University of Szeged
Department of Computer Algorithms and Artificial Intelligence

Online algorithms for clustering problems

Summary of the Ph.D. thesis

by

Gabriella Divéki

Supervisor
Dr. Csanád Imreh

University of Szeged
Ph.D. School in Computer Sciences

Szeged
2014

-

1 Introduction

In the practice often arise such optimization problems where the input – the numbers which define the problem – is known piece by piece, it is also unknown if there are any more request points. These problems are called *online problems* and the so called *online algorithms* which solve them can process the input in a serial fashion without having the entire input available from the start and they are not able to "see the future". In contrast, the optimal offline algorithm can view the sequence of requests in advance. An online algorithm is forced to make decisions that may later turn out not to be optimal. The study of online algorithms has focused on the quality of decision-making that is possible in this setting.

Two basic methods are used to measure the effectiveness of online algorithms. One of the possibilities is analyzing an average case. We have to assume some kind of probability distribution on the possible input and for this distribution the expected value of the algorithm is examined. The main disadvantage of this method is that the distribution of the input is usually unknown.

Another approach to this problem is the worst case analysis, which is called competitive analysis. It compares the relative performance of an online and offline algorithm for the same problem instance. Specifically, the *competitive ratio* of an algorithm, is defined as the worst-case ratio of its cost divided by the optimal cost, over all possible inputs. The competitive ratio of an online problem is the best competitive ratio achieved by an online algorithm.

Let $A(I)$ denote the value of the objective function of an arbitrary online algorithm A on the input I. The value of the objective function of the optimal offline algorithm on the input I is $OPT(I)$. Using this system of notations, we may define the competitiveness as follows: the algorithm A is c-competitive if $A(I) \leq c \cdot OPT(I)$ is valid for every input I. The factor c is called the competitive ratio of A if c is the least such number.

We are studying a part of the wide field of online algorithms, the clustering problems. The request points need to be grouped: assigned to a cluster or a facility, while a given objective function, that depends on the distance between points in the same cluster, is minimized. The online clustering algorithm maintains a set of clusters, where a cluster is identified by its name and the set of points already assigned to it. Each

point must be assigned to a cluster at the time of arrival; the chosen cluster becomes fixed at this time. The clusters cannot be merged or split. The costs are based on the number of clusters and their properties, and they depend on the exact specification of the problem.

Many objective functions can be defined; in this work two main groups are considered: those which depend on the size of the cluster and those which cost depends on the distance of the demand points from the facility. In this work new models are studied which have not been examined yet. We present a solving algorithm for every considered model and determine its competitive ratio. Furthermore, we provide lower bounds for the competitive ratio of any online algorithm which solves the given model.

2 Clustering problems

In the field of online clustering problems the researchers used to study algorithms for unit sized clusters and the first more general approach to the problem can be found in [2] which deals with variable sized clusters in 1-dimension and the cost is a constant setup cost plus the diameter of the cluster. In this work the extensions of this model are considered.

We study the 1-dimensional and the 2-dimensional variants of the online clustering with variable sized clusters problem which are presented in [3], [5] and [6]. The clusters are intervals (squares in 2D, since we use the l_∞ norm), the cost of each cluster is the sum of the constant setup cost scaled to 1 and the square of the length of the interval (side of the square in 2D). The goal is to minimize the total cost of the clusters.

We consider two variants, both having property that a point assigned to a given cluster must remain in this cluster, and clusters cannot be merged or split. In the strict variant, the size and the location of the cluster must be fixed when it is initialized. In the flexible variant, the algorithm can shift the cluster or expand it, as long as it contains all the points already assigned to it.

The basic idea behind many of our algorithms is the algorithm *GRID* defined in [7] which covers the d-dimensional space with d-dimensional unit cubes. When a demand point arrives which does not belong to an existing cluster, the algorithm opens a new cluster: the closed cube of the grid.

2.1 Clustering problems in 1 dimension

We consider first the offline problem where the whole input is given in advance and we offer a simple dynamic programming algorithm DP to solve the problem optimally.

The input is n request points $(x_1, ..., x_n)$. The dynamic programming algorithm uses an algorithm for the variation of the k-median problem and is shown in Algorithm 1.

