2nd. International Conference on Applied and Theoretical Information Systems Research, December 27-29, 2012, Taipei, Taiwan

SOLVING LARGE LOCATION-ALLOCATION PROBLEMS BY CLUSTERING AND SIMULATED ANNEALING

Ferran Torrent,
University of Girona, Spain
ferran.torrent@udg.edu

Víctor Muñoz,
Newronia S.L., Spain
victor.munoz@newronia.com

Beatriz López,
University of Girona, Spain
beatriz.lopez@udg.edu

ABSTRACT

Globalization involves several facility location problems that need to be handled at large scale. Location-Allocation (LA) is a combinatorial problem in which the distance among points in the data space matter. Precisely, taking advantage of the distance property of the domain we exploit the capability of clustering techniques to partition the data space in order to convert an initial large LA problem into several simpler LA problems. Particularly, our motivation problem involves a huge geographical area that can be partitioned under overall conditions. We present different types of clustering techniques and then we perform a cluster analysis over our dataset in order to partition it. After that, we solve the LA problem applying simulated annealing algorithm to the clustered and non-clustered data in order to work out how profitable is the clustering and which of the presented methods is the most suitable.

Keyword: Clustering, location-allocation, problem partition, simulated annealing

1. INTRODUCTION

Location allocation (LA) problem is a two-step problem: first it consists in determining the location of a set of facilities given a set of customers; and second it reassigns the customers (demand) to the located facilities [9].

There are several types of LA problems, but the problem that concerns us is called Immobile LA (ILA) [16] which consists in determining the service, among a set of services, each facility should offer given a set of facilities with known positions and given a known demand. This problem is a k^n combinatorial problem where k is the number of possible services and n the number of facilities. Therefore, the applicability of optimization methods is tied up to the dimensionality of the problem.

Particularly, the problem we need to tackle is to decide which sport event (service) should be displayed in each bar (facility), with known position, as to maximize the satisfaction of the customers, i.e., each customer watches the desired match maximizing the global number of attendants to the bars.

The sport globalization implies that a lot of sport events are followed around the world causing that some matches are played simultaneously what implies that, often, not all of them can be displayed by a single premises or bar. Therefore, this sport globalization entails the decision problem of which is the optimum match for each bar to broadcast taking into account the competence.

To tackle the problem, we can take advantage of the particularities of it. The relationship between customers and facilities is measured by a distance function (we used the Euclidean distance) in the physical space. Customers tend to go to their closer facility that offers their desired service. Then, if the elements of the dataset are the coordinates of the facilities, thus they are points in the data space, a clustering analysis will detect groups in this dataset according to the distance between the elements and so it will partition the dataset.

While clustering will not give us the answer of the ILA problem (which service to assign to each facility), it offers the possibility of splitting the original problem into sub-problems and finding an approximation (the original problem, due to the size, cannot be solved by any optimization technique off the shelf). Then, the global solution can be achieved by combining the partial solutions. Since the relationship between customers and facilities is inversely to their distance, the global result would be quite approximate to the optimum. Therefore, one way to deal with the dimensionality of location allocation problem is partitioning the space with clustering techniques and finding the solution for each subspace separately. Obviously, the partition of the space strongly determines the quality of the overall result and each clustering technique results in different clusters. Regarding the ILA problem, we are looking for clusters that are separate and with more or less the same size, or at least that none of them has most of the elements to significantly minimize the complexity.

This paper is organized as follows. First we present some related work. Then, we describe the motivation problem. Next we present the method used for solving the problem describing the different algorithms used. Afterwards, we present the results achieved and a comparison between the algorithms is made. Finally, we end the paper with some conclusions and discussion about future work.

2. RELATED WORK

In [11, 12] the authors propose a model to formulate a location-allocation problem. In the model several facilities are static and LA is used to determine the location of some extra facilities to serve the demand. In these previous works the facilities offered the same service, whereas in this paper we propose a model for an ILA problem where all facilities are located and we have to decide the service each one offers.

In [16] the authors propose a mathematical model for an ILA problem that consists in locating service units in a pre-established set of immobile server locations to minimize the travel and queue time. Therefore, their problem consists in determining the number of service units of each immobile server, offering each facility the same service. This paper is different from [16] in that we have to determine the service each facility has to offer. Moreover, this paper differs from [11, 12, 16] in the dimensionality of the problem since the number of facilities we have to deal with is much higher. Therefore, the methods proposed to solve the respective problems are different. Hence in [11, 16] the authors propose heuristic methods we propose a combination of clustering methods (to reduce the complexity) and heuristic methods (to solve the problem). In [12] the authors propose an exact algorithm and an integer programming approach depending on the features of the problem.

