A Comparative Study on the Influence of Similarity Measures in Hierarchical Clustering in Complex Distributed Object-Oriented Databases

ADRIAN SERGIU DARABANT, HOREA TODORAN, OCTAVIAN CRET, GEORGE CHIS
Dept. of Computer Science
Babes Bolyai University, Technical University
1, Kogalniceanu and 15 C-tin Daicoviciu, Cluj Napoca,
ROMANIA

Abstract: - Class fragmentation is an essential phase in the design of Distributed Object Oriented Databases (DOODB). Due to their semantic similarity with the purpose of database fragmentation (obtaining sets of similar objects with respect to the user applications running in the system), clustering algorithms have recently begun to be investigated in the process of database fragmentation. This work proposes a study on the impact of different similarity measures applied in hierarchical agglomerative clustering algorithms for horizontal fragmentation of classes with complex attributes. This study would eventually help finding formal, automatic, approaches in choosing a particular similarity measure in accordance with: the applied clustering algorithm, the structure of the database inheritance/aggregation hierarchies, the semantics of data, etc.

Key-Words: - Distributed database design, horizontal fragmentation, data mining methods, performance evaluation.

1 Introduction

As opposed to centralized databases where the design phase handles only logical and physical data modelling, the design process in Distributed Object Oriented Databases involves both data partitioning and allocation to the nodes of the system. This process is usually called database fragmentation and is an important aspect of distributed database design. Horizontal fragmentation, in Object Oriented Database Systems, distributes class instances into fragments.

Many of the existing Object Oriented (OO) fragmentation approaches are usually inspired from the relational fragmentation techniques. While this proves to be a good starting point for approaching the fragmentation problem, there is definitely a limit in applying these techniques to data models featuring all the complex characteristics of a real OO model.

As a result, new approaches for OO database fragmentation are emerging. While some of them are based on graph theory, others use clustering techniques for splitting the classes and their extensions into fragments. In this paper we study the performance influence of similarity measures in clustering-based object-oriented fragmentation methods on complex class hierarchies.

In [11, 12, 13, 14] we have proposed some new database fragmentation techniques based on data mining methods. In this paper we will study an important aspect – the significance of the similarity measure and its impact on the performance of the resulting database schema.

1.1 Related Work

Fragmentation methods for OODB environments, or flat data models have been generally considered in Karlapalem [2], Ezeife [3], Karlapalem [4][5]. Ravat [6] uses the Bond Energy Algorithm (BEA) for vertical and horizontal fragmentation. Ezeife [7] presents a set of algorithms for horizontally fragmenting models with simple attributes/methods and complex attributes/methods. Bellatreche et al. [8] propose a method that emphasizes the role of queries in the horizontal fragmentation. [11] presents a first fragmentation approach based on hierarchical agglomerative clustering while in [13] the original problem modeling is improved so that complex class hierarchies could be taken in account.

1.2 Contributions

Clustering fragmentation methods in complex class hierarchies proposed in earlier papers are generally based on similarity measures used to determine the similarity between different instances of the same class. Based on this similarity, objects are grouped...
into clusters (fragments). This work proposes a comparative study on the influence of different choices of similarity measures on the performance and quality of the obtained fragmentation.

2 Data Model

The used object-oriented model is one with the basic features as described in the literature [11]. Objects with common attributes and methods are grouped into classes. A class is an ordered tuple \( C=(K,A,M,I) \), where \( K \) is the set of object attributes, \( M \) is the set of methods, \( K \) is the class identifier and \( I \) is the set of instances of class \( C \). Every object in the database is uniquely identified by an OID. Classes are organized in an inheritance hierarchy, in which a subclass is a specialization of its superclass. An OODB is a set of classes from an inheritance hierarchy, with all their instances. There is a special class Root that is the ancestor of all classes in the database.

An entry point into a database is a meta-class instance bound to a known variable in the system. An entry point allows navigation from it to all classes and class instances of its sub-tree (including itself). There are usually more entry points in an OODB.