Algorithm 1 Algorithm DP

- The request points are sorted by their coordinates in ascending order.
- Define the subproblem $F(i, r)$ $(i \geq r)$: the first i request points are divided into r clusters. Then the optimal cost of the clustering problem is $min_r(F(n, r) + r)$.
- The values of $F(i, r)$ can be calculated by the following recursions:
  $F(i, 1) = (x_i - x_1)^2$
  $F(i, r) = min_{j=r}^{i}\{F(j - 1, r - 1) + (x_i - x_j)^2\}$

In the strict variant of the clustering problem, the size and the exact location of the cluster must be fixed when it is initialized. We consider both the online and semi-online versions. "Semi-online" usually means that the algorithm knows something about the future demand points. In our case the points are sorted in ascending order. For the solution of the above problem we offer the algorithm $GRID_a$ where the size of the intervals covering the line is a parameter a. When a request point x arrives which does not belong to an existing cluster, the algorithm opens a new cluster: the interval $[ka, (k + 1)a]$ where $k = \lfloor x/a - 1 \rfloor$.

In the semi-online strict model the points arrive in ascending order. We propose algorithm $SOSM_a$ to solve this problem.

Algorithm 2 Algorithm $SOSM_a$

1. Let p be the new point.
2. If the algorithm has a cluster which contains p, then assign p to that cluster.
3. Else, open a new cluster $[p, p + a]$ and assign p to the new cluster.

In the flexible variant, the algorithm can shift the cluster or expand it, as long as it contains all the points assigned to it. We offer the algorithm $FGRID_a$ for the solution of this problem.

Algorithm 3 Algorithm $FGRID_a$

1. Let p be the new point.
2. If the algorithm has a cluster whose current associated interval contains p, then assign p to that cluster and do not modify the cluster.
3. Else, consider the cell from the grid which contains p.
   (a) If this cell does not have a cluster, then open a new cluster and assign p to the new cluster. This new cluster consists of a single point p.
   (b) Otherwise, extend the cluster contained in the interval to cover p.

We note that a similar modification to the algorithm $SOSM_a$ like in the online case (modification of $GRID_a$ that led to the algorithm $FGRID_a$) does not result in a better competitive ratio than 2 (like in the online case with algorithm $FGRID_a$).

Furthermore, we provide lower bounds for the competitive ratio of any online algorithm which solves the given model.

We summarize our results published in [3] for the strict model in Thesis 1 and for the flexible model in Thesis 2 below.

Thesis 1

Lemma 1 *The offline problem can be solved optimally by the dynamic programming algorithm DP.*

Theorem 1 *The competitive ratio of algorithm $GRID_a$ is*

$$\max\{F(\lfloor -2+\sqrt{4+\frac{1}{a^2}}\rfloor), F(\lceil -2+\sqrt{4+\frac{1}{a^2}}\rceil), 2+2a^2\}$$

where $F(k)=\frac{(k+2)(1+a^2)}{1+k^2a^2}$, $k\geq 1$.

Corollary 1 *The smallest competitive ratio of $GRID_a$ is obtained if $\frac{1}{2\sqrt{2}}\leq a\leq\frac{1}{\sqrt{2}}$, then the competitive ratio of the algorithm is 3.*

Theorem 2 *The competitive ratio of any online algorithm for the strict model is at least* 2.3243.

Theorem 3 *The competitive ratio of algorithm* $SOSM_a$ *is*

$$\max\{F(\lfloor -1+\sqrt{1+\frac{1}{a^2}}\rfloor), F(\lceil -1+\sqrt{1+\frac{1}{a^2}}\rceil), 1+a^2\}$$

where $F(k)=\frac{(k+1)(1+a^2)}{1+k^2a^2}$, $k \geq 1$.

Corollary 2 *The smallest competitive ratio of* $SOSM_a$ *is accomplished if* $\frac{1}{\sqrt{5}} \leq a \leq 1$, *then the competitive ratio of the algorithm is* 2.

Theorem 4 *The competitive ratio of any semi-online algorithm for the strict model is at least* 1.6481.

Thesis 2

Theorem 5 *The competitive ratio of algorithm* $FGRID_a$ *is 2 if* $\frac{1}{\sqrt{5}} \leq a \leq 1$.

Theorem 6 *The competitive ratio of any online algorithm for the flexible model is at least* 1.2993.

