A maximum profit coverage algorithm with application to small molecules cluster identification

Refael Hassin[1] and Einat Or

Department of Statistics and Operations Research, Tel Aviv University, Tel Aviv, 69978, Israel.

1 Introduction

1.1 Problem Definition

In this article we model, and analyze the CLUSTER IDENTIFICATION OF MOLECULES (CIM), which is a clustering problem in a finite metric space. CIM[2] has the following characteristics which separate it from other clustering models:
1. In most models outliers are a small portion of the data set, whereas in CIM they may be the vast majority of the objects. (see Figure 1)
2. The clusters identified by CIM are compact and their diameter is bounded.
3. There is a lower bound on the number of objects in a cluster.
4. Clusters may be very close to one another, as a result of the bound on the diameter. What may be considered as one cluster in other clustering models is considered as several clusters in CIM. (see Figure 2).
5. The number of clusters is not known a-priori to the clustering procedure.

In this paper we present CIM and model it as a MAXIMUM PROFIT COVERAGE PROBLEM (MPCP). The model is a measure to be optimized, rather then a heuristic.

Consider a finite set S in a metric space M with a distance function d. A ball with center t and radius r is the subset $B(t,r) = \{x \in M | d(t,x) \leq r\}$. We say that the ball *covers* the points of S that it contains. Given a set of balls $\mathcal{B}$ of radius r, a *coverage* $P = \{S'_1, \ldots, S'_l\}$ is a set of clusters such that each of them consists of points covered by a single ball of $\mathcal{B}$. Let $S'_P = \cup_{i=1}^{l} S'_i$, and define the *profit* of P as $\sum_{q \in S'_P} w_q - c|P|$, where c is the cost of a ball used by P, and w_q is a revenue obtained by covering $q \in S$. The MAXIMUM PROFIT COVERAGE PROBLEM (MPCP) is the problem of finding a coverage with maximum profit.

[1] hassin@post.tau .ac.il

[2] The problem originated in *COMPUGEN* LTD. in the field of "in silico" drug design.

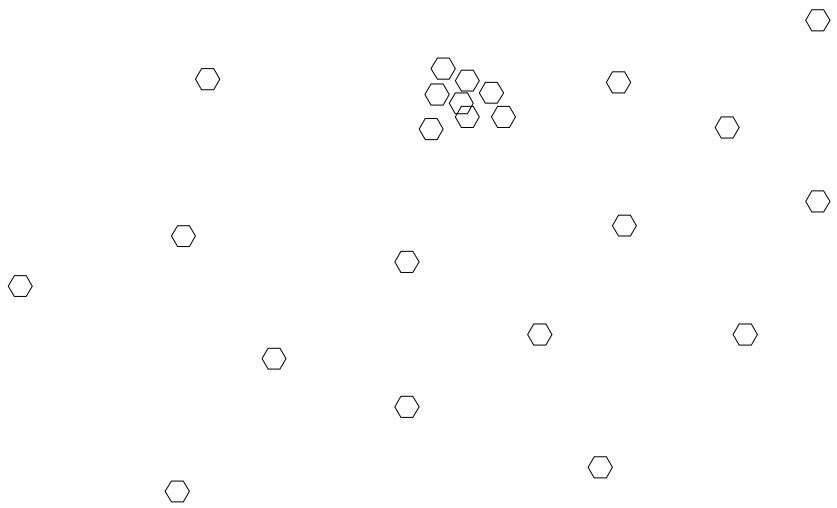

Figure 1: One cluster and some outliers

1.2 Related Work

The $MPCP$ is related to the BUDGETED MAXIMUM COVERAGE PROBLEM, which has a $\left(1-\frac{1}{e}\right)$-approximation using a greedy algorithm [14]. This approximation bound cannot be used to approximate $MPCP$.

1.2.1 Clustering

There is a large variety of clustering models, as reflected for example in the survey by Du and Paradalos [5], and they have numerous applications in many different areas. Even simple variants are known to be NP-hard and therefore research has been focused on approximation algorithms and heuristics. The heuristics used in practice are often based on empirical experience (e.g [16]). Most measures (i.e. objective functions to be optimized) are based on a given number of clusters and the relation of the between-clusters weight and the in-cluster weights.

Most heuristics are based on two main methods. One is the *hierarchical* method, which re-partitions the set until a stopping condition is met, or an aggregation presses which begins by considering each point as a cluster and then merging close clusters until a stopping condition is met [11, 15, 18, 23, 24]. The second is the k-means heuristic, where a *mean* of a cluster is the average of the clusters points. This non hierarchical method initially takes k

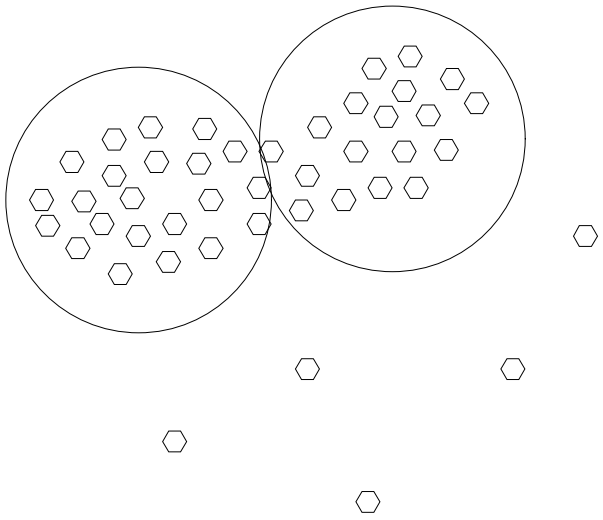

Figure 2: Due to the bound on the diameter, two clusters and outliers

point of the set which are mutually farthest apart, at this point each cluster has one point. Next, it examines each point in the set and assigns it to the nearest cluster. This continues until all the points are grouped into k clusters [11]. Both heuristics take the number of clusters as input, and do not return compact clusters of bounded diameter.

An important class of clustering problems is probabilistic clustering. It arises when the data consists of a set of points generated by an unknown *mixture* of distributions, and the problem is to estimate the parameters of each of the distributions creating the mixture, and its weight in the mixture. This task is commonly done by likelihood maximization (for example [19]). The mixture models solved by likelihood maximization can also be applied to cases where the number of distributions is unknown. There are methods for estimating the number of distributions [22].

1.2.2 Clustering with outliers

In statistics, *outliers* are defined as data objects which originated from a different probabilistic mechanism [2]. When clustering of data is considered, the intuitive definition of outliers becomes "points which do not belong to any of the clusters". A more specific definition is derived from the clustering target or the clustering process [4, 6, 7, 10, 16, 27, 28]. In the case of the CIM, the outliers are objects in a neighborhood not dense enough, according

to the given definition of density.