Given a complex hierarchy \( H \), a path expression \( P \) is defined as \( C_i.A_i. \ldots A_n. n \geq 1 \) where: \( C_i \) is an entry point in \( H \), \( A_i \) is an attribute of class \( C_i \), \( A_i \) is an attribute of class \( C_i \) in \( H \) such that \( C_i \) is the domain of attribute \( A_i \), of class \( C_i.1 \leq i \leq n \). In the general case, \( A_i \) can be a method call. If \( i < n \), then \( A_i \) must return a single complex type value (an object).

3 Vector Space Modelling

Let \( Q=\{q_1, \ldots, q_l\} \) be the set of all queries in respect to which we want to perform the fragmentation. Let \( \text{Pred}=\{p_1, \ldots, p_q\} \) be the set of all atomic predicates \( Q \) is defined on. Let \( \text{Pred}(C)=\{p \in \text{Pred} | p \text{ imposes a condition to an attribute of class } C \text{ or to an attribute of its parent}\} \). Given the predicate \( p \equiv C_i.A_i. \ldots A_n. \theta \) value, then \( p \in \text{Pred}(C) \) if class \( C_i \) is the complex domain of \( A_i.1 \leq i \leq n \), and \( A_n \) has a complex type or simple type.

Given two classes \( C \) and \( C' \), where \( C' \) is subclass of \( C \), \( \text{Pred}(C') \subseteq \text{Pred}(C) \). The reason behind this fact is explained in [11].

We construct the object-condition matrix for class \( C \), \( \text{OCM}(C)=\{a_{ij},1 \leq i \leq \text{Inst}(C), 1 \leq j \leq \text{Pred}(C)\} \), where \( \text{Inst}(C)=\{O_1, \ldots, O_n\} \) is the set of all instances of class \( C \), \( \text{Pred}(C)=\{p_1, \ldots, p_q\} \):

\[
a_{ij} = \begin{cases} 0, & \text{if } p_j(O_i) = \text{false} \\ 1, & \text{if } p_j(O_i) = \text{true} \end{cases}
\]

\[
\forall i=\overline{1,m}, j=\overline{1,n} \}
\]

Each line \( i \) in \( \text{OCM}(C) \) is the object-condition vector of \( O_i \), where \( O_i \in \text{Inst}(C) \). \( \text{OCM}(C) \) is used then to obtain the characteristic vectors for all instances of \( C \). The characteristic vector for object \( O_i \) is \( w_i = (w_{i1}, w_{i2}, \ldots, w_{in}) \), where each \( w_{ij} \) is the ratio between the number of objects in \( C \) respecting the predicate \( p_j \in \text{Pred}(C) \) in the same way as \( O_i \) and the number of objects in \( C \). We denote the characteristic vector matrix as \( \text{CVM}(C) \) [11].

3.1 Derived Fragmentation Modelling

All characteristics of simple attributes and methods have been captured so far. The next part focuses on the class relationships in our vector space model.

We first model the aggregation and association relations.

Given two classes \( C_0 \) (owner) and \( C_M \) (member), where \( C_M \) is the domain of an attribute of \( C_0 \), a path expression traversing this link navigates from instances of \( C_0 \) to one or more instances of \( C_M \). When fragmenting \( C_0 \) we should take in account the fragmentation of \( C_M \).

Let \( \{F_1, \ldots, F_k\} \) be the fragments of \( C_M \). Let \( \text{Agg}(O_i, F_j)=\{O^m \mid O^m \in \text{Inst}(F_j), O \in \text{Inst}(C_0) \} \).

Given the set of fragments for \( C_M \), the attribute-link induced object-condition vectors for derived fragmentation are defined as \( \text{ad}_x = (\text{ad}_{x1}, \text{ad}_{x2}, \ldots, \text{ad}_{xq}) \), where each vector component is expressed by the following formula:

\[
\text{ad}_{ij} = \text{sgn} \left[ \text{Agg}(O_i, F_j) \right]
\]

For an object \( O_i \in \text{Inst}(C_0) \) and a fragment \( F_j \) of \( C_M \), \( \text{ad}_{ij} \) is 1 if \( O_i \) is linked to at least one object of \( F_j \) and is 0 otherwise.

