STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LI**, Number 2, 2006

A STUDY ON DISTANCE METRICS FOR PARTITIONING BASED ASPECT MINING

GRIGORETA SOFIA MOLDOVAN AND GABRIELA ŞERBAN

Abstract. The aim of this paper is to make a study on the influence of distance metrics for partitioning based aspect mining. For this purpose, we comparatively present, from the aspect mining point of view, the results of three algorithms in Aspect Mining, *kAM* ([3]), *HAM* ([6]) and *GAAM* ([5]), for different distance metrics. The evaluation is based on a set of quality measure that we have previously defined in [1] and [2], and a case study is also reported. We introduce three different criteria on which our study is based.

Keywords: aspect mining, distance metrics, clustering.

1. Introduction

1.1. **Aspect Mining.** *Separation of concerns* ([13]) is a very important principle of software engineering that, in its most general form, refers to the ability to identify, encapsulate and manipulate those parts of software that are relevant to a particular concept, goal, or purpose. Some of the benefits of a good separation of concerns are reduced software complexity, improved comprehensibility, limited impact of change, easy evolution and reuse.

Aspect Oriented Programming (AOP) ([10]) provides means to encapsulate concerns which cannot be modularized using traditional programming techniques. These concerns are called *crosscutting concerns*. Logging and exception handling are well known examples of crosscutting concerns. Aspect oriented paradigm offers a powerful technology for supporting the separation of crosscutting concerns. Such a concern is explicitly specified as an *aspect*. Aspects encapsulate the implementation of a crosscutting concern. A special tool, called *weaver*, integrates a number of aspects to obtain the final software system.

In order to apply AOP principles to legacy software systems, it is necessary to analyze the existing implementation to discover the crosscutting concerns and

Received by the editors: November 5, 2006.

2000 *Mathematics Subject Classification.* 68N99, 62H30.

1998 *CR Categories and Descriptors.* D.2.7 [**Software Engineering**]: Distribution, Maintenance, and Enhancement –*Restructuring, reverse engineering, and reengineering*; I.5.3 [**Computing Methodologies**]: Pattern Recognition – *Clustering*.

refactor them into aspects. The research on *aspect mining* refers to the identification and analysis of non-localized crosscutting concerns throughout an existing legacy software system ([9]). The goal of aspect mining is to support aspect-oriented refactoring to improve software comprehensibility, reusability and maintainability.

1.2. **Related Work.** In [4] a vector space model based clustering approach in aspect mining is proposed. This approach is improved in [3], by defining a new *k-means* based clustering algorithm in aspect mining (*kAM*).

In [1], a part of a formal model for clustering in aspect mining is introduced and a set of quality measures for evaluating the results of clustering based aspect mining techniques is presented. This model is extended in [2].

A Hierarchical clustering algorithm in Aspect Mining (*HAM*) is introduced in [6]. In [5], the problem of identifying crosscutting concerns is defined as a search problem in a *graph* and *GAAM* algorithm (*Graph Algorithm in Aspect Mining*) is introduced for this purpose.

Each of *kAM*, *HAM* and *GAAM* algorithms make use of distance metrics between multi-dimensional vectors in order to determine the distance (dissimilarity) between the methods from a software system to be mined.

In this paper we study the influence of distance metrics on the results obtained by the above mentioned algorithms. We intend to identify the most suitable distance metric between methods. The comparison of the results is from the aspect mining point of view and is made based on some quality measures that were previously introduced in [1] and [2].

The paper is structured as follows. A theoretical model for the problem of crosscutting concerns identification is given in Section 2. Section 3 presents a vector space model based partitioning approach in aspect mining. The comparative results for different distance metrics, based on some quality measures, is presented in Section 4. Some conclusions and further work are given in Section 5.

2. Background

In [5] the problem of identifying *crosscutting concerns* is defined as a problem of identifying a partition of a software system.