The natural problem of clustering a data set containing outliers, when the number of clusters is not predetermined, is usually solved by heuristics. Models for *k-median* and *k-center* with outliers are introduced in [3]. For *k-median* outliers were considered by demanding payment on unclustered points and for *k-center* the number of points which are considered as outliers was added to the model. Heuristics for clustering data containing outliers were introduced, among others, in [4, 6, 7, 10, 16, 27, 28]. In hierarchical methods clusters that "grow" slowly are considered as outliers, whereas in the k-means method points that are far from all means are considered as outliers.

All algorithms presented in these articles are not suitable for CIM. The clusters found by the algorithms are not bounded in diameter, and/or have no lower bound on their density. Since the nature of clusters found is different from the one defined in CIM, the outliers definition is also different. A clustering method called *CLARANS* which is based on random search on a graph is presented in [16, 7]. Their objective is to find the k best mean points. They also propose a method for determining the value of k. *CLARANS* detects outliers as in the classical k-means method mentioned above. A clustering method called *BRITCH* is presented in [28]. It uses the hierarchical clustering algorithm from [18]. *BRITCH* works well for compact well separated clusters, it does not work well when clusters have different sizes or are connected, even by outliers.

For clustering of arbitrarily shaped collections of points, a density based algorithm called *DBSCAN* is proposed in [6]. The clustering is done by a definition of density. *DBSCAN* requires the user to specify two parameters that are used to define the minimum density for clustering- the radius Esp of a neighborhood of a point and the minimum number of points $MinPts$ in the neighborhood. The algorithm begins with an arbitrary point and if its neighborhood satisfies the density requirements it is inserted into the cluster with its neighborhood. *DBSCAN* defines the points in non-dense neighborhoods of the data set as outliers. The intuition of outliers is similar to that of the CIM, but since there is no diameter restriction the clustering outcome is different. *DBSCAN* is very sensitive to the parameters Esp and $MinPts$, which in tern are difficult to determine. It also tends to join clusters that are not well separated.

A two phase clustering process for outliers detection is presented in [13]. It is a modification of the k-means algorithm that builds a minimum spanning tree between the clusters centers. The algorithm then removes the longest edges of this spanning tree, and the clusters whose centers are dis-

connected from the tree, are considered as outliers.

A clustering method that combines k-means and hierarchical method is presented in [10]. The algorithm uses an input parameter α which determines "how much" hierarchical or k-means the algorithm is. This parameter, in turn, is hard to define and has great effect on the outcome of the clustering procedure. The algorithm is shown to work well when clusters are of different shapes and close to one another. It prunes outliers by removing the clusters which grow slowly in the aggregative part of the algorithm.

Probabilistic clustering of data containing outliers is presented in [26]. Assuming the data follows a given distribution while the noise follows a different distribution, this approach applies the Bayes rule of classification to extract the clusters one after the other. It computes the first cluster, then removes the points of the cluster from the set and computes the next cluster. When clusters are no longer found, the remaining of the data is considered as noise. This approach does not perform well when data from several distributions overlap in space (creating close clusters). In that case the removal of points from the set that were generated by one distribution creates errors in estimating the parameters of distributions close to it.

1.3 Our Contribution

We present two models for CIM, one as a PARAMETER ESTIMATION VIA LIKELIHOOD MAXIMIZATION and the other as MPCP. We introduce a polynomial time approximation scheme (PTAS) to MPCP in Euclidean space using the shifting strategy [12, 9]. We present two practical heuristics for MPCP, one greedy and the other random, which introduce good results in numerical studies of CIM. We also study the problem of MPCP when the number of clusters is bounded using dynamic programming and the shifting strategy [12, 9].

This paper is organized as follows: In Section 2 we introduce CIM and model it as PARAMETER ESTIMATION VIA LIKELIHOOD MAXIMIZATION. In Section 3 we model the problem as MPCP. In Section 4 we introduce a random heuristic and a greedy heuristic for MPCP, and in Section 5 we introduce numerical results. In Section 6 we give some theoretical results, We also analyze a greedy algorithm. In Section 7 we introduce the variant of MPCP in which the number of balls is bounded.

2 Cluster identification of molecules

Finding a drug to an illness is a problem of a lock and a key. The lock is a protein (or, more precise, its active site), that should be inhibited or exhibited in the body. The key is a small molecule that binds to the protein and inhibits or exhibits its action in the body. There is no doubt that the key should have a structure that fits the lock, but the biochemical system in which this lock and key function is dynamic, and hence the Structure Activity Relation (SAR), of the small molecule and the given protein, was not yet unfolded.

One of the approaches to this problem is to investigate the relation between a secondary structure of the small molecule to the biochemical activity, rather than the regular structure model of atoms and bonds. The secondary structure views the small molecule as a set of chemical attributes such as base, acid, hydrophobic, hydrophilic, hydrogen bonds etc. Given a set of small molecules that bind to the same protein, we wish to check whether there is a similarity in their secondary structure. Since the protein binding site is big, different small molecules may bind in different parts of it, and several secondary structures may explain the binding. Still it is natural to assume that a secondary structure that appeared in many of the small molecules that bind to the protein, characterizes the protein, and hence explains the binding. The CLUSTER IDENTIFICATION OF MOLECULES is the problem of identifying recurring secondary structures (clusters of secondary structures) in a set of small molecules that bind to the same protein.

Formally, we consider the secondary structure as a set of colored points, called *nodes*, in $\mathbb{R}^3$. The coordinates represent the location of a chemical functionality and the color represents the nature of the functionality. We will use this representation throughout this section and refer to it as a *molecule structure*.

Denote by V_t the set of nodes of a molecule structure t, and let $n = |V_t|$. The $3-D$ structure of t can be expressed as a vector of $\binom{n}{2}$ distances $d^t(i,j)$ between the pairs $v_i, v_j \in V_t$. The distance measuring process has a normally distributed error with a constant variance.

Thus, t can be viewed as a $\binom{n}{2}$-dimensional multi-normal random variable. The variance of the distances is constant, and caused due to measurement errors.

The distance between two molecule structures with the same number of nodes and the same multi-set of colors of the nodes, is defined as follows. Let P denote the set of mappings $p : V_t \to V_s$ such that the color of the source and the objective is the same. The distance between the molecule

structures t and s is

$$D(s,t) = min_{p\in P}\sqrt{\sum_{i,j\in V_t}[d^t(i,j) - d^s(p(i),p(j))]^2}.$$

If the number of nodes is different or the multi-set of node colors is different, then the distance is infinity.
The set of all molecule structures with a given number of nodes, a given multi-set of node colors and a distance function D is a metric space (see a proof in Appendix A). In the remaining of this paper we denote such a metric space by M. An efficient method for finding the distance between two given molecule structures is given in Appendix B.

Given a set of molecule structures, the CLUSTER IDENTIFICATION OF MOLECULES can be viewed as the problem of estimating the parameters of a mixture of distributions, since each molecule is represented by the vector of its distances, which is a multi-normal random variable. We will now build the likelihood function L of this probabilistic clustering.