Given the set of fragments for \( C_M \), the attribute-link induced characteristic vectors for derived fragmentation are defined as \( \text{wd}_{ij} = (\text{wd}_{ij1}, \text{wd}_{ij2}, \ldots, \text{wd}_{ijn}) \), where each vector component is expressed by one of the following formulas (two alternatives):

\[
\text{wd}_{ij} = \left[ \bigcup_{O_i \in \text{Inst}(C_i)} \text{Agg}(O_i, F_j) \right]
\]
\[ wd^2_{ij} = \left[ \frac{|\{O_i \in \text{Inst}(C_1) | \text{sgn} \left( \text{Agg}(O_i, F_j) \right) = \text{sgn} \left( \text{Agg}(O_i, F_j) \right) \}|}{|\text{Inst}(C_1)|} \right] \]

\[ wd^2_{ij} \] gives the ratio between the number of objects in fragment \( F_j \) of class \( C_M \) linked to \( O_i \) and the number of all objects in fragment \( F_j \) linked to instances of \( C_M \). Each \( wd^2_{ij} \) component gives the percentage of objects in \( C_M \) that aggregate in the same way as \( O_i \) objects from \( F_j \). Two objects \( O_i \) and \( O_j \) are said to aggregate in the same way if they are both either linked or not linked with objects from \( F_j \). According to the second criteria, two objects are candidate to be placed in the same fragment of \( C_M \) in respect to \( F_j \) if they are both related in the same way to \( F_j \).

Usually, the fragmentation of a class \( C_M \) is performed in two steps: primary fragmentation, according to query conditions, and derived fragmentation, according to the fragments of the member or owner classes. In our case the phases are merged into one single step capturing the semantic of both primary and derived fragmentations. For this we unify the characteristic vector and the attribute-link induced characteristic vectors of each object \( O_i \) of the class \( C_M \) and obtain the extended characteristic vector.

If the class \( C_M \) is linked with classes \( C_{M1}, C_{M2}, \ldots, C_{M_p} \), the extended characteristic vector \( w_{Eij} \) for object \( O_i \in \text{Inst}(C_M) \) is obtained by appending the attribute-link induced characteristic vectors of \( C_{M1}, C_{M2}, \ldots, C_{M_p} \) to the characteristic vector of \( O_i \).

The extended object-condition vector \( a_{Ec} \) for an object \( O_i \) is obtained in the same way by appending its attribute-link induced object-condition vectors to its object-condition vector.

Let \( EOCM(C) \) and \( ECVM(C) \) be the extended object-condition and characteristic matrices for class \( C \).

### 3.2 Similarity between objects

The aim of our method is to group into a cluster those objects that are similar to one another. Similarity between objects is computed using the following pseudometrics:

\[ \cos(w_{Ei}, w_{Ej}) = \frac{\sum_{k=1}^{n} w_{ik} \times w_{jk}}{\sqrt{\sum_{k=1}^{n} (w_{ik})^2} \times \sqrt{\sum_{k=1}^{n} (w_{jk})^2}} \]  

\[ d_M(w_{Ei}, w_{Ej}) = \sum_{k=1}^{n} |w_{ik} - w_{jk}| \]

\[ d_E(w_{Ei}, w_{Ej}) = \frac{\sum_{k=1}^{n} (w_{ik} - w_{jk})^2}{|F_i| \times |F_j|} \]

Given two objects \( O_i \) and \( O_j \), we define the following similarity measures between them in (8):

\[ \text{sim}_{cos}(O_i, O_j) = \cos(w_{Ei}, w_{Ej}) \]

\[ \text{sim}_{M}(O_i, O_j) = 1 - d_M(w_{Ei}, w_{Ej}) \]

\[ \text{sim}_{E}(O_i, O_j) = 1 - d_E(w_{Ei}, w_{Ej}) \]

The cosine similarity is not defined for any two object-condition vectors. For extended vectors that have all components zero the cosine similarity measure is not defined. On the other side having all components zero means that the corresponding object is not referred by any application, so its resemblance with other objects is not significant in the fragmentation process in this case. It should be noted that all characteristic vectors have positive coordinates by definition.