Let $S = \{s_1, s_2, \ldots, s_n\}$ be a software system, where $s_i, 1 \leq i \leq n$, is an element from the system. An *element* can be a statement, a method, a class, a module, etc. We denote by n ($|S|$) the number of elements of the system.

In the following, we will consider a crosscutting concern as a set of elements $C \subset S$, $C = \{c_1, c_2, ..., c_{cn}\}$, elements that implement this concern. Let $CCC = \{C_1, C_2, ..., C_q\}$ be the set of all crosscutting concerns that exist in the system S. The number of crosscutting concerns in the system S is $q = |CCC|$. Let

$NCCC = S - (\bigcup_{i=1}^{q} C_i)$ be the set of elements from the system S, elements that are not used to implement any crosscutting concerns.

Definition 1. ([2]) ***Partition of a system*** S.
The set $\mathcal{K} = \{K_1, K_2, ..., K_p\}$ *is called a* ***partition*** *of the system* S *iff* $1 \leq p \leq n$, $K_i \subseteq S, K_i \neq \emptyset, \forall\ i, 1 \leq i \leq p$, $S = \bigcup_{i=1}^{p} K_i$ *and* $K_i \cap K_j = \emptyset,\ \forall\ i, j, 1 \leq i, j \leq p, i \neq j$.

In the following we will refer to K_i as the i-th *cluster* of $\mathcal{K}$.

In fact, the problem of aspect mining can be viewed as the problem of finding a partition $\mathcal{K}$ of the system S such that $CCC \subset \mathcal{K}$. Definition 2 introduces the notion of **partitioning aspect mining technique**, that is used in this paper.

Definition 2. ([5]) ***Partitioning aspect mining technique.***
Let $\mathcal{T}$ *be an aspect mining technique and* S *a software system to be mined. We say that* $\mathcal{T}$ *is a* ***partitioning aspect mining technique*** *if the result obtained by* $\mathcal{T}$ *is a* ***partition*** (Definition 1) $\mathcal{K}$ *of* S.

We mention that *kAM*, *HAM* and *GAAM* algorithms determine partitions of a software system, but using different approaches, and are used in the partitioning aspect mining techniques introduced in [3], [6] and [5], The first two algorithms use clustering ([11]) approaches and the last algorithm uses a graph based approach.

3. Vector Space Model based Partitioning in Aspect Mining

Let us consider a software system S to be mined.

In approaches [3], [5] and [6], the software system S is composed of a set of methods $m_1, m_2, \ldots, m_n$, so the objects to be grouped (partitioned) are the methods from S . The methods belong to the application classes or are called from the application classes.

Based on the vector space model, each method is considered as an l-dimensional vector: $m_i = (m_{i1}, \ldots, m_{il})$.

Crosscutting concerns in non AO systems have two symptoms: *code scattering* and *code tangling*. *Code scattering* means that the code that implements a crosscutting concern is spread across the system, and *code tangling* means that the code that implements some concern is mixed with code from other (crosscutting) concerns.

We have considered a vector-space model that illustrate only the *scattered code* symptom. Future development will also consider the *code tangling* symptom.

The vector associated with a method m is $\{FIV, B_1\ B_2, ...B_{l-1}\}$, where FIV is the fan-in value ([12]) of m and B_i $(1 \leq i \leq l-1)$ is 1, if the method m is called from a method belonging to the application class AC_i, and 0, otherwise.

As in a vector space model based clustering ([11]), we consider the *distance* between two methods m_i and m_j as a measure of dissimilarity between them.