Let t denote a molecule structure in M. t has a positive probability to be generated by any of k multi-normal distributions considered by the mixture model. Consider the possibility that molecule structure t has originated from the j-th distribution. Let V_j denote the set of nodes of the molecule structure which is the mean of the j-th distribution. Let $L_{(j,p)}(t)$ denote the likelihood that t has originated from the j-th distribution under the mapping p of their nodes.

$$L_{(j,p)}(t) = \frac{1}{\sqrt{2\pi|\Sigma_j|}}\exp^{-\frac{1}{2}([x(p)-m_j])^T\Sigma_j^{-1}[x(p)-m_j]},$$

where (m_j, Σ_j) are the mean vector and covariance matrix of the j-th distribution, and $x(p)$ is x when adapted to m_j under the permutation p. The normality is a result of the error in the measure of the distance, as mentioned above. Denote by $\beta_{(j,t)}(p)$ the probability that p was the mapping by which t originated from the j-th distribution. Clearly, $\sum_{p\in P}\beta_{(j,t)}(p) = 1$. The likelihood of the molecule structure t given that it is obtained from the j-th distribution is:

$$L_j(t) = \sum_{p\in P}\beta_{(j,t)}(p)L_{(j,p)}(t).$$

No information exists on the distribution $\beta_{(j,t)}$, but it could be estimated by solving $\max_{p\in P} L_{(j,p)}(t)$. Even if we assume $\beta_{(j,t)}(p)$ is known, since

$|P| = O(|V_t|!)$ this step is exponential in the dimension of M, this likelihood function is expensive to calculate for one point, let alone to maximize.

3 Cluster identification of molecules as a maximum profit coverage problem

Solving the likelihood maximization associated with the cluster identification of molecules in a general metric space is computationally difficult even for small sized instances, as demonstrated in Section 2. We now show how the problem can be approximated via the MAXIMUM PROFIT COVERAGE PROBLEM.

A general finite metric space can be described by a graph $G = (V, E)$, and a distance function D on its edges. In such a case a ball can be defined only as centered at a vertex. In the CIM the balls are defined by the subset of molecule structures which they cover, i.e for each subset that can be covered by a ball of radius r we define one ball in the input set of balls $\mathcal{B}$. We compute the set of balls exhaustively, by checking for each subset whether it is coverable by a ball of radius r. This step is theoretically exponential i the size of S, but in practice the number of balls is very small due to the distribution of points as presented in Section 2. In simulations the number of balls defined was $O(n^2)$, and the process of defining them was efficient.

Consider a set S of molecule structures. The CLUSTER IDENTIFICATION OF MOLECULES can be viewed as the search for such dense balls, using the average of the points covered by each ball as estimate for the mean value of the distribution that generated the points of S in the ball[3]. We use a ball since the variance is constant and equal in all dimensions, i.e. for all the distances in the molecule structure. The intuition behind this approach relies on the fact that a point close to t is likely to be generated by t, in terms of the value of the likelihood function.[4]

[3]This point is a molecule structure which is not in S, and can be reconstructed to a set of nodes in $\mathbb{R}^3$ from the vector of its distances, with an error negligible in relation to the variance of the measuring process.

[4]This view of density ignores some aspects of the normal behavior, such as greater density in proximity to the expected value. A change in the objective function may better integrate this characteristic of the problem into the model. If a value of a ball will include not only the number of points it contains but also an indication on their variance in the ball, another aspect of the normality of the data origin will be integrated into the solution, but other difficulties will airs. We will show that even the simple definition of density used in MPCP gives very good results.

A ball $B(t, r)$, covers the molecule structures which are contained in it. There is a revenue of 1 from covering a molecule structure $t \in S$, and every ball used by the solution costs c. The cost c represents a lower bound on the number of molecule structures in the cluster, whereas $2r$ represents an upper bound on the diameter of the cluster. Let S_F denote the molecule structures covered by a coverage F. The profit of a coverage F is $|S_F| - c|F|$. The maximum profit coverage of S can be used to estimate the large clusters of S. Since many covers may have the same profit, the center of the cluster will be defined by the average of the molecule structures it covers.

4 Heuristics

4.1 Heuristics for MPCP

In this section we describe two heuristics for computing a coverage with high profit in a general metric space. The first is a greedy heuristics, and the second is based on a random selection of subsets. We denote the set of possible balls defined in Section 3 by $\mathcal{B}$. The number of such balls is $O(|S|^d)$, where $d = \binom{n}{2}$, where n is the number of nodes in a molecule structure of S.

Consider the following "greedy" algorithm denoted as **GR**:
Let $B_j \subset \mathcal{B}$ be the set of balls already chosen before the j-th iteration, $S'_j \subseteq S$ the set of points that are not covered by B_j, and n_B^j the number of points of S'_j covered by $B \in \mathcal{B}$. Recall that c denotes the cost of a ball. Let $R_B^j = n_B^j - c$ denote the profit from a ball B. A *profitable ball* is a ball that covers at least $c + 1$ points of S'_j, i.e. $R_B^j > 0$. Let $B_j^* = \arg\max_{B \in \mathcal{B}}\{R_B^j\}$ (break ties arbitrarily). **GR** is performed as follow:
Let $\mathcal{B}_1 = \emptyset$. As long as there is a profitable ball, let $\mathcal{B}_{j+1} = B_j \cup \{B_j^*\}$.

The greedy algorithm is performed in $O(|S|^{d+1})$ time. The number of iterations is bounded by $\frac{|S|}{c+1}$, since at least $c + 1$ new points are covered at each iteration since, and the number of operations at each iteration is $|\mathcal{B}| = O(|S|^d)$.

The randomized heuristic **RA** repeatedly generates random solutions to the problem, and eventually chooses the one with the maximal profit. A solution is generated by randomly choosing a profitable ball from the list of profitable balls, then updating the list and choosing again, until there is no profitable ball and the list is empty. The generated solution is compared with the previous best solution and saved if it gives a higher profit. The randomized algorithm is performed in $O(|S|^d)$ iterations, since the number

of balls dominates the number of iterations[5].

4.2 A heuristic for CIM in the Euclidean space

If each molecule in S has nodes of distinct colors, then the metric space M is Euclidean. The distribution $\beta_{(k,t)}$ becomes deterministic and the problem becomes the known PARAMETER ESTIMATION FOR GAUSSIAN MIXTURE (PEGM) [17]. It requires to estimate the parameters of a set of multi-normal distributions $\{(m_k, \sum_k)\}_{k=1,\ldots,K}$, and their mixture proportions $\alpha_j \geq 0$, $\sum_{k=1}^{K} \alpha_k = 1$, when a set of points generated from these distributions is given. Note that the number of distributions is assumed to be predetermined. There are methods for estimating the number of distributions. The parameter estimation is usually done by likelihood maximization. In this case the likelihood function can be maximized numerically. We will elaborate on this case since we will use this model for comparison to the results obtained by modeling CIM as MPCP and using GR and RA. We use algorithm **MEM** presented in Figure 3 below. **MEM** is based on the common method of likelihood maximization via Expectation Maximization, i.e. the **EM** algorithm [17, 21, 25]. See a description of the algorithm in Appendix D. The input to the **EM** algorithm includes the number of distributions k and will return a maximum likelihood estimator of the parameters of k distributions that best explain S. In order to find the optimal k we use the rule given by [22] to estimate the number of generating distributions, i.e k that maximizes $\max \log\_\text{like}(k) + k \log(N)$, where N is the number of samples, and $\max \log\_\text{like}(k)$ is the maximal value of the likelihood function for k distributions. The evaluation of the models' size k in [22] is based on studying the asymptotic behavior of Bayes estimators under a special class of priors. These priors are not absolutely continuous, since they put positive probability on the subspace that corresponds to the competing model. Since we are only interested in the distributions which generated a large number of points, i.e. the dense sets of S, we define a dense subset of S as one generated from a distribution with a mixture parameter α greater then $\frac{c}{|S|}$.

