Available online at www.sciencedirect.com
SCIENCE DIRECT®
Pattern Recognition 38 (2005) 2256–2269
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/patcog

A statistics-based approach to control the quality of subclusters in incremental gravitational clustering

Chien-Yu Chen[a,*], Shien-Ching Hwang[a], Yen-Jen Oyang[a,b]

[a]*Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan*
[b]*Graduate Institute of Networking and Multimedia, National Taiwan University, Taipei 106, Taiwan*

Received 30 July 2004; received in revised form 21 March 2005; accepted 21 March 2005

Abstract

As the sizes of many contemporary databases continue to grow rapidly, incremental clustering has emerged as an essential issue for conducting data analysis on contemporary databases. An incremental clustering algorithm refers to an abstraction of the distribution of the data instances generated by the previous run of the algorithm and therefore is able to cope well with the ever-growing contemporary databases. There are two main challenges in the design of incremental clustering algorithms. The first challenge is how to reduce information loss due to the data abstraction (or summarization) operations. The second challenge is that the clustering result should not be sensitive to the order of input data. This paper presents the GRIN algorithm, an incremental hierarchical clustering algorithm for numerical datasets based on the gravity theory in physics. In the design of GRIN, a statistical test aimed at reducing information loss and distortion is employed to control formation of subclusters as well as to monitor the evolution of the dataset. Due to the statistical test-based summarization approach, GRIN is able to achieve near linear scalability and is not sensitive to input ordering.

Keywords: Data clustering; Hierarchical clustering; Incremental learning; Gravity theory

1. Introduction

Data clustering is an important mechanism for solving various real-world problems such as segmentation, database compression, vector quantization, and pattern recognition [1–5]. Due to rapidly emerging application domains in recent years such as data mining and bioinformatics, data clustering has attracted a new round of attention [6–8]. One of the main challenges in the design of modern clustering algorithms is that, in many applications, new data instances are continuously added into an already huge database. Therefore, it is impractical to carry out data clustering from scratch whenever new data instances are added into the database. One way to tackle this challenge is to incorporate a clustering algorithm that operates incrementally.

The development of incremental clustering algorithms can be traced back to 1970s [4]. The LEADER [9] algorithm uses a threshold to determine if an instance can be placed in an existing cluster or it should form a new cluster by itself. Many incremental algorithms follow this model for clustering data instances incrementally. COBWEB [10] and CLASSIT [11] are incremental hierarchical clustering algorithms designed for categorical and numerical datasets, respectively. When processing incoming data instances, COBWEB and CLASSIT employ four operations *insert*, *create*, *split*, and *merge* to adjust the hierarchical structure locally. A clustering dendrogram is desired in many applications due to the need of taxonomies [4]. However, both COBWEB

* Corresponding author. Tel.: +886 3 4638800x2185; fax: +886 2 23688675.
E-mail addresses: cychen@mars.csie.ntu.edu.tw (C.-Y. Chen), schwang@mars.csie.ntu.edu.tw (S.-C. Hwang), yjoyang@csie.ntu.edu.tw (Y.-J. Oyang).

0031-3203/$30.00
doi:10.1016/j.patcog.2005.03.005

and CLASSIT could result in highly unbalanced trees [6]. In recent years, several incremental clustering algorithms have been proposed for mining and monitoring evolving datasets [12–16]. Among them, Ribert's algorithm and the BIRCH algorithm keep maintaining a hierarchy as clustering outputs. Ribert's algorithm suffers a higher time complexity when compared with a linear time algorithm and therefore is not suitable for handling large datasets. On the other hand, the BIRCH algorithm [14,16] features low time and space complexity by means of grouping similar instances as a subcluster and using the derived subclusters as the primitives when generating a hierarchy.