In our approach we will consider three possible distance metrics between methods:

- *Euclidian Distance.* The distance between m_i and m_j is expressed as:

$$d_E(m_i, m_j) = \sqrt{\sum_{k=1}^{l}(m_{ik} - m_{jk})^2} \tag{1}$$

- *Hamming Distance.* The distance between m_i and m_j is expressed as:

$$d_H(m_i, m_j) = |\{k|1 \leq k \leq l,\ m_{ik} \neq m_{jk}\}| \tag{2}$$

- *Manhattan Distance.* The distance between m_i and m_j is expressed as:

$$d_M(m_i, m_j) = \sum_{k=1}^{l} |m_{ik} - m_{jk}| \tag{3}$$

4. Experimental Evaluation

In order to evaluate the results of *kAM*, *HAM* and *GAAM* algorithms for different distance metrics, from the aspect mining point of view, we use four quality measure defined in [1] (*DISP*, *DIV*, *PREC* and *PAM*) and two quality measures defined in [2] (*ACTE* and *PANE*). We mention that the last two measures are considered for the case in which the software system consists of a set of methods.

These measures are applied on a case study and the comparative results are reported in Subsection 4.1.

We make the comparison of the obtained results based on three criteria:

(1) **Partitioning criterion**. The degree to which each crosscutting concern is well placed in the partition. For this criterion we use measures *DISP* and *DIV* ([1]).
(2) **Selection criterion**. How well the clusters to be analyzed are chosen. For this criterion we use measures *PREC* and *ACTE* ([1, 2]).
(3) **Ordering criterion**. How relevant is the order in which the clusters are analyzed. For this criterion we use measures *PAM* and *PANE* ([1, 2]).

In order to compare two partitions of a software system S from the above defined criteria, we introduce Definitions 3, 4 and 5. The definitions are based on the properties of the quality measures defined in [1] and [2].

Definition 3. *If $\mathcal{K}_1$ and $\mathcal{K}_2$ are two partitions of the software system S, CCC is the set of crosscutting concerns in S, then $\mathcal{K}_1$ is **better** than $\mathcal{K}_2$ from the **partitioning** criterion point of view iff the following inequalities hold:*

$$DISP(CCC, \mathcal{K}_1) \geq DISP(CCC, \mathcal{K}_2), DIV(CCC, \mathcal{K}_1) \geq DIV(CCC, \mathcal{K}_2).$$

Definition 4. *If $\mathcal{K}_1$ and $\mathcal{K}_2$ are two partitions of the software system S, CCC is the set of crosscutting concerns in S and $\mathcal{T}$ is a partitioning aspect mining technique, then $\mathcal{K}_1$ is **better** than $\mathcal{K}_2$ from the **selection** criterion point of view iff the following inequalities hold:*

$$PREC(CCC, \mathcal{K}_1, \mathcal{T}) \geq PREC(CCC, \mathcal{K}_2, \mathcal{T}),$$

$$ACTE(CCC, \mathcal{K}_1, \mathcal{T}) \geq ACTE(CCC, \mathcal{K}_2, \mathcal{T}).$$

Definition 5. *If $\mathcal{K}_1$ and $\mathcal{K}_2$ are two partitions of the software system S, CCC is the set of crosscutting concerns in S and $\mathcal{T}$ is a partitioning aspect mining technique, then $\mathcal{K}_1$ is **better** than $\mathcal{K}_2$ from the **ordering** criterion point of view iff the following inequalities hold:*

$$PAM(CCC, \mathcal{K}_1) \leq PAM(CCC, \mathcal{K}_2), PANE(CCC, \mathcal{K}_1) \leq PANE(CCC, \mathcal{K}_2).$$

Remark 1. *If at least one of the inequalities from Definitions 3, 4 and 5 is not satisfied, we cannot decide which of the partitions $\mathcal{K}_1$ or $\mathcal{K}_2$ is better related to its corresponding criterion.*

4.1. **Results.** In order to evaluate the results of the algorithms presented in [3], [6] and [5] we have considered as case study Carla Laffra's implementation of Dijkstra algorithm ([8]).

This case study is a Java applet that implements Dijkstra algorithm in order to determine the shortest path in a graph. It was developed by Carla Laffra and consists of **6** classes and **153** methods.

In this subsection we comparatively present the results obtained after applying the selected algorithms, for the vector space model and distance metrics defined in Section 3, with respect to the quality measures, for the case study presented above.