In [8] the authors have to deal with large scale continuous location-allocation problem which complexity is very high. Due to this complexity, they examine three decomposition strategies to reduce the complexity of the problem where two of them use clustering to define the sub-problems. The clustering techniques explored in this paper are different from the examined in [8] due the nature of the problem since they are dealing with a continuous domain with a few facilities but we are dealing with a discrete domain with a lot of facilities.

3. THE MOTIVATION PROBLEM

As said in the previous section, the problem under concern is to decide the optimum match each bar has to broadcast given a set of matches and given the demand. The optimal solution is the one that maximizes equation (1) and satisfies the constraints described by equations (2), (3) and (4)

$$Fitness(q) = \sum_{i=1}^{N_{bars}} \sum_{j=1}^{N_{customers}} \frac{z_{ij}^q}{1+d_{ij}^2} \qquad (1)$$

$$\forall i \quad \sum_{j=1}^{N_{customers}} z_{ij}^q \le C_i \qquad (2)$$

$$\forall j \quad \sum_{i=1}^{N_{bars}} z_{ij}^q \le 1 \qquad (3)$$

$$x_i^q \ne M_j \rightarrow z_{ij}^q = 0 \qquad (4)$$

where z_{ij}^q is 1 (or 0) if the jth customer is assigned to the ith bar (or not), d_{ij}^q is the squared Euclidean distance between the jth customer and the ith bar, N_{bars} is the number of bars, $N_{customers}$ is the number of customers, C_i is the capacity of the ith bar and x_i^q and M_j are the match that broadcasts the ith bar (in the qth solution) and the desired match by the jth customer respectively. Therefore, the problem consists in maximizing the total number of customers weighting them by $\frac{1}{1+d_{ij}^2}$ (equation (1)) considering that the number of customers assigned to a bar do not overflow its capacity (equation (2)), that a customer cannot be assigned to more than one bar (equation (3)) and that a customer cannot be assigned to a bar that does not broadcast its desired match (equation (4)). Note that the satisfaction of the bars is not included in the problem since it is out of scope of our research.

4. THE METHOD

The problem we have to deal with is an optimization problem which main difficulty is its dimensionality since the datasets used have a huge number of bars. Therefore it is needed a strategy or a method that reduces the complexity of it. The method developed to tackle this problem consists of two steps:

- Clustering: clustering the data space to convert the initial problem into several smaller problems, reducing the complexity.
- Optimization: solving each sub-problem separately and then joining all sub-solutions to build the global solution.

In this section we describe the methods used in each step.

4.1.Clustering

First step of the method consists in clustering the data space (bars) to divide the problem into several sub-problems. Clustering techniques can be classified according to different criteria. For example, we propose the classification provided in Figure 1 based on [1]. In this paper we used some hard clustering techniques such as k-means or hierarchical clustering to provide a comparison of their performances. We discarded the analysis of soft clustering techniques as we want to divide the initial problem into separated problems to simplify its resolution. Fuzzy clusters introduce some complexity when combining the partial solutions. Thus, we left for future such kind of clustering techniques.