[5] In practice the execution time of both the greedy and the random heuristics is much lower since $\mathcal{B}$ is smaller.

MEM

input

1. A set $x = \{x^{(1)}, \ldots, x^{(N)}\}$ *of molecule structures in* $[0, 10]^3$ *presented by the vector of distances between their nodes.*

2. An integer c. [A lower bound on the size of a cluster]

3. num_iterations.

output

The means of the clusters, and their size.

begin

for $i = 1 : N$

$max\_log\_like = -\infty$

$best\_log\_like = -\infty$

for $j = 1 : num\_iterations$

rand := random initial solution[a] $\Theta = \{\alpha_1, \ldots, \alpha_K, m_1, \ldots, m_K, \Sigma_1, \ldots, \Sigma_K\}$ *created from i molecule structures randomly chosen in* $[0, 10]^3$

$(log\_like, \Theta) := \mathbf{EM}(x, i, rand)$

if $log\_like > \max\_\log\_like$

then

$\max\_\log\_like = \log\_like$, $\Theta\_max = \Theta$

end if

end for

if $best\_\log\_like < \max\_\log\_like + i \log(N)$

then

$best\_log\_like := max\_log\_like$,

$best\_\Theta := \Theta\_max$, $num\_distributions := i$.

end if

end for

for $j = 1 : num\_distributions$

if $\alpha_j N > c$

then

insert the pair $(\alpha_j N, m_j)$ *to the output set.*

end if

end for

return *(output set)*

end MEM

[a]Several methods for generating initial solutions were used. One is random, another is choosing independently k points of x, and another is choosing without replacements points of x, such that the distance between the chosen points is at most $2r$.

Figure 3: Algorithm **MEM**

5 Numerical comparison of the heuristics

We described algorithms **GR** and **RA** for MPCP in a general metric space. However, in the Euclidean space CIM becomes the well-known PEGM where the number of distributions is unknown. Since PEGM has a well-known method of solution, presented in **MEM**, we conduct our numerical analysis in the Euclidean space, where we can compare **GR** and **RA** to **MEM**. Our numerical analysis is hence a comparison of the three algorithms on simulated input for the problem in the Euclidean space.

The algorithms were applied with the following parameters:

1. **MEM** was applied with $num\_iterations = 10000$ and $c = 3, 5, \ldots, 23$.
2. **GR** was applied with $c = 3, 5, \ldots, 23$. The algorithm returns the mean of the molecule structures covered by each ball in the solution as the estimate of the expectation, and the number of molecule structures covered by the ball. We choose the mean since it is a natural estimate of the expectation.
3. **RA** was applied with $c = 3, 5, 7, \ldots, 23$. For each value of c **RA** was run 10000 times and the best solution was chosen. The algorithm returns the mean of the molecule structures covered by each ball in the solution as the estimate of the expectation, and the number of molecule structures covered by the ball.

Remark 5.1 *1. Although c is part of the input, we ran the algorithms on all possible values of c in order to check the sensitivity of the algorithms to the value of c.*

2. The running time of all algorithms was a couple of minutes.

The data was simulated in the following way:

1. Randomly choose $|V| = 4$ points in the cube $[0, 10]^3 \subset I\!R^3$. The points represent the nodes of a molecule structure.[6]
2. Compute the vector $m = (m_1, \ldots, m_6)$ of $\binom{|V|}{2} = \binom{4}{2} = 6$ distances between the $|V| = 4$ points.

[6] We applied this simulation to molecule structures of all sizes between $|V| = 4$ and $|V| = 10$ nodes, and the results were similar. We therefore present the results of molecule structures of size 4 as representative results.

3. For every $i = 1, \ldots, 6$, choose a distance from a normal distribution with mean m_i and a known constant variance.

Steps $1 - 2$ create a molecule structure and represent it as a vector of the distances between its nodes. Step 3 generates a sample from a distribution with mean value created by Steps $1 - 2$.

We present the results of the following cases:

- In a simulation of type a we simulate a CIM instance by generating 20 samples from each of 5 mean molecule structures. A total of 100 molecule structures. The set A of simulations includes 100 CIM instances of type a. In simulations of type a there is no noise, there are 5 clusters that we wish to identify.

- In a simulation of type b we simulate a CIM instance by generating 20 samples from each of 5 mean molecule structures. In addition we generated noise by generating 27 molecule structures from which we generated one sample, 20 molecule structures from which we generated two samples, and 6 molecule structures from which we generated three samples. Type b simulations include 5 clusters that we wish to identify and another 85 points that are considered as noise. The set B of simulations includes 100 cases of type b.

For each of the three algorithms, in each set A, B we calculated the following measures:
$ABN :=$ the average number of balls (distributions) used in the solution.
$MSE :=$ the average distance between a center of a cluster defined by the algorithm and the closest mean value. The closest mean value, is the closest vector m used in Step 2 of the data simulation process.
$GMSE :=$ the average distance between a mean value (only of the 5 clusters we wish to identify, i.e. with number of samples $> c$) and the closest center of a cluster defined by the algorithm.

MSE associates for each cluster representative, the nearest mean value, so some mean values may not be associated with any cluster, while $GMSE$ associates each mean value with the nearest cluster representative, so if a representative is not the closest representative to any mean value, it would not be considered.

The results are presented in Table (1). In general **RA** and **MEM** give accurate estimations of the mean values, while **GR** has slightly less accurate results.

The types of errors in estimation presented by the algorithms are the following:

1. Joining close clusters, i.e. returning only one expectation instead of two close expectations. This error is common with **MEM** [21] and is expected from **GR**. It was less common with **RA**. Such an error is reflected by the number of clusters estimated, and by the value of **GMSE** in Table 1 set A.

2. Cutting a cluster in two. This error is mainly characteristic to **RA**. If c is small enough then one cluster may be split into several profitable balls, so that one cluster is estimated by several close expectations.

3. Omitting a cluster. If c is large then it may happen that there is no ball of radius r containing $c+1$ points, although there is a mean value which generated more then $c+1$ points. For example for $c > 15$ in **GR** and **RA**, the $GMSE$ is much greater than the MSE because the clusters found are fairly accurate, but there are clusters omitted from the solution. This is less common with **MEM**.

4. **MEM** is sensitive to outliers, since the likelihood maximization includes the explaining of all data points, and the size of the model is not known. This explains the large MSE in Table 1 set B, that diminishes as c grows since less small clusters are included.