Theorem 7 *The competitive ratio of any semi-online algorithm for the flexible model is at least* 1.1991144.

2.2 Clustering problems in higher dimensions

2.2.1 Clustering problems in 2 dimensions

Similarly to the previous section, points of the now 2-dimensional Eucledian space arrive one by one and the algorithm has to define a cluster for them: an already opened cluster or a new one. The cost of each cluster is the sum of the constant setup cost scaled to 1 and the square of the length of the side of the cluster (square) and the objective function is to minimize the total cost of all clusters.

We study the strict model first. The algorithm covers the space with a square-grid of $a \times a$ sized cells. If a request arrives which is not covered by any of the existing clusters then the algorithm creates a new one which is the closed cell containing the request. Using this simple grid yields an algorithm which is 9-competitive for the best choice of the parameter a. Using shifts of $a/2$ (each row is shifted right by $a/2$ compared to the row below) in the grid gives an 8-competitive

algorithm for the best choice of a. Here we study an improved algorithm called $Shift(1/3)GRID_a$ with shifts of $a/3$. This modification leads to a better result: the algorithm is 7-competitive.

We propose the algorithm $Shift(1/3)FGRID_a$ for the solution of the flexible 2-dimensional model: if the cell from the grid which contains the new request point p does not contain a cluster then open a new one (in this case this cluster consists of a single point p) else extend the cluster contained in the cell to cover p.

In the competitive analysis it is obtained that the algorithm $Shift(1/3)GRID_a$ achieves its best competitive ratio for an interval of the parameter a. We studied how the average behavior of the algorithm depends on this parameter. Random request sequences of different length are generated with uniform distribution in a square of size 30×30. Both $Shift(1/3)GRID_a$ and $Shift(1/3)FGRID_a$ are investigated on the same sequences with the parameter a in the interval where the competitive ratio of $Shift(1/3)GRID_a$ is minimal and also with slightly smaller and larger parameter values. Based on the results of the experimental analysis the following conclusions are drawn. The best value of the parameter in the case of $Shift(1/3)GRID_a$ depends on the number of points which is related to their density. As the density is increasing it is better to use larger grid cells. In the case of $Shift(1/3)FGRID_a$ there is still a dependence between the density and the better parameter values but it is not so strong. The reason could be that $Shift(1/3)FGRID_a$ does not use the whole cells. This dependence on the density can be very useful if an a priori information about the points is known. As it is expected $Shift(1/3)FGRID_a$ has significantly smaller cost in the average case than $Shift(1/3)GRID_a$ not only its competitive ratio is less.

We collect our results published in [5] in Thesis 3.

Thesis 3

Theorem 8 *If* $\sqrt{1/2} \leq a \leq \sqrt{27/29}$*, then the competitive ratio of algorithm* $\textsc{Shift}(1/3)\text{GRID}_a$ *is* 7*.*

Theorem 9 *The competitive ratio of any online algorithm for the strict model is at least* 2.768*.*

Theorem 10 *Let* $x = (31 + \sqrt{10529})/59 \approx 2.2748$ *be the positive root of the following equation*

$$\frac{2 + 2x}{1 + x/9} = \frac{6 + 44x/9}{1 + x}.$$

If $a = \sqrt{x} \approx 1.508$*, then* SHIFT$(1/3)$FGRID$_a$ *is* C*-competitive where*

$$C = \frac{2+2x}{1+x/9} = \frac{6+44x/9}{1+x} \approx 5.228.$$

Theorem 11 *The competitive ratio of any online algorithm for the flexible model is at least* $C = (x+4)/(x+1) \approx 1.743$*, where* $x \approx 3.0351$ *is a root of the equation* $4x^3 + 4x^2 - 48x - 3 = 0$*.*

2.2.2 Extensions: d-dimensional space and general power instead of square of the side of a cluster

We also considered a general version of the variable sized clustering problem: d-dimensional Euclidean spaces and a more general cost function are studied. In our model the cost of a cluster is a unit setup cost plus the p-th power of the side of the grid cell. The strict version of the problem is examined where the exact location of the cluster is fixed when it is created. We analyze the algorithm $GRID_a$ for the general case and we prove that it is not constant competitive if $p < d$ and 3^d-competitive if $p \geq d$. In two dimensional Euclidean space we investigate the algorithm SHIFT$(1/3)$GRID$_a$ and we prove that it is 7-competitive for an adequate choice of the parameter a.