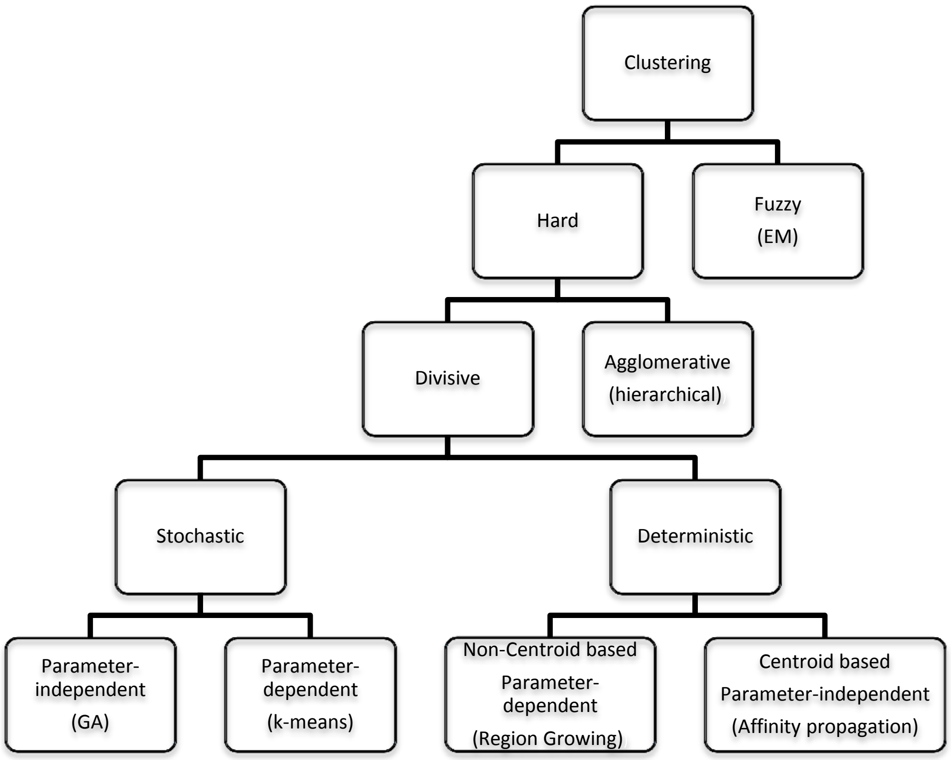
Clustering
Hard
Fuzzy (EM)
Divisive
Agglomerative (hierarchical)
Stochastic
Deterministic
Parameter-independent (GA)
Parameter-dependent (k-means)
Non-Centroid based Parameter-dependent (Region Growing)
Centroid based Parameter-independent (Affinity propagation)

Figure 1. Taxonomy of the clustering algorithms

Regarding to hard clustering techniques we have chosen one technique for each group of hard clustering methods according to Figure 1. The performance of the algorithms has been analyzed using the running time, the Calinski Index (CI) [7], the Davies-Bouldin Index [14], the number of clusters and the maximum cluster size:

- Running time: amount of time (in seconds) needed to perform the clustering analysis.
- CI: clustering quality index that measures the differences inside each cluster and the differences between clusters. The greater CI the better.
- DBI: index used, like CI, to evaluate the quality of a clustering analysis that measures the differences within and between clusters. The lower DBI the better.
- Number of clusters: number of clusters from the clustering analysis. The optimal number depends in each case, but a great number of clusters can mean that the algorithm over-divides the data space and a low number of clusters may not simplify the problem as much as we desire.
- Maximum cluster size: number of elements of the biggest cluster.

K-means

K-means [13, 15], was proposed in 1960s, nevertheless it is still a very useful algorithm because of its simplicity. It consists of planting some seeds across the data space and then building clusters assigning to them the closer elements to the seed of each cluster. After that it iteratively recalculates the centroids of every cluster and re-assigns elements to the cluster with the closer centroid. Its main weakness is the problem of initialization [4]. It needs the user to specify the number of clusters, and the final result is highly dependent on the initial position of the seeds. Thus, it is necessary to execute the algorithm several times with different initializations in order to reach an appropriate result.

Algorithm 1. K-means1 algorithm

```
Require: $N_{iterations}$
 1: Choose the number of clusters $K$ randomly
 2: Select $K$ geographical coordinates randomly (cluster centroids)
 3: for $iteration = 1$ to $N_{iterations}$
 4:   Assign every bar to the closest centroid
 5:   if $cluster_k$ is empty
 6:       Assign it a new random centroid $c_k$
 7:   else
 8:       $c_k = \frac{\sum_{j=1}^{N_k} x_j}{N_k}$
 9:   end if
10: end for
```

Another problem of k-means is the problem of empty clusters [4]. If the seeds are initially set across the data space randomly, some of them generally will be placed in empty regions like in the middle of the sea. Thus, the clusters will be empty. There are two ways to deal with this problem: deleting the empty clusters, or moving them to another place. If the clusters are deleted, this induces the algorithm to reduce the number of clusters. The second option, (the one we implemented) as Algorithm 1 shows, keeps the number of clusters by moving the empty clusters to another place. Henceforth we call this algorithm k-means1.