We mention that in the *analysis* step ([3], [5], [6]) for identifying the crosscutting concerns from a software system only a part of the obtained clusters are analyzed, i.e., the clusters whose distance from 0_l point is greater than a given threshold, α. Because α depends on the distance metric used for partitioning, in Table 1 we give the values for this threshold and in Table 2 we present the comparative results.

The reasons for choosing the values from Table 1 for the threshold α, from the aspect mining point of view, are as follows:

- For the *Euclidian* distance metric we analyze only the methods that are called from at least two different contexts.

Distance metric	α
d_E	2
d_H	3
d_M	3

TABLE 1. The values of the threshold α for the distance metrics.

- For the *Manhattan* distance metric the inequality (4) holds:

$$d_M(m, 0_l) \geq \frac{3}{2}\dot{d}_E(m, 0_l), \forall m \in S \tag{4}$$

- For the *Hamming* distance metric we analyze only the methods that are called from at least two different classes.

Algorithm	Distance metric	DISP	DIV	PAM	PREC	ACTE	PANE
kAM	d_E	0.75	0.8854	0.1486	1	0.6667	0.2009
kAM	d_H	0.75	0.8910	0.3316	0.25	0.25	0.6846
kAM	d_M	0.75	0.8854	0.1633	1	0.66	0.2058
HAM	d_E	0.75	0.8981	0.4199	0.5	0.5	0.4493
HAM	d_H	0.75	0.8988	0.3545	0.25	0.25	0.6944
HAM	d_M	0.75	0.8981	0.4183	0.5	0.5	0.4673
GAAM	d_E	0.75	0.8333	0.4133	0.5	0.5	0.4493
GAAM	d_H	0.75	0.8583	0.6111	0.25	0.25	0.6601
GAAM	d_M	0.75	0.8333	0.4117	0.5	0.5	0.4477

TABLE 2. The values of the quality measures for LaffraGraph case study.

From Table 2, based on Definitions 3, 4 and 5 we observe the following:

- For *kAM* algorithm we conclude that:
  - **Partitioning criterion**. Better results are obtained for *Hamming distance*, followed by *Euclidian distance* and *Manhattan distance* (the last two metrics provide the same results).
  - **Selection criterion**. Better results are obtained for *Euclidian distance* and *Manhattan distance*, followed by *Hamming distance* (the first two metrics provide the same results).
  - **Ordering criterion**. Better results are obtained for *Euclidian distance*, followed by *Manhattan distance*, and then by *Hamming distance*.
- For *HAM* algorithm we conclude that:
  - **Partitioning criterion**. Better results are obtained for *Hamming distance*, followed by *Euclidian distance* and *Manhattan distance* (the last two metrics provide the same results).

  - **Selection criterion**. Better results are obtained for *Euclidian distance* and *Manhattan distance*, followed by *Hamming distance* (the first two metrics provide the same results).
  - **Ordering criterion**. We cannot decide which of the distance metrics provide better results, because not all the inequalities from Definition 5 are simultaneously satisfied. This lack of decidability, in our opinion, may be determined by the vector space model and the method chosen for ordering the clusters.
- For *GAAM* algorithm we conclude that:
  - **Partitioning criterion**. Better results are obtained for *Hamming distance*, followed by *Euclidian distance* and *Manhattan distance* (the last two metrics provide the same results).
  - **Selection criterion**. Better results are obtained for *Euclidian distance* and *Manhattan distance*, followed by *Hamming distance* (the first two metrics provide the same results).
  - **Ordering criterion**. Better results are obtained for *Manhattan distance*, followed by *Euclidian distance*, and then by *Hamming distance*.

We observe that, for the *partitioning* and *selection* criteria, the classification of distance metrics is the same for all algorithms. But, for the *ordering* criterion a general classification cannot be decided.

In order to solve the lack of decidability for the *ordering* criterion, we intend:

- To improve the vector space model by taking into account the *code tangling* symptom.
- To find a better order for the clusters analysis.

