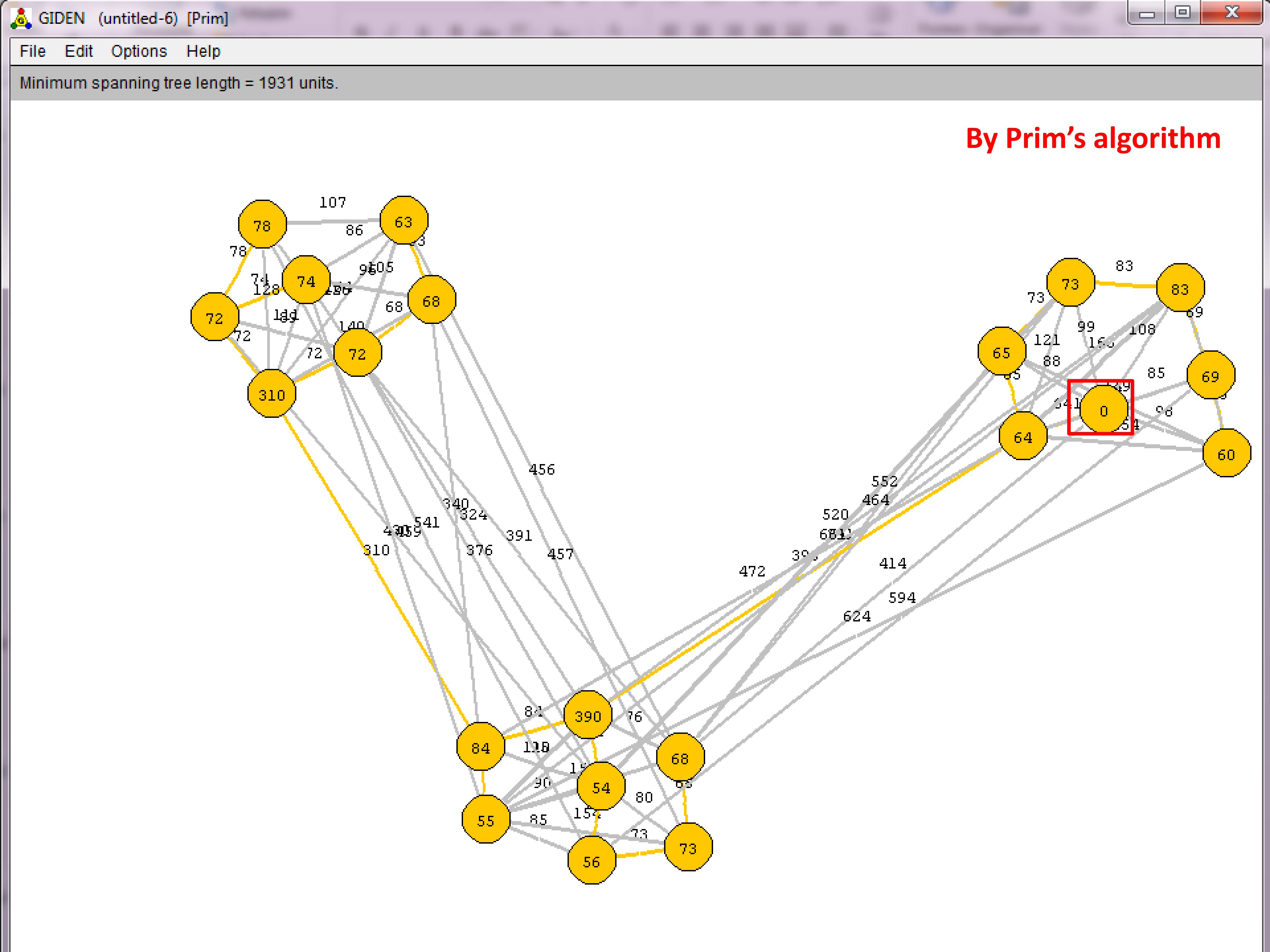
Minimum spanning tree length = 1931 units.

By Prim's algorithm, it gets 3 clusters after removing 2 arcs with largest distance.


GIDEN (untitled-6) [Prim]

Minimum spanning tree length = 1931 units.

By Prim's algorithm



Minimum spanning tree length = 1931 units.

By Prim's algorithm