Another way to deal with the problem of empty clusters is avoiding generating them. If step 2 is substituted by selecting randomly K objects as centroids, the resulting clusters will have, at least, one element. However, this methodology generates another problem because if it selects randomly K objects, and there are very dense regions, there is a high probability to choose more than one object for each dense region. This provokes that the algorithm will divide these homogeneous regions into several smaller regions. We will refer to this algorithm k-means2.

Lloyd's algorithm [15], follows the philosophy of avoiding the generation of empty clusters. It follows Algorithm 1 scheme but, instead of step 2, it divides into K groups the dataset and then computes the centroid of each group. These centroids are the seeds. This procedure solves the problem of selecting several elements from the same cluster as seeds.

Region growing

Region growing [10], is a region based algorithm. It is widely used for image segmentation, but we can use it also for our purposes. It starts from one or more initial points (seeds) and expands them across the space with the same features of the initial point. The region expansion can be done in parallel (every region expands itself at the same time) or one after the other. We have chosen the second way (see Algorithm 2) because it does not require the number of clusters and it does not depend on the initial position of the seeds. The algorithm just needs the maximum allowed distance between objects D_{max} to be included in the same cluster. Since we are dividing the data space and clustering the data set, this threshold is used as a parameter of control. The region growing can be classified as stochastic or deterministic, depending on if it plants seeds across the data space and then increases each region in parallel or if it grows up each region one after the other what avoids the need of planting seeds (see Figure 1). We followed the deterministic approach for two main reasons. First, k-means and Lloyd's algorithm are examples of stochastic methods. And second, the implementation avoids the problem of the number of clusters by introducing a distance threshold which is quite more intuitive to set. For example, a customer is not likely to walk to a facility that is 1km away from him.

Algorithm 2. Region growing

```
Require: $D_{max}$
1:  Assign to each object all objects closer than $D_{max}$
2:  Create a list of non-assigned objects $List_{ObjLeft}$
3:  while $List_{ObjLeft}.Count > 0$ do
4:    $object = List_{ObjLeft}[0]$
5:    Create a new cluster and add, to it, $object$, the neighbors, the neighbors of the neighbors and so on
6:    Remove assigned objects from $List_{ObjLeft}$
7:  end while
```

Agglomerative hierarchical clustering

Hierarchical clustering [13], consists in doing a hierarchy of clusters. It starts making a cluster for each object and as it goes up through the hierarchy it joins pairs of clusters (see Algorithm 3). The algorithm terminates when the needed jump to join the next pair of clusters is greater than a certain threshold Th_{jump}. The jump is the relative increase of maximum distance to join a pair of clusters. This threshold can be used as a parameter of control. After some experiments we set $Th_{jump} = 0.05$ as it was one of the values with the best results.

Algorithm 3. Agglomerative hierarchical clustering

```
Require: $Th_{jump} = 0.05$
1:  Create one cluster per object
2:  while $N_{clusters} > 1$ do
3:    Calculate distances between clusters
4:    Find the minimum distance $d_{min}$ between clusters
5:    Join the pair of closest clusters
6:    $jump = \frac{d_{min} - d_{old}}{d_{min}}$
7:    if $jump > Th_{jump}$ then
8:      Save the new clustering and exit the while loop
9:    end if
10: end while
```

Genetic algorithm based clustering

All genetic algorithms consist in the combination of some evolutionary operators (selection, crossover and mutation) to improve an amount of solutions (chromosomes) coded into strings, [3, 5]. The genetic algorithm based clustering presented here (see Algorithm 4) is able to search the optimum number K of clusters and their positions [14].

Each chromosome contains the $\boldsymbol{z}_{i,1}, \cdots, \boldsymbol{z}_{i,L}$ centroids of the clusters, where $\boldsymbol{z}_{i,k}$ is the kth centroid of the ith chromosome and L is the length of every chromosome. However, some centroids are empty or null centroids as not all chromosomes have L different clusters (see Figure 2).