The summary of our results can be found in Thesis 4. Most of the results of Thesis 4 are published in [6].

Thesis 4

Theorem 12 *If* $p < d$ *then* $GRID_a$ *is not constant competitive if the cost is the* p*-th power of the side of the* d*-dimensional cube.*

Theorem 13 *If* $p \geq d$ *there exists a parameter* a *such that* $GRID_a$ *is* 3^d*-competitive. Moreover,* $GRID_a$ *has never smaller competitive ratio than* 3^d*.*

Theorem 14 *Algorithm* SHIFT$(1/3)$GRID$_a$ *is 7-competitive if*

$$\max\left\{\sqrt[p]{\frac{1}{2^p-2}}, \sqrt[p]{\frac{1}{7\cdot(4/3)^p-8}}\right\} \leq a \leq \sqrt[p]{\frac{27}{29}}.$$

$\sqrt[p]{\frac{1}{2^p-2}}$ *is greater for* $2 < p \leq x$ *and* $\sqrt[p]{\frac{1}{7\cdot(4/3)^p-8}}$ *is greater for* $p > x$ *where* $x \approx 4.0257$ *is the root of the equation:*

$$\frac{1}{2^p-2} = \frac{1}{7\cdot(4/3)^p-8}$$

.

3 Online facility location

In the facility location problem a metric space $\mathbf{M} = (M, d)$ is given, M is the set of points. To serve the requests facilities should be opened at the points of the set M. The input is a sequence $s_1, \ldots, s_n$ of requests, each request is a point of M. For the request sequence $I = s_1, \ldots, s_n$, we denote the prefix $s_1, \ldots, s_i$ by I_i. The goal is to find a set of facility locations which minimizes the sum of the facility costs and assignment costs (the cost of assigning a request point to a given facility). The offline version is a well-known problem in combinatorial optimization. In some applications (building a computer or sensor network, some clustering problems) the set of request points is not known in advance, demand points arrive one at a time and after their arrival the algorithm has to decide whether to open a new facility or to assign the demand to an existing facility without any information about the further demand points. This online version of the facility location problem has been studied first by Meyerson in [11]. In that paper it is not allowed to move a facility which is opened by the algorithm.

In [9] the problem is further investigated and an $O(\frac{\log n}{\log \log n})$-competitive deterministic algorithm is presented for uniform and nonuniform facility cost. Also, it is proved that no deterministic or randomized algorithm exists with smaller competitive ratio than $\Omega(\frac{\log n}{\log \log n})$.

In this work we introduce a model where the algorithm is allowed to move the facilities to other positions after the arrival of a demand point but the already opened facilities cannot be closed so increasing the number of facilities is an irrevocable decision that may later turn out to be a bad choice.

We restrict our attention to the special case of uniform facility cost, where the cost of opening a facility, denoted by f, is the same for all points. Our results are presented in [4]. If a solution SOL opens facilities at the points $a_1, \ldots, a_k$ the total cost of this solution is

$$c(SOL) = k \cdot f + \sum_{i=1}^{n} \min_{j=1,\ldots,k} d(s_i, a_j).$$

The goal is to find the solution which minimizes this cost. For any algorithm A let $C_A(I)$ denote the total cost, $S_A(I)$ the service cost and $F_A(I)$ the facility opening cost of the solution achieved on the input I.

We propose the algorithm OFW (Optfollow) for the solution of the online problem. The basic idea is to mimic the behavior of the optimal

offline algorithm. OFW uses the following rules to build the online solution after the arrival of the i-th point of the request sequence s_i:

Algorithm 4 Algorithm OFW

- Step 1. Determine an optimal offline solution for the input I_i.
- Step 2/a. If $F_{OFW}(I_{i-1}) \leq F_{OPT}(I_i)$ then open $(F_{OPT}(I_i) - F_{OFW}(I_{i-1}))/f$ new facilities, and move the $F_{OPT}(I_i)/f$ facilities into the optimal positions.
- Step 2/b. If $F_{OFW}(I_{i-1}) > F_{OPT}(I_i)$ then do not open new facilities, solve the offline $F_{OFW}(I_{i-1})/f$-median problem on I_i and move the facilities into the resulting positions.