5. Since **MEM** is sensitive to outliers it clusters many of these points for small values of c, while **GR** and **RA** hardly cluster outliers.

6 Theoretical analysis of the maximum profit coverage problem

In this section we present a PTAS for MPCP in the Euclidean space and analyze algorithms **GR** and **RA**.

6.1 PTAS using the shifting method

We adapt the shifting method of Hochbaum and Maass [12] for MPCP. We introduce the method in the plane, which could be generalized to a fixed dimension d as in [12].

Let the set S of n given points in the plane be enclosed in a region I.

		Set A			**Set** B	
c	ABN	MSE	GMSE	ABN	MSE	GMSE
		MEM			**MEM**	
3	5.313	0.981	1.193	10.029	3.534	1.736
5	5.059	0.983	1.195	8.926	3.400	1.736
7	4.876	0.950	1.164	7.925	3.192	1.721
9	4.692	0.964	1.189	7.000	2.837	1.704
11	4.673	0.966	1.227	6.000	2.461	1.708
13	4.681	0.960	1.216	5.442	2.086	1.729
15	4.562	0.986	1.365	5.271	1.987	1.746
17	4.428	1.011	1.507	5.087	1.823	1.717
19	3.000	1.534	1.598	4.938	1.725	1.708
21	1.114	6.322	30.736	3.454	2.492	12.724
23	1.107	6.514	34.363	2.614	2.722	19.243
		GR			**GR**	
3	5.360	1.453	1.405	5.680	1.441	1.382
5	5.223	1.448	1.426	5.286	1.416	1.398
7	5.1322	1.446	1.428	5.202	1.411	1.402
9	5.031	1.442	1.478	5.052	1.400	1.486
11	4.950	1.434	2.088	4.940	1.392	2.349
13	4.859	1.431	2.502	4.829	1.388	3.117
15	4.708	1.425	3.775	4.748	1.376	3.790
17	4.507	1.407	5.577	4.627	1.370	5.068
19	3.755	1.411	10.918	4.938	1.725	1.708
21	2.251	1.176	37.567	1.622	1.523	48.630
23	1.750	1.097	46.450	2.622	2.255	106.512
		RA			**RA**	
3	6.070	0.857	0.813	6.540	0.852	0.794
5	5.340	0.855	0.844	5.465	0.836	0.818
7	5.153	0.850	0.846	5.224	0.832	0.822
9	5.041	0.847	0.879	5.072	0.817	0.896
11	4.950	0.831	1.483	4.950	0.810	1.775
13	4.864	0.833	1.958	4.837	0.798	2.527
15	4.708	0.822	3.187	4.748	0.791	3.195
17	4.507	0.801	4.956	4.627	0.785	4.474
19	3.755	0.788	10.332	3.736	0.752	10.959
21	2.251	1.064	38.414	1.622	0.872	46.169
23	1.750	1.132	47.785	2.622	1.259	101.040

Table 1: results for the set A and B

The goal is to cover a subset of S with disks of diameter D such that the profit is maximized. Let the shifting parameter be l. In the first phase the area I is subdivided into vertical strips of width D, where each strip is left closed and right open. Groups of l consecutive strips, resulting in strips of width lD each, are considered. For any fixed subdivision of I into strips of width D, there are l different ways of partitioning I into strips of width lD. These partitions can be ordered such that each can be derived from the previous one by shifting it to the right over distance D. Repeating the shift l times we return to the initial partition. We denote the l distinct shift partitions by $P_1, P_2, \ldots, P_l$. Let A be any algorithm that delivers a solution of width lD in any strip of width lD (or less). For a given partition P_i let $A(P_i)$ be the algorithm that applies algorithm A to each strip in the partition P_i and outputs the union of all disks used. This process is repeated for each partition P_i, $i = 1, 2, \ldots, l$. The shifting algorithm A_P, defined for a given local algorithm A, delivers the set of disks of maximum profit among the l sets delivered by $A(P_1), A(P_2), \ldots, A(P_l)$. We begin by introducing two properties of the problem.

Property 1: Denote by $opt(S)$ the value of the optimal solution for the set S. If $S' \subset S$, $opt(S') \leq opt(S)$.
Property 2: Let OPT be the set of unit balls in the optimal solution for the set S, and $OPT' = OPT \setminus \{B\}$ where $B \in OPT$. Let $S \setminus S'$ be the subset of points which B covers uniquely then OPT' is optimal for S'.
Proof: Assume there exists a solution for S', that is better than OPT'. Denote it by BET. Then the solution $BET \cup \{B\}$ is better than OPT for S, contradicting the optimality of OPT. ∎

Let the performance ratio of an algorithm A be denoted by r_A.

Lemma 6.1 *$r_{S_A} \leq r_A(1 - \frac{1}{l})$, where A is a local algorithm and l is the shifting parameter.*

Proof: Denote by $a(P_i)$ the value of the output of algorithm $A(P_i)$. By the definition of r_A we have:

$$a(P_i) \geq r_A \sum_j opt(P_{ij}), \tag{1}$$

where j runs over all strips in P_i and $opt(P_{ij})$ is the value of the maximum profit coverage of the points in strip j of partition P_i. This follows since if a unit ball is considered twice in adjacent strips j and $j+1$, we pay for it twice, once in $opt(P_{ij})$ and once in $opt(P_{i,j+1})$. Since $a(P_i)$ is the union, we only pay for it once in $a(P_i)$.

Let $opt(S)$ be the value of the optimal solution for S and $OPT(S)$ be the set of unit balls used in it. Let DIS^i be the set of unit balls in the optimal solution covering points in two adjacent strips in P_i and dis^i is the profit (revenue minus cost) from it. (If a point is covered by two balls we arbitrarily assign it to one of the balls, and consider it in its profit.)

Let OPT^i denote the set of unit balls in the optimal solution for the subset S^i of S that is not covered by DIS^i, and opt^i the profit from it. Clearly:

$$opt^i = opt(S) - dis^i. \tag{2}$$

By Property 2:

$$opt(S^i) = opt^i, \tag{3}$$

and by Property 1, for S,

$$\sum_{j \in P_i} opt(P_{ij}) \geq opt(S') = opt^i. \tag{4}$$

By (2) and (4)

$$\sum_{j \in P_i} opt(P_{ij}) \geq opt(S) - dis^i. \tag{5}$$

Since $DIS^1, \ldots, DIS^l$ are disjoint and add up to $OPT(S)$, summing (5) over $i = 1, \ldots, l$ gives:

$$\sum_{i=1}^{l} \sum_{j \in P_i} opt(P_{ij}) \geq \sum_{i=1}^{l} \left[opt(S) - dis^i \right] \geq (l-1) opt \tag{6}$$

and we get:

$$\max_i \left(\sum_{j \in P_i} opt(P_{ij}) \right) \geq \frac{1}{l} \sum_{i=1}^{l} \sum_{j \in P_i} opt(P_{ij}) \geq \left(1 - \frac{1}{l} \right) opt(S). \tag{7}$$

Finally from (1) and (7),

$$\max_i (a(P_i)) \geq r_A (1 - \frac{1}{l}) opt(S).$$

■

Given a coverage of S, an equivalent coverage using unit balls could be defined by shifting the unit balls such that a maximum number of the points they cover will be on their sphere, and the set of points covered by each unit ball will not change. Denote the set of balls with a maximum number of the points they cover will be on their sphere by B_S. In the following we consider B_S as the set of possible unit balls in any coverage of S.