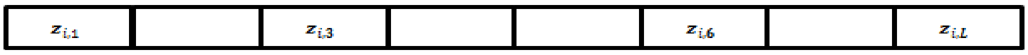

Figure 2. Chromosome example

The reproduction step used in this algorithm uses crossover and mutation operators. Reproduction consists in breeding two new solutions from a pair of parent solutions. Regarding crossover, a single point crossover is used. It consists of randomly selecting a number p with $1 < p < L$. Then, two children are created assigning to the first child the first p *genes* of the first parent and the last $L - p$ *genes* of the second parent. The second child is created taking the first p *genes* from the second parent and the last $L - p$ *genes* from the first parent. Regarding mutation, each (non-empty) centroid $\boldsymbol{z}_{i,k}$ of each chromosome is changed with probability $\mu_{mutation}$ according equation (5), where + or - signs occur with equal probability and $\boldsymbol{\delta}$ is a uniform random number between 0 and 1.

$$\boldsymbol{z}_{i,k} = \begin{cases} \boldsymbol{z}_{i,k} \cdot (1 \pm 2\boldsymbol{\delta}) & z_{i,k} \neq 0 \\ \pm 2\boldsymbol{\delta} & z_{i,k} = 0 \end{cases} \tag{5}$$

Algorithm 4. Genetic algorithms based clustering

```
Require: $Th_{jump} = 0.05$
 1:  Create one cluster per object
 2:  while $N_{clusters} > 1$ do
 3:     Calculate distances between clusters
 4:     Find the minimum distance $d_{min}$ between clusters
 5:     Join the pair of closest clusters
 6:     $jump = \frac{d_{min} - d_{old}}{d_{min}}$
 7:     if $jump > Th_{jump}$ then
 8:        Save the new clustering and exit the while loop
 9:     end if
10:  end while
```

The selection of parents is done according to the roulette selection that consists in generating a number of copies of every chromosome proportional to the fitness and put them into a pool. Then, pairs of them are selected randomly, having more chances those copies from the chromosomes with a higher fitness.

Once a new generation is bred, the best P individuals from the previous and the current generations are selected to breed the next generation, deleting the worst individuals. We stated a population of 100 individuals. The fitness of the individuals is computed as $1/_{DBI}$, where DBI is the Davies-Bouldin Index [14].

Affinity propagation

Affinity propagation is a recent clustering technique developed by Frey and Dueck [2]. It consists in that the elements of the dataset exchange messages voting the most representative elements (exemplars). Figure 3 illustrates how the elements of the dataset exchange messages and vote the exemplars. An important strength of this algorithm is that, as GA based clustering, it does not need that the user introduces any parameter to perform clustering.

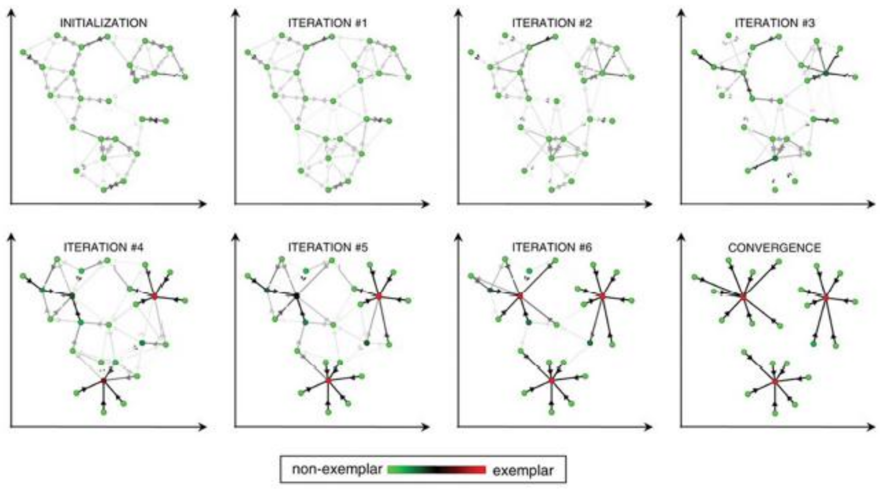

Figure 3. Affinity propagation illustration, [2]

4.2.Optimization

The second step of the method consists in finding the optimal solution to each sub-problem generated by the clustering step. The chosen algorithm is the heuristic method called Simulated Annealing (SA). It consists in given an initial solution, improve it iteration after iteration, generating a neighbor solution and dropping the worst of both. Nevertheless, sometimes some bad movements that consist of dropping the best of both solutions are allowed to avoid SA of getting stacked in local optimums or flat regions.

Despite SA is applicable to a vast variety of problems, it has the disadvantage that it needs a coordinate system of the solution space to generate neighbor solutions. The solution space of the problem we concern does not have such coordinate system. Thus, we used SA approach proposed in [6] that avoids the coordinate system need using a new neighbor function.

