Discovering Sequential Patterns from Non-uniform Databases
(Extended Abstract)

Du Dang and X. Sean Wang
ISE Department, George Mason University
Fairfax, VA 22030, USA
xywang@gmu.edu
http://www.ise.gmu.edu/~xywang

Abstract. Databases, when used to collect data in the dynamic world, change over time. As a consequence, different parts of a database that come into the database at different times may contain activities with different characteristics. In other words, the data in the database are “non-uniform”. This non-uniformity of the database gives rise to the idea of a Divide and Conquer strategy that divides a very large and non-uniform database into component parts for better mining. Experiments conducted show that the method is effective under various situations.

1 Introduction

Most of the current mining methods treat a database as a single uniform block of data. In other words, these methods do not look into the characteristics of the data to find out whether there are disparities among these characteristics and then adapt the mining process to deal with these disparities.

Take the example of a grocery chain located in a dynamic environment where the population varies significantly over time, both in number and in social and ethnic structures. The grocery chain tries to discover the buying behaviors or patterns of its customers. Based on the past experience, a group of grocery items that is purchased together by more than 50% of the activities (or transactions) is considered as frequent and will be used as guideline for future replenishment supply. The mining process, operated on the whole database, provides the following statistics: the group of grocery items \{salt, pepper, sugar\} is procured together in more than 90% of the activities, whereas the group of grocery items \{soy sauce, tofu, lomein\} is purchased together in only 18% of the activities. Based on this finding, only the group of grocery items \{salt, pepper, sugar\} is frequent. However, if we look into the details of the activity structures, we find that about 20% of the activities are somewhat closely related, namely, only these 20% of activities include one or more items in the group \{soy sauce, tofu, lomein\}. (These related activities may originate from customers of a particular ethnic group.) Using this group of “related activities” as basis for computation, we find that about 90% of these activities include the group of grocery items \{soy
sauce, tofu, lomein}. So, this group of grocery items should also be considered as frequent and should be purchased together in future replenishment supplies. This interesting behavior may not be discovered if the database of customer activities is considered as uniform.

In order to discover some interesting behaviors or patterns, a database should be treated as “non-uniform”, that is, at its different parts, there exist activities with different characteristics. Non-uniformity is not restricted to the example presented above. In our dynamic world, databases, that collect data on activities and behaviors, are supposed to change significantly over time. The changes relate not only to the number of activities that increases continuously, but also to the characteristics of these activities. As a consequence of these changes, different “parts” of a database would contain activities with different characteristics. Addressing the non-uniformity issue in databases is helpful in many respects.

1. As shown in the above example, recognizing the disparity of activities in various parts of a database would help discover some interesting behaviors or patterns not discoverable if considering a database as a uniform block.
2. By identifying the parts with different characteristics, we will be able to divide the database following the lines of these parts and mine each one separately. This is the idea behind our Divide and Conquer strategy.

The rest of this paper is organized as follows: After some definitions, we will present the characteristics of the non-uniform environment (section 2). We then discuss methods that are applied to non-uniform databases, especially the Divide and Conquer strategy (section 3). The effectiveness of these methods will be shown through some experiments (section 4). Section 5 discusses related work. This paper concludes with a Conclusion and Future Work section.

2 The Non-uniform Environment

An event is represented as \( e = (a_1, a_2, ..., a_m, T) \) where each \( a_i \) \((i = 1, 2, ..., m)\) is an attribute value and \( T \) is a time stamp value. An event is often abbreviated as \( (\bar{a}, T) \) where \( \bar{a} \) represents \( a_1, a_2, ..., a_m \). A finite set of events is called an activity. An event with no time stamp value is an event type (represented as \( \bar{a} \)). An event sequence is defined as \( s = (\bar{a}_1, T_1), (\bar{a}_2, T_2), ..., (\bar{a}_n, T_n) \) where \( T_i \leq T_{i+1} \) for \( i = 1, ..., n – 1 \). A sequence of event types is called a sequence type. A sequence type is a sequential pattern if it satisfies a frequency requirement.

The non-uniformity level \( h(a_1, a_2) \) between two activities \( a_1 \) and \( a_2 \) is the ratio of the number of distinct disparate event types to the total number of distinct event types in these activities. For two activities A and B that have \( i \) and \( j \) distinct event types respectively, and \( p \) and \( q \) are the number of distinct event types that can be found in only one of the activities (\( p \) is specific to A and \( q \) is specific to B), we have \( h = (p + q)/(i + j) = (p + q)/(j + p) \).

The non-uniformity level \( h(c_1, c_2) \) between two component parts\(^1\) \( c_1 \) and \( c_2 \) is the minimum of the levels of non-uniformity between the pairs of activities within

---

\(^1\) A component part is a subset of activities in the database.