### 4 The Hierarchical Agglomerative Fragmentation

The algorithm presented here is similar to the one in [11] and performs horizontal fragmentation on complex class hierarchies using the numerical database model presented above.

**Algorithm HierarchicalAggFrag**

**Input:** Class \( C_i \), \( \text{Inst}(C) \) to be fragmented, the similarity function \( \text{sim}: \text{Inst}(C) \times \text{Inst}(C) - \{0,1\}, \) \( \text{m} = |\text{Inst}(C)|, 1 < k \leq m \) desired number of fragments, \( EOCM(C), ECVM(C) \).

**Output:** The set of hierarchical clusters \( F = \{F_1, \ldots, F_k\} \).

**Begin**

For \( i = 1 \) to \( \text{Inst}(C) \) do \( F_i = \{O_i\} \);

\( F = \{F_1, \ldots, F_n\} \);

While \( |F| > k \) do

\( (F_u, F_v) := \text{argmax}(F_u, F_v) \{\text{sim}(F_u, F_v)\} \);

\( F_{\text{new}} = F_u \cup F_v \);

\( F = F \setminus (F_u, F_v) \cup \{F_{\text{new}}\} \);

End While;

**End.**

At each iteration the algorithm chooses the two most similar clusters and merges them into a single cluster \( \{\text{argmax}(F_u, F_v) \{\text{sim}(F_u, F_v)\}\} \). As similarity between two clusters \( F_u \) and \( F_v \), the average similarity of all pairs of objects is considered:

\[ \text{sim}(F_u, F_v) = \frac{\sum_{a \in F_u} \sum_{b \in F_v} \text{sim}(a, b)}{|F_u| \times |F_v|} \]
The algorithm always ends up with \( k \) clusters representing the class fragments.

5 Results and Evaluation

This section illustrates the experimental results obtained by applying our fragmentation schemes on real and test object databases. Given a set of queries, we first obtain the horizontal fragments for the classes in the database; afterwards we evaluate the quality and performance of the fragmentation results. It should be noted that the order in which classes are fragmented is significant as it captures the semantic of query path expressions into the fragmentation process [12].

The sample object database in Fig. 1 represents a reduced university database. The inheritance hierarchy is shown in Fig. 1. It represents the average results obtained during our tests.

As explained in [11], the EM term calculates the local access cost, while ER calculates the remote relevant access cost for all fragments of a class. The fragmentation is better when the local (irrelevant) costs and the remote relevant access costs are smaller. Globally, PE measures how well fragments fit the object sets requested by queries.

Each measurement considers a set of applications running on the database. They are given in [11, 12, 13, 14]

For measuring the fragmentation quality we use the partition evaluator function presented in [13]. The cost formulas are:

\[
PE(C) = EM^2 + ER^2
\]

\[
EM^2(C) = \sum_{i=1}^{d} \left( \sum_{r=1}^{t} freq_r x Acc_{ir} \right) \left( 1 - \frac{Acc_{ir}}{|F|} \right)
\]

\[
ER^2(C) = \sum_{r=1}^{t} \min \left( \sum_{i=1}^{d} freq_{ir} x Acc_{ir} \right) \left( \frac{Acc_{ir}}{|F|} \right)
\]

As explained in [11], the EM term calculates the local access cost, while ER calculates the remote relevant access cost for all fragments of a class. The fragmentation is better when the local (irrelevant) costs and the remote relevant access costs are smaller. Globally, PE measures how well fragments fit the object sets requested by queries.

Using the given query access frequency and other input data, the fragments above are allocated to \( N \) distributed sites. The presented method uses a simple allocation scheme that assigns fragments to sites where they are most needed.