5. Experimentation

5.1. Experimentation set-up

For the experiments described in this paper we have used three datasets of bars of different sizes that include the geographical position and capacity of each bar:

- Dataset 1: 373 bars and 6678 customers
- Dataset 2: 458 bars and 8258 customers
- Dataset 3: 1925 bars and 34954 customers

The demand has been simulated generating a random number of customers between 0 and 30 around each bar. The geographical position of each customer has also been randomly set using a 2D Gaussian distribution centered on the bar. Then a single match from a list of matches is randomly assigned to each customer. In the mentioned list of matches each match is related with a probability of being assigned to a customer and this probability refers to the estimated audience percentage of the match. Simulating the demand as said before the densest regions will be placed in those regions where the density of bars is higher. The computer used uses Windows 7 and has an Intel® Core™ i5 CPU@2.80 GHz and 8.00 GB of RAM.

5.2. Results

Tables 1-3 show the clustering results obtained by the beforehand presented algorithms in the three different sized datasets. As it can be seen, the different algorithms provide different clustering analysis (except GA and hierarchical clustering for datasets 1 and 2); then we can see that the faster algorithm is region growing and it is also the algorithm that experiment a lower growing of the elapsed time when the dataset grows up. On the other hand, GA based clustering, k-means algorithms and especially Lloyd’s algorithm, experiment and exponential growing what implies that they may be unfeasible for very big datasets.

Another result showed by Tables 1-3 is that k-means and Lloyd’s algorithms find a lot of clusters and achieve good CI (the higher CI the better) values respect the other techniques. However, region growing also finds a lot of clusters (especially for small D_{max}) but does not achieve good CI values (except for the third dataset). This last result is because region growing does not tend to find circular clusters as it does not use the centroid of the clusters to assign elements to the clusters. Therefore, CI is not useful to evaluate clustering analysis provided by region growing since to compute the CI the centroids of the clusters are used to evaluate the differences within and between clusters. On the other hand, DBI (the lower DBI the better) indicates us that region growing approach provides good clustering analysis, even better than the other algorithms, reinforcing the premise that CI is not useful to evaluate region growing clustering analysis. But also, the best DBI results are usually achieved for clustering analysis where a lot of clusters are found and vice versa. This relationship also exists with CI (except for results from region growing).

Table 1. Clustering results for dataset 1. Best results are in bold face.

Clustering alg.	CI	DBI	Num. of clusters	Max cluster	Time (s)
Genetic	181.11	0.563	8	230	34.543
Hierarchical	181.11	0.563	8	230	0.125
K-means1	1056.95	0.460	101	43	12.987
K-means2	**1146.58**	0.350	191	34	4.351
Lloyd’s algorithm	759.19	0.391	163	**33**	17.813
RG $\boldsymbol{D_{max}} = \mathbf{0.1}$km	282.77	**0.330**	121	62	**0.005**
RG $\boldsymbol{D_{max}} = \mathbf{0.2}$km	48.54	0.394	39	231	0.007
RG $\boldsymbol{D_{max}} = \mathbf{0.5}$km	35.03	0.499	14	405	0.021
RG $\boldsymbol{D_{max}} = \mathbf{1.0}$km	6.26	0.381	3	371	0.029
Affinity propagation	325.62	0.782	16	47	3.172

Table 2. Clustering results for dataset 2. Best results are in bold face.

Clustering alg.	CI	DBI	Num. of clusters	Max cluster	Time (s)
Genetic	257.45	0.664	8	234	34.574
Hierarchical	257.45	0.664	8	234	0.136
K-means1	**1194.38**	0.462	125	55	17.336
K-means2	823.29	0.522	159	53	4.564
Lloyd’s algorithm	628.10	0.473	170	**44**	18.081
RG $\boldsymbol{D_{max}} = \mathbf{0.1}$km	419.74	**0.272**	172	73	**0.004**
RG $\boldsymbol{D_{max}} = \mathbf{0.2}$km	61.26	0.348	71	248	0.008
RG $\boldsymbol{D_{max}} = \mathbf{0.5}$km	35.03	0.499	14	405	0.027
RG $\boldsymbol{D_{max}} = \mathbf{1.0}$km	2.91	0.466	2	457	0.043
Affinity propagation	439.93	0.742	20	69	3.115

Table 3. Clustering results for dataset 3. Best results are in bold face.