Remark: Note that in the case of general metric spaces OFW solves an NP-hard problem in Step 1, also in Step 2/b. These NP-hard problems are well studied and several exact solution algorithms are developed for them (see [1], [8], [10] for details). On the other hand, if the metric space is the line then both problems can be solved in polynomial time.

The following polynomial version of OFW called $POFW$ can be defined using approximation algorithms as the solutions of the NP-hard problems.

Algorithm 5 Algorithm $POFW$

- Step 1. Use a polynomial time approximation offline algorithm $FAPPR$ on the input I_i in the facility location problem.
- Step 2/a. If $F_{POFW}(I_{i-1}) \leq F_{APPR}(I_i)$ then open $(F_{FAPPR}(I_i) - F_{POFW}(I_{i-1}))/f$ new facilities, and move the $F_{FAPPR}(I_i)/f$ facilities into the positions given by $FAPPR$.
- Step 2/b. If $F_{POFW}(I_{i-1}) > F_{FAPPR}(I_i)$ then do not open new facilities, use a polynomial time approximation offline algorithm $MAPPR$ for the $F_{POFW}(I_{i-1})/f$-median problem on I_i and move the facilities into the received positions.

Remark: Several approximation algorithms have been developed for the facility location and k-median problems, see [12] and [13] for surveys on these areas.

We summarize our results presented in [4] in Thesis 5.

Thesis 5

Theorem 15 *Algorithm OFW is* 2*-competitive.*

Theorem 16 *If FAPPR and MAPPR are c_1 and c_2 approximation algorithms respectively, then the algorithm POFW is $c_1(1+c_2)$-competitive.*

Theorem 17 *The algorithm OFW where we use the optimal solution which uses the smallest number of facilities is $\frac{3}{2}$-competitive on the line with the Euclidean distance.*

Theorem 18 *No online algorithm is C-competitive on the line for any $C < (\sqrt{13}+1)/4 \approx 1.15$.*

References

[1] Christofides, N., and Beasley, J.E. A tree search algorithm for the p-median problem. *European Journal of Operational Research*, **10(2)**, pp. 196–204, 1982.

[2] Csirik, J., Epstein, L., Imreh, Cs., and Levin, A. Online Clustering with Variable Sized Clusters. *Algorithmica*, **65(2)**, pp. 251–274, 2013.

[3] Divéki, G. Online Clustering on the Line with Square Cost Variable Sized Clusters. *Acta Cybernetica*, **21(1)**, pp. 75–88, 2013.

[4] Divéki, G. and Imreh, Cs. Online facility location with facility movements. *Central European Journal on Operations Research*, **19(2)**, pp. 191–200, 2011.

[5] Divéki, G. and Imreh, Cs. An Online 2-dimensional Clustering Problem with Variable Sized Clusters. *Optimization and Engineering*, **14**, pp. 575–593, 2013.

[6] Divéki, G. and Imreh, Cs. Grid based online algorithms for clustering problems. *CINTI 2014*, accepted for publication, 2014.

[7] Epstein, L., Levin, A., and van Stee, R. Online unit clustering: Variations on a theme. *Theoretical Computer Science*, **407(1-3)**, pp. 85–96, 2008.

[8] Erlenkotter, D. A Dual-Based Procedure for Uncapacitated Facility Location. *Operations Research* **26(6)**, pp. 992–1009, 1978.

[9] Fotakis, D. On the Competitive Ratio for Online Facility Location. *Algorithmica*, **50(1)**, pp. 1–57, 2008.

[10] Garfinkel, R.S., Neebe, A.W., and Rao, M.R. An Algorithm for the M-Median Plant Location Problem. *Transportation Science*, **8**, pp. 217–231, 1974.

[11] Meyerson, A. Online facility location. *In Proceedings of the 42nd Annual Symposium on Foundations of Computer Science (FOCS2001)*, pp. 426–431, 2001.

[12] Shmoys, D. Approximation algorithms for facility location problems. *Proceedings of 3rd International Workshop of Approximation Algorithms for Combinatorial Optimization, Springer-Verlag LNCS vol. 1913 (2000)*, pp. 27–33, 2000.

[13] Solis-Oba, R. Approximation Algorithms for the k-Median Problem. *In Efficient Approximation and Online Algorithms, LNCS 3484*, pp. 292–320, 2006.