In Fig. 2 and Fig. 3, M1 conforms to eqn (3) while M2 conforms to eqn (4) for expressing derived fragmentation. Classes are represented in each figure in the order they have been fragmented (from left to right). In each figure we compare the PE values on each fragmented class. It can be seen that all measures perform in about the same way for the first classes, even though the composition of resulting clusters is different. Classes have been fragmented in the same order for both M1 and M2. As we approach the right side of each figure we can see that the different composition of clusters of the already fragmented classes influences the resulting fragments (class Undergrad for example). This leads to a more clear separation in the induced PE costs for the Undergrad class for each similarity measure. It can be seen that the Euclidian similarity has an overall best place, as it obtains the smaller costs. The next measure in terms of performance is Manhattan applied on object-condition vectors (Manhattan ObCond). It can also be seen that generally the M2 method behaves better than M1 in terms of costs. The cosine (COS) similarity has generally the worst results. There are, however, particular situations where it outperforms the other similarity measures. Manhattan similarity applied on characteristic vectors (ManhattanVC) has in almost all cases an average behaviour.

The first conclusion that can be drawn from the above is that besides similarity measures, the length of dependency cycles in the aggregation hierarchy greatly influences the fragmentation results.
As the dependency chain is longer – i.e. the number of classes to be fragmented is higher – the small placement errors of objects in clusters tend to have a negative influence on the fragmentation of the classes at the end of the dependency chain. As long as the dependency chains are very short the choice of the similarity measure is insignificant.

In Fig. 4 and Fig. 5 we show the overall results of primary-only and complex (primary+derived) class fragmentation for all classes and for both ways of constructing the extended characteristic matrices. The left side of both figures contains the results of fragmentation in complex class hierarchies, while the right side displays the results of primary-only fragmentation for each similarity measure (P-COS, P-Euclid, P-ManhattanVC). Both figures show that for the primary-only fragmentation case the choice of the similarity measure doesn’t affect much the resulting fragments. All similarity measures obtain similar costs, leading to the idea that they have an equal clustering power.

As seen in Fig.5 and Fig. 6 and as it has already been noted in [13], the results of primary+derived fragmentation are always better than primary-only fragmentation. This means that the disseminative power of the three similarity measures is not particularly influenced by the type of fragmentation: primary-only or primary+derived. In other words, the Euclidian measure would not behave better just because it is used in the context of primary-only fragmentation or primary+derived fragmentation, compared to the other similarity measures. Both figures show that the best overall result is obtained by the Euclidian measure. In the second place comes the Manhattan measure applied on object condition vectors.

The cosine measure shows an important dependence on the construction of the extended characteristic matrices for derived fragmentation. It performs poorly for method M1 and quite well for method M2. It should be noted though that the cosine similarity cannot always be successfully applied in practice as it depends on the values of the individual components of the vectors. As shown in [14] there are cases where sets of clearly delimited objects cannot be correctly clustered using this measure. Solutions have been proposed to correct this issue in [14]. On the other side, the Euclidian and Manhattan measures do not have this kind of degenerated behaviour.
Finally, the hierarchical agglomerative clustering method is characterized by the fact that once a step is done it can never be undone. This could help increase the gap between the results obtained by different similarity measures, while a more behaved clustering method that would allow correcting misplaced objects in the next few iterations is more likely to better reflect the real differences between similarities.

6 Conclusions and Future Work

This paper presents a comparative study of the influence of three similarity measures in the fragmentation process of an Object Oriented Database with complex class relationships, using a hierarchical agglomerative clustering algorithm. We show that the choice of different similarity measures influences the resulting fragmentation in a real database where the number of classes to fragment and the number of inter-class dependencies is likely to be important. Results on multiple database schemas show that the Euclidean similarity measure generally outperforms the other two similarity measures. Our conclusions are based on initial suppositions that we support and confirm with experimental results. Given the empirical nature of proofs only by experiments, we aim to find formal, mathematical ways for proving the characteristics of different similarity measures applied in the fragmentation process using clustering techniques. This would help improve the positive impression gained by applying clustering methods in the horizontal fragmentation process of distributed databases.

References:


* This work has been partially funded from the CNCSIS research grant A_C No. 1/8 -2005, "Collaborative Information Systems In The Global Economy"