Clustering alg.	CI	DBI	Num. of clusters	Max cluster	Time (s)
Genetic	1346.45	0.507	18	548	279.138
Hierarchical	4745.74	0.451	48	395	69.871
K-means1	18168.30	0.390	185	131	471.654
K-means2	15825.53	0.342	834	**39**	141.654
Lloyd’s algorithm	44204.58	0.391	654	**39**	1629.672
RG $\boldsymbol{D_{max}} = \mathbf{0.1}$km	**199033.78**	**0.100**	1082	**39**	**0.018**
RG $\boldsymbol{D_{max}} = \mathbf{0.2}$km	81860.09	0.174	770	102	0.045
RG $\boldsymbol{D_{max}} = \mathbf{0.5}$km	11297.91	0.216	401	257	0.098
RG $\boldsymbol{D_{max}} = \mathbf{1.0}$km	7047.35	0.186	258	364	0.133
Affinity propagation	2819.31	0.565	28	382	49.415

Focusing on the size of the biggest cluster found in each case we can see that the smallest ones are achieved by k-means and Lloyd’s algorithms and for region growing with a small enough D_{max}. Finding small clusters is, a priori, beneficial since it reduces the complexity of the problem.

Table 4 shows the quality (fitness) of the solutions found by SA using the data clustered by the beforehand presented clustering algorithms and the original data (non-clustered) and the elapsed time by SA to find such solutions. Note that we have not included the clustering time because once the clustering is done, it does not need to be computed again if there are the same bars, but SA has to be run every time that there are different matches and a different demand.

Table 4. Global solutions found to the ILA problem. Best results are in bold face.

Technique	**Dataset 1**		**Dataset 2**		**Dataset 3**	
	Fitness	**Time (s)**	**Fitness**	**Time (s)**	**Fitness**	**Time (s)**
Non-clustered	**6354.23**	456.442	**7816.24**	569.964	28778.06	2261.027
Genetic	6207.79	167.961	7624.84	196.576	29041.34	669.030
Hierarchical	6203.76	169.994	7632.91	205.237	29160.86	507.649
K-means1	5116.04	**15.648**	6311.87	16.456	28877.25	200.576
K-means2	4534.77	21.321	6120.70	**14.702**	23972.13	**76.919**
Lloyd's algorithm	4724.83	19.790	5983.05	25.618	25306.32	93.778
RG $D_{max} = 0.1$km	5012.65	19.000	5968.20	20.947	22371.45	77.392
RG $D_{max} = 0.2$km	5907.13	157.465	7113.03	188.358	24888.39	106.163
RG $D_{max} = 0.5$km	61.53.46	215.971	7726.98	512.064	28192.99	292.310
RG $D_{max} = 1.0$km	6275.37	451.653	7801.64	569.461	29091.78	460.817
Affinity propagation	6345.02	18.611	7794.61	24.546	**29172.79**	504.292

If we take a look at Table 4 we can see that the solution found using the clustered data by those algorithms that found a lot of clusters (such as k-means2, Lloyd's algorithm and region growing with small D_{max}) has a poor quality, but also the search time is very low. Thus, over-clustering the dataset reduces the search time needed by SA but also reduces the quality of the final solution. Moreover, we can say that CI and DBI are useless two predict how profitable is a clustering analysis to divide a dataset because, those clustering analysis with more clusters had the best CI and DBI values, but also the worst solutions found by SA.

In general we can say that clustering the facilities dataset to convert the initial problem into several smaller problems reduces the time needed to find a solution to the problem. This time reduction depends on the size of the clusters found, especially for the biggest clusters. Thus, considering the size of the biggest cluster one can figure out how this reduction is.

Focusing on the results of datasets 1 and 2 we can say that clustering the dataset reduces the time needed to find a solution to the ILA problem but also reduces the quality of the solution. But with an appropriate clustering analysis the quality reduction is minimum and the search time reduction is quite important, as happens with affinity propagation. But if we focus on dataset 3 results we can see that clustering the dataset allows SA to find a better solution in less time as happens with GA clustering, hierarchical clustering, k-means1 or affinity propagation. This result is due we use a heuristic method to find a solution, but heuristic methods do not guarantee that the optimal is found. Thus, when we apply SA to dataset 3, the solution space is too large for the algorithm to find a good solution, but if we divide the solution space and we apply SA to each subspace (cluster) it is able to find very good sub-solutions for each subspace.