Remark 6.2 *If the set of balls $\mathcal{B}$ is a given set, as in the general definition of $MPCP$, then it can be shown using the bound on the volume of the cube, that the following local algorithm still holds.*

THE MAXIMUM PROFIT COVERAGE PROBLEM could be solved in a small cube in R^d. Let A denote a cube of volume $(2l)^d$ and $S' \subseteq S$ denote the set of points in it. The set of unit balls $B_{S'}$ that could be defined by S' is bounded by $2(|S'|_d)$ as demonstrated in Section 3. Since A could be covered by $\sqrt{2}l^d$ balls, checking all subsets of at most $\sqrt{2}l^d$ balls yields the optimal solution. The complexity is $O(|S'|^{d\sqrt{2}l^d})$ time.

6.2 The greedy algorithm GR

Consider Algorithm **GR** presented in Section 4.1. Let OPT denote the set of balls in an optimal solution and *opt* denote its value, and let G denote the set of balls returned by **GR**, sorted in the order by which they were chosen, and g denotes its value. Let α denote the minimum number of unit balls that cover a ball of radius 3.

Theorem 6.3 $g \leq \frac{1}{\alpha} opt$

Proof: Define the *primary profit* from a ball B is $p_B = \sum_{q \in B} w_q - c$, while the *effective profit* from a ball considers only the points that are covered uniquely by that ball. Points that are covered by more then one ball are assigned to an arbitrary ball. The proof is by induction on $|G|$. Assume $|G| = 1$, and let $B_1 = B(s, 1) \in G$ denote the first and only greedy choice. Denote the set of unit balls contained in $B(s, 3)$ by H, clearly $B_1 \in H$. Since $|G| = 1$, there were no balls with positive primary profit outside of H, and after the choice of B_1 all effective profit of the balls in $H \setminus \{B_1\}$ is non-negative. The choice of B_1 cost us the loss of the effective profit of at most α balls with effective profit p_{B_1}, this is because there is no ball in H that has a primary profit greater then p_{B_1}. Hence the claim holds for $|G| = 1$.

Assume $\frac{g}{opt} \leq \frac{1}{\alpha}$ if $|G| = n - 1$. Let H_i denote the set H of the i-th ball in G. The proof for $|G| = n$ is straightforward if H_n is disjoint of all H_i, $1 \leq i \leq n - 1$. If this is not the case, then clearly the bound is only tighter. ■

We are left with bounding α. This problem remains open. We present an example in the plane, we believe to be tight. Consider the case where $w_i = w \ \forall i$ and $c = 8w$. (c is the cost of a cycle and w is the revenue from covering a point). We now introduce the location of the set S on the plane,

see figure 4. Consider a unit cycle C centered at the origin. There are 9 points uniformly located on the boundary of C. For every one of the 9 points, there is a cycle tangential to C at that point. On each tangential cycle there are 8 points located on its boundary, on the other end of the diameter from the tangential point. The points are dense around that point. It is optimal to choose the set of tangential balls getting a profit of $9w$. The greedy might choose C, getting a profit of w. The approximation ratio is $1/9$.

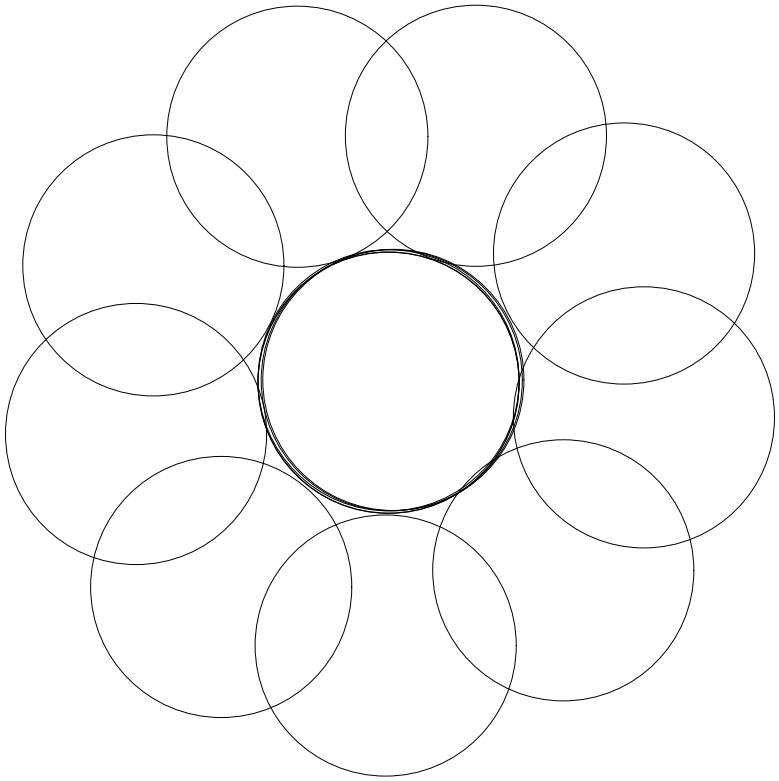

Figure 4: The profitable balls covering set S in the plane

7 Maximum profit with L balls

Consider a set S of points in a metric space with a distance function d. A ball with center c and radius r is the subset $B(c, r) = \{x \in S | d(c, x) \leq r\}$. The ball *covers* the points that it contains. An *L-coverage* $P = \{S'_1, \ldots, S'_L\}$

is a set of clusters such that each of them consists of points covered by a single ball with radius r. Let $S'_P = \cup_{i=1}^{L} S'_i$, and define the *profit* of P as $\sum_{q \in S'_P} w_q - cL$, where c is the cost per each ball used by P, and w_q is a revenue obtained when covering $q \in S$. The MAXIMUM PROFIT L-COVERAGE PROBLEM (MPLCP) is the problem of finding an L-coverage with maximum profit using at most L balls.

This problem is NP-hard, since by solving it in polynomial time for each $L = 1, 2, \ldots, |S|$, we obtain a polynomial time algorithm for the MAXIMUM PROFIT COVERAGE PROBLEM. The solution we offer is based on the shifting strategy as defined in Section 6.

Let $g_j^i(a)$ be the optimal solution to this problem for the j-th strip in partition P_i, using at most a unit balls. Let $f_j^i(L)$ denote the value of the cover of the first j strips of partition P_i. We use dynamic programming:

$$f_j^i(b) = \max_a \{g_j^i(a) + f_{j-1}^i(b-a)\}, \ a \leq L. \tag{8}$$

The value of the approximation algorithm would be $f(L) = \max_i \{f_l^i(L)\}$.