Grouping data instances as a subcluster is considered as a process of summarization or data abstraction [4]. Data summarization keeps playing an important role in developing incremental clustering algorithms. Furthermore, as Ganti and Zhang showed in their papers [14,16], grouping data instances as a subcluster provides a good solution when maintaining hierarchies for large datasets incrementally. This idea has also been employed to scale up the hierarchical clustering algorithms successfully [17]. Since the subclusters are the primitives for generating a hierarchy as the clustering results, the quality of subclusters is crucial to the quality of the hierarchy derived.

There are three common issues associated with data summarization or abstraction. The first issue lies in how to choose a threshold while utilizing a fixed threshold to control subclusters. A new instance can be inserted into an existing subcluster as long as the dissimilarity between the new instance and the representative of the subcluster is smaller than a given threshold. Fig. 1(a) shows an example where a global threshold may fail. In Fig. 1(a), the gray balls are data instances in the database so far. As we can see, unreasonable clustering results arise due to the improper threshold. The second issue associated with data abstraction is the information loss due to data abstraction. As illustrated in Fig. 1(b), the distribution of the instances in the cluster is not consistent with the abstraction model employed to summarize a cluster. In this case, the abstraction model is the centroid and the radius of a cluster. When only the centroid and the radius of the subcluster are given, the algorithm will insert new data instances (i.e. the white balls) into wrong clusters. The third issue concerns how to properly monitor the transition of dataset. We observed that the insertions of new data points into an existing subcluster might result in the shifting of distribution within the subclusters. As exemplified in Fig. 1(c), once there are some more new instances falling in the left part of the circle, the elements inside the subcluster is not uniformly distributed any more. Splitting this subcluster would be necessary; otherwise the information loss will be amplified in the remaining clustering process.

Besides, the sensitiveness to the arriving ordering of input data is also a main problem associated with modern incremental clustering algorithms. Improper arriving order makes designing an incremental clustering a more challenging task when data abstraction is considered. Fig. 2 presents

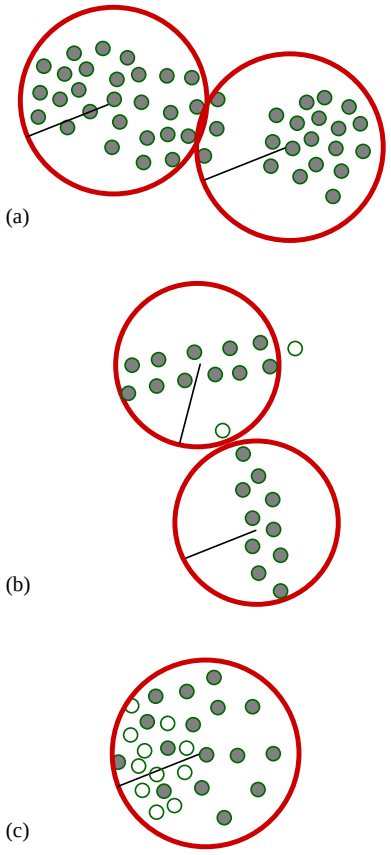

Fig. 1. Problems might happen due to data abstraction. Gray balls stand for the data instances being issued so far, and white balls stand for the new instances that will come later. (a) Flaws arise if using a fixed distance threshold to control the formation of subclusters. (b) New data instances (i.e. the white balls) will fall in wrong subclusters due to information loss after data abstraction has been executed. (c) The subcluster is not homogeneous any more after new data instances are added.

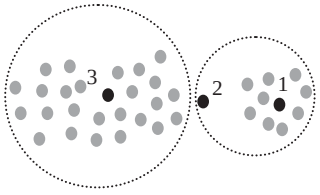

Fig. 2. A case in which the distance-based controlling approach may fail to deliver satisfactory clustering quality due to the improper arriving order of data instances.

an example where data instances arrive in unexpected order. In this example, θ denotes the threshold imposed on the diameters of leaf subclusters and it is assumed that distance (instance 1, instance 2) $< \theta$, distance (instance 2, instance 3) $< \theta$, and distance (instance 1, instance 3) $> \theta$. As the example shows, if the data enters in the following order