Finally, we can say that affinity propagation is the algorithm that finds the best partition in all cases since the final solutions, found using its clustering analysis, are better than the solution found using other clustering analysis. Therefore, we finally decided to use this algorithm to perform the problem partition.

6. CONCLUSIONS AND FUTURE WORK

Globalization is posing new challenges to optimization techniques, due to the increase in size of the problem to be solved. In this paper we tackle the problem of deciding which match to show in a bar, when multiple sport events are disputed simultaneously, and there are costumers with different preferences. The problem of deciding the service each facility should offer given a set of facilities and a set of demand points (or customers) is modeled as the immobile location-allocation problem. The objective is to maximize the global covered demand while minimizing the distance between customers and facilities.

The paper includes a method to solve such problem that consists of two steps: (i) dividing the problem into sub-problems and (ii) solving each sub-problem independently and then joining every sub-solution to build the global solution. For the first step, we proposed the use of some clustering techniques for which we gave a taxonomy. For the second step we proposed the heuristic method SA according to [6].

We conduct experimentation on a few datasets, and the results show that the use of clustering techniques (especially affinity propagation) reduces the seek time and also improves the quality of the final solution found by SA when the dataset is big enough. For small datasets it entails a small reduction of the quality of the solution.

The complexity of our problem is mostly given by the number of facilities and we have used clustering analysis to reduce this complexity. However, the number of customers also increases the complexity of the problem, specifically over the allocation process. To reduce the number of customers, demand aggregation could be used and it would be interesting to see how clustering analysis can be used in minimizing the errors resulted from aggregation.

7. ACKNOWLEDGEMENTS

This research project has been partially funded through the project labeled DPI2011-24929 and BRGR Scholarship of the University of Girona granted to Ferran Torrent.

8. REFERENCES

[1] A.K. Jain and M.N. Murty and P.J. Flynn, Data clustering: a review. *ACM Computing Surveys (CSUR),* 31(3), 264-323, 1999.

[2] B.J. Frey and D. Dueck, Clustering by passing messages between data points. *Science*, 315(5814), 972-976, 2007

[3] D.E. Goldberg. *Genetic algorithms in search, optimization and machine learning,* Adison-Wesley, 1989.

[4] D. Arthur and S. Vassilvitskii, K-means++: the advantages of careful seeding. *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, 1027-1035, Society for Industrial and Applied Mathematics, Philadelphia, 2007.

[5] E.R. Hruschka and R.J.G.B. Capello and A.A. Freitas and A.C. Ponce Leon F. de Carvalho, A survey of evolutionary algorithms for clustering. *IEEE Transactions on Systems, Man and Cybernetics – Part C: Applications Reviews*, 39(2), 133-155, 2009.

[6] F. Torrent and V. Muñoz and B. López, Exploring genetic algorithms and simulated annealing for immobile location-allocation problem. *15th International Conference of the Catalan Association of Artificial Intelligence (CCIA 2012),* Spain: Alicante, 2012.

[7] G.R. Raidl and J. Gottlieb. *Evolutionary computation in combinatorial optimization strategies,* Springer, Vienna, 2005.

[8] J. Brimberg and P. Hansen and N. Mladenovic, Decomposition strategies for large-scale continuous location-allocation problems. *Journal of Management Mathematics*, 17(4), 307-316, 2006.

[9] L. Cooper, Location-allocation problems. *Operations Research,* 331-343, 1963.

[10] L.G. Shapiro and G.C. Stockman. *Computer Vision*, Prentice-Hall, New Jersey, 2001.

[11] R. Aboolian and O. Berman and Z. Drezner, Location and allocation of service units on a congested network. *IIE Transactions*, 40(4), 422-433, 2008.

[12] R. Aboolian and Y. Sun and G. Koehler, A location-allocation problem for web services provider in a competitive market.

[13] R.O. Duda and P.E. Hart and D.G. Stork. *Pattern Classification*, Wiley, New York, 2001.

[14] S. Bandyopadhyay and U. Maulik, Genetic clustering for automatic evolution of clusters and application to image classification. *Pattern Recognition*, 35(6), 1197-1208, 2002.

[15] S. Lloyd, Least Square Quantization in PCM. *IEEE Transactions on Information Theory,* 28(2), 129-137, 1982.

[16] S.H.R. Pasadideh and S.T.A. Niaki, Genetic application in facility location problem. *Journal of Intelligent Manufacturing*, 23(3), 651-659, Springer, 2010.