Lemma 7.1 *$r_{S_A} \leq r_A(1 - \frac{1}{l})$, where A is a local algorithm and l is the shifting parameter.*

Proof: By the definition of r_A (Section 6 and $f_l^i(L)$ we have $f_l^i(L) \geq r_A \sum_{j \in P_i} g_j^i(a_j)$, where j runs over all strips in P_i, and a_j is the number of unit balls used to cover strip j in $f_l^i(L)$. This follows since the same unit ball may be chosen in two adjacent strips.

Let $opt(S)$ be the value of the optimal solution of MPLCP on S. Let DIS^i, $i = 1, 2, \ldots, l$, be the set of disks in this solution covering points in two adjacent strips in P_i, and dis^i is the profit from it. Let OPT^i denote the set of unit balls in the optimal solution for the subset S^i of S that is not covered by DIS^i and opt^i the profit from it. Clearly,

$$opt^i = opt(S) - dis^i, \tag{9}$$

and

$$opt(S^i) \geq opt^i. \tag{10}$$

Since on the right L unit balls cover the points of S' with an optimal partition of the unit balls to the strips, and on the left at most L unit balls are used to cover S^i, and their partition to the strips may not be optimal for S'.

Since the partition of the L unit balls is optimal when applying the dynamic program (8), then if it is applied on S',

$$\sum_{j \in P_i} g_j^{i,S'}(a_j) = opt(S'), \tag{11}$$

where $\sum_{j \in P_i} g_j^{i,S'}(a_j)$ is the value of the profit when applying the dynamic program on S^i.

Since $S^i \subset S$ and L unit balls are used in both cases,

$$\sum_{j \in P_i} g_j^i(a_j) \geq \sum_{j \in P_i} g_j^{i,S^i}(a_j). \tag{12}$$

From (9), (11) and (12) we get:

$$\sum_{j \in P_i} g_j^i(a_j) \geq opt - dis^i.$$

Since $DIS^1, DIS^2, \ldots, DIS^l$ are disjoint, their union adds at most up to OPT, and,

$$\sum_{i=1}^{l} \sum_{j \in P_i} g_j^i(a_j) \geq \sum_{i=1}^{l} [opt - dis^i] \geq (l-1)opt,$$

We get,

$$\max_i \{f_l^i(L)\} \geq \frac{1}{l} \sum_{i=1}^{l} \sum_{j \in P_i} g_j^i(a_j) \geq \left(1 - \frac{1}{l}\right) opt,$$

and we conclude, $f(L) \geq r_A(1 - \frac{1}{l})opt$. ∎

The local algorithm in this case is straightforward. The finite space is partitioned to strips until small cubes are created. For each cube the number of unit balls L' used for its cover is determined by exhaustive search. Finding the optimal solution using at most L' unit balls can be done in $O(n^d L')$ time.

References

[1] A.A. Ageev and M.I. Sviridenko, "Approximation algorithms for the maximum coverage and Max cut with given sizes of parts," *Proc. 8th IPCO* (1999), 17-30.

[2] V. Barnett and T. Lewis, "Outliers in statistical data" *Wiley* 1984.

[3] M. Charikar, S. Khuller, D.M. Mount and G. Narasimhan, "Algorithms for facility location problems with outliers", *SODA*,2001,642-651.

[4] R.N. Dave and R. Krishnapuram, "Robust Clustering Methods: A Unified View", *IEEE Transactions on Fuzzy Systems* **5** 1997, 270-293.

[5] D.-Z. Du and P.M. Paradalos, "Handbook of Combinatorial Optimization" *Kluwer Academic Publishers* 1998, 261-329.

[6] M. Ester, H. Kreigel, J. Sander and X. Xu, "A density based Algorithm for Discovering Clusters in Large Spatial Databases with Noise", *KDD-96* 1996, 226-231.

[7] M. Ester, Hans-Peter Kriegel and Xiaowei Xu "A Database Interface for Clustering in Large Spatial Databases", *KDD-95* (1995).

[8] R.J Fowler, M. S. Paterson and S. L. Tanimoto, "Optimal packing and covering in the plane are NP-complete," *Information Processing Letters* **12** (1981), 290-308.

[9] T. F. Gonzalez, "Covering a set of points in multidimensional space," *Information Processing Letters* **40** (1991), 181-188.

[10] S. Guha, R. Rastogi and K. Shim, "CURE: A Efficient Clustering Algorithm for large Databases", *Proc. of the ACM SIGMOND Conference on Management of Data* (1998).

[11] J.W.Hanand M.Kamber, "Data Mining: Concepts And Techniques", *San Francisco: Morgan Kaufmann Publishers*, (2001).

[12] D.S.Hochbaum and W.Maass, "Approximation schemes for covering and packing problems in image processing and VLSI," *Journal of ACM*, **32** (1985), 130-136.

[13] M.F. Jiang, S.S Tseng and C.M. Su, "Two-phase clustering process for outliers detection", *Pattern Recognition Letters* **22** (2001), 691-700.

[14] S. Khuller, A. Moss and J. Naor, "The budgeted maximum coverage problem," *Information Processing Letters* **70** (1999), 290-308.

[15] R. Kannan, S. Vempala and A. Vetta, "On clustering: good, bad and spectral," FOCS (2000), 367-377.

[16] R. Nag and J. Han , "Efficient and Effective Clustering Methods for Spatial Data Mining", *Proceedings of the 20th VLDB conference* (1994) 145-155.

[17] G. J. McLachlan and T. Krishnan, "The EM Algorithm and Extensions", *Wiley-Interscience* (1996) .

[18] C. F. Olson "Parallel Algorithms for Hierarchical Clustering", Technical report, Computer Science Division, Univ. of California at Berkley, (1993).

[19] Y. Pawitan, "In All Likelihood: Statistical Modelling and Inference Using Likelihood",*Oxford University Press* (2000).

[20] S. Richardson and P.J. Green, "On Bayesian Analysis of mixtures with an Unknown number of components," *J. R. Stat. Soc. B* **59** (1997), 731-792.

[21] R.A. Redner and H.F. Walker, "Mixture densities, maximum likelihood and the EM algorithm," *SIAM Review* **26** 1984, 195-239.

[22] G. Schwarz, "Estimating the dimension of a model," *The Annals of Statistics* **6** (1978), 461-464.

[23] D. Spielman and S-H Teng, "Spectral partitioning works: planar graphs and finite element meshes," *Proc. of 37th FOCS* (1996), 96-105.

[24] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22** (2000), 888-905.

[25] L. Xu and M. Jordan, "On the convergence properties of the EM Algorithm for Gaussian Mixtures", *Neural Computation* **8** (1996), 129-151.

[26] X. Zhuang, Yan Huang, K. Palaniappan and Yunxin Zhao "Gaussian Mixture Density Modelling, and Applications", *IEEE Transactions on Image Processing* **5** (1996), 1293-1301.