5. Conclusions and Future Work

In this paper we have comparatively present the results obtained by three algorithms in aspect mining (*kAM* ([3]), *HAM* ([6]) and *GAAM* ([5])) for different distance metrics.

In order to evaluate the obtained results from the aspect mining point of view, we have used a set of quality measures defined in [1] and [2].

We have also given definitions in order to compare two partitions from the aspect mining point of view, based on three different criteria. Based on these definitions, we have comparatively analyzed the influence of distance metrics on the results obtained by the selected partitioning aspect mining techniques.

Further work can be done in the following directions:

- To improve the vector space model by also considering the *code tangling* symptom.
- To identify the most suitable values for the threshold α and the proper order for analyzing the clusters in a partition.

- To identify new distance metrics suitable in aspect mining, considering weighted attributes, too.

References

[1] Moldovan, G.S., Serban, G., "Quality Measures for Evaluating the Results of Clustering Based Aspect Mining Techniques", In: Proceedings of Towards Evaluation of Aspect Mining(TEAM), ECOOP, 2006, pp. 13–16.

[2] Moldovan, G.S, Serban, G., "Clustering Based Aspect Mining Formalized", WSEAS Transactions on Computers, Issue 2, Vol.6, 2007, pp. 199–206

[3] Serban, G., Moldovan, G.S., "A new k-means based clustering algorithm in aspect mining", In: 8th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC'06), 2006, pp. 60–64.

[4] Moldovan, G.S., Serban, G., "Aspect Mining using a Vector-Space Model Based Clustering Approach", In: Proceedings of Linking Aspect Technology and Evolution (LATE) Workshop, 2006, pp. 36-40.

[5] Serban, G., Moldovan, G.S., "A graph algorithm for identification of croscutting concerns", Studia Universitatis "Babes-Bolyai", Informatica, LI(2), 2006, to appear.

[6] Serban, G., Moldovan, G.S., "A New Hierarchical Agglomerative Clustering Algorithm in Aspect Mining", The 24th International Symposium on Theoretical Aspects of Computer Science, Aachen, Germany, 2007, submitted.

[7] Marin, M., van, A., Deursen, Moonen, L., "Identifying Aspects Using Fan-in Analysis", In: Proceedings of the 11th Working Conference on Reverse Engineering (WCRE2004), IEEE Computer Society, 2004, pp. 132–141.

[8] Laffra, C.: "Dijkstra's Shortest Path Algorithm", http://carbon.cudenver.edu/∼hgreenbe /courses/dijkstra/DijkstraApplet.html, 1996.

[9] Magiel Bruntink, Arie van Deursen, Remco van Engelen, and Tom Tourwe. On the Use of Clone Detection for Identifying Crosscutting Concern Code. *IEEE Transactions on Software Engineering*, 31(10), 2005, pp. 804–818.

[10] Kiczales, G., Lamping, J., Menhdhekar, A., Maeda, C., Lopes, C., Loingtier, J.M., Irwin, J., "Aspect-Oriented Programming", In *Proceedings European Conference on Object-Oriented Programming*, volume 1241, Springer-Verlag, 1997, pp. 220–242.

[11] Jain, A., Murty, M.N., Flynn, P., "Data clustering: A review", *ACM Computing Surveys*, 31(3), 1999, pp. 264–323.

[12] Marin, M., van Deursen, A., Moonen, L., "Identifying Aspects Using Fan-in Analysis", In *Proceedings of the 11th Working Conference on Reverse Engineering (WCRE2004)*, IEEE Computer Society, 2004, pp. 132–141.

[13] David. L. Parnas. On The Criteria To Be Used in Decomposing Systems Into Modules. *Communications of the ACM*, 15(12), 1972, pp. 1053–1058.

Babeş-Bolyai University, Faculty of Mathematics and Computer Science, Cluj-Napoca, Romania

E-mail address: grigo@cs.ubbcluj.ro

Babeş-Bolyai University, Faculty of Mathematics and Computer Science, Cluj-Napoca, Romania

E-mail address: gabis@cs.ubbcluj.ro