[27] J. Zhang and Y. Leung, "Robust Clustering by Pruning Outliers", *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics* **33** (2003), 983-998.

[28] T. Zhang, R. Ramakrishnan and M. Livny, "BRITCH: An Efficient Data Clustering Method for Very Large Databases", *Proc. of the ACM SIGMOND Conference on Management of Data* (1996), 103-114.

Appendix A

Let δ denote the Euclidean distance function. Define the distance function between two n node structures as follows:

$$D(s,t) = min_{p \in P} \sqrt{\sum_{(i,j) \in V} [d^t(i,j) - d^s(p(i),p(j))]^2}.$$

Lemma 7.2 *D is a metric*

Proof: Let p_{12} and p_{23} be the permutations that give the minimum distance in the computation of $D(s_1,s_2)$ and $D(s_2,s_3)$ respectively. Then, $D(s_1,s_2) = \delta(s_1, p_{12}(s_2))$ and, $D(s_2,s_3) = \delta(s_2, p_{23}(s_3)) = \delta(p_{12}(s_2), p_{12}p_{23}(s_3))$, and hence $D(s_1,s_2) + D(s_2,s_3) = \delta(s_1, p_{12}(s_2)) + \delta(p_{12}(s_2), p_{12}p_{23}(s_3)) \geq \delta(s_1, p_{12}p_{23}(s_3)) \geq D(s_1,s_3)$. ∎

Appendix B

Let P denote the set of mappings $p : T_t \rightarrow T_s$ such that the color of the source and the target is the same. The distance between the molecule structures t and s is $D(s,t) = min_{p \in P} \sqrt{\sum_{(i,j) \in V} [d^t(i,j) - d^s(p(i),p(j))]^2}$. In order to find the mapping $p \in P$ that minimizes the distance, one need not go over all members of P. Since a molecule structure is embedded in $I\!R^3$ we can check only mappings that are likely to minimize the sum. We find such mappings by mapping three nodes of V_t to three nodes of V_s, such that the colors of the nodes agree. This mapping will determine a coordinates to which both t and s are then transformed. It is left to map the remaining nodes of t and s. Such a mapping will be done according to the position of the nodes in $I\!R^3$ and their colors. Basically each node of t should be mapped to the closest node of its color in s. If several options exist, the one minimizing the sum should be chosen. There are at most $6 \left(\binom{|V|}{3} \right)^2$ possible transformations between t and s. In the worst case, for each coordinates system one would have to consider then $(n-3)!$ mappings of the remaining nodes of t and s. That may be the case if all nodes of V are of the same

color and very close to each other. But in such cases the distance function is very well approximated by choosing any one of the mappings. Since the distance between two nodes in a molecule structure is between 2 and 12, such events do not occur with real life data.

Appendix C

Let $\Theta = \{\alpha_1, \ldots, \alpha_K, m_1, \ldots, m_K, \Sigma_1, \ldots, \Sigma_K\}$, where $\alpha_j \geq 0$, $\sum_{j=1}^{K} \alpha_j = 1$. Let

$$P(x|\Theta) = \sum_{j=1}^{K} \alpha_j P(x|m_j, \Sigma_j),$$

where

$$P(x|m_j, \Sigma_j) = \frac{1}{\sqrt{2\pi|\Sigma_j|}} \exp^{-\frac{1}{2}(x-m_j)^T \Sigma_j^{-1}(x-m_j)},$$

Given k and given $N = |S|$ independent, identically distributed samples $\{x^{(t)}\}_{t=1}^{N}$, we obtain the following log likelihood:

$$l(\Theta) = \log\left(\Pi_{t=1}^{N} P(x^{(t)}|\Theta)\right) = \sum_{t=1}^{N} \log P(x^{(t)}|\Theta).$$

The Θ values that maximize $l(\Theta)$ are the maximum likelihood estimates of the distributions parameters.

Algorithm **EM** is described in Figure 5.

EM

input

1. A set $x = \{x^{(1)}, \ldots, x^{(N)}\}$ *of points in a d dimensional metric space.*

2. An integer K *[the number of distributions].*

3. $\Theta^{(0)} = \{\alpha_1^{(0)}, \ldots, \alpha_K^{(0)}, m_1^{(0)}, \ldots, m_K^{(0)}, \Sigma_1^{(0)}, \ldots, \Sigma_K^{(0)}\}$ *[the initial solution].*

output

$\Theta = (\alpha_1, \ldots, \alpha_K, m_1, \ldots, m_K, \Sigma_1, \ldots, \Sigma_K)$.

$l(\Theta^{(i)})$.

begin

$z := 0$.

$P(x^{(t)}|\Theta^{(0)}) = \frac{1}{\sqrt{2\pi|\Sigma_j^{(0)}|}} \exp^{-\frac{1}{2}(x-m_j^{(0)})^T(\Sigma_j^{(0)})^{-1}(x-m_j^{(0)})}$.

$l(\Theta^{(0)}) = \sum_{t=1}^{N} \log P(x|\Theta^{(0)})$.

while $z := 0$

for $j = 1, \ldots, K$

for $t = 1, \ldots, N$

$$h_j^{(i)}(t) := \frac{\alpha_j^{(i)} P\left(x^{(t)}|m_j^{(i)}, \Sigma_j^{(i)}\right)}{\sum_{i=1}^{K} \alpha_i^{(i)} P\left(x^{(t)}|m_i^{(i)}, \Sigma_i^{(i)}\right)}$$

end for

$\alpha_j^{i+1} := \frac{1}{N} \sum_{t=1}^{N} h_j^{(i)}(t)$

$m_j^{(i+1)} := \frac{\sum_{t=1}^{N} h_j^{(i)}(t) x^{(t)}}{\sum_{t=1}^{N} h_j^{(i)}(t)}$

$\Sigma_j^{(i+1)} := \frac{\sum_{t=1}^{N} h_j^{(i)}(t)[x^{(t)} - m_j^{(i)}][x^{(t)} - m_j^{(i)}]^T}{\sum_{t=1}^{N} h_j^{(i)}(t) x^{(t)}}$

end for

$\Theta^{(i+1)} := (\alpha_1^{(i+1)}, \ldots, \alpha_K^{(i+1)}, m_1^{(i+1)}, \ldots, m_K^{(i+1)}, \Sigma_1^{(i+1)}, \ldots, \Sigma_K^{(i+1)})$.

$P(x|\Theta^{(i+1)}) = \frac{1}{\sqrt{2\pi|\Sigma_j^{(i+1)}|}} \exp^{-\frac{1}{2}(x-m_j^{(i+1)})^T(\Sigma_j^{(i+1)})^{-1}(x-m_j^{(i+1)})}$.

$l(\Theta^{(i+1)}) = \sum_{t=1}^{N} \log P(x^{(t)}|\Theta^{(i+1)})$

if $l(\Theta^{(i+1)}) \leq l(\Theta^{(i)})$.

then

$z = 1$.

end if

end while

return $\Theta^{(i)}, l(\Theta^{(i)})$.

end EM

Figure 5: Algorithm **EM**