In many applications, the content of a database changes over time. For example, new customer sequences are periodically added to a bookstore’s database as the store attracts new customers. Similarly, every visit to a Web site will add a new log to the site’s log database. There are also situations in which we have to delete sequences from the database. As an example, when mining current access patterns of a Web site, we may need to delete some out-of-date logs such as those that are more than a year old. Since an operational database changes continuously, the set of frequent sequences has to be updated incrementally.

One simple strategy is to apply an existing mining algorithm on the updated database. However, this strategy fails to take advantage of the valuable information obtained from a previous mining exercise. We note that this infor-
information is particularly useful if the updated database and the old one share a significant portion of common sequences. An incremental update algorithm that makes use of a previous mining result should therefore be much more efficient than a mining-from-scratch approach.

In this chapter, we analyze and improve the I/O performance of the GSP algorithm (Agrawal & Srikant, 1996). We also study the problem of incremental maintenance of frequent sequences.

The rest of this chapter is organized as follows. We first give a formal definition of the problem of mining frequent sequences and that of incremental update. Then we present the sequence mining algorithm MFS and its candidate generation function MGen. Next, the pruning technique and two incremental algorithms GSP+ and MFS+ are described. Experiment results comparing the performance of the algorithms given before we conclude the chapter.

## Problem Definition

In this section, we formally define the problem of mining frequent sequences and the incremental update problem. We also define some notations to simplify our discussion.

Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of literals called items. An itemset $X$ is a set of items (hence, $X \subseteq I$). A sequence $s = <t_1, t_2, \ldots, t_n>$ is an ordered set of itemsets. The length of a sequence $s$ is defined as the number of items contained in $s$ (denoted by $|s|$). If an item occurs several times in different itemsets of a sequence, the item is counted for each occurrence. For example, if $s = <\{1\}, \{2, 3\}, \{1, 4\}>$, then $|s| = 5$.

Consider two sequences $s_1 = <a_1, a_2, \ldots, a_m>$ and $s_2 = <b_1, b_2, \ldots, b_n>$. We say that $s_1$ contains $s_2$, or equivalently, $s_2$ is a sub-sequence of $s_1$ if there exist integers $j_1, j_2, \ldots, j_n$, such that $1 \leq j_1 < j_2 < \ldots < j_n \leq m$ and $b_1 \subseteq a_{j_1}, b_2 \subseteq a_{j_2}, \ldots, b_n \subseteq a_{j_n}$. We represent this relationship by $s_2 \subseteq s_1$. As an example, the sequence $s_2 = <\{a\}, \{b, c\}, \{d\}>$ is contained in $s_1 = <\{e\}, \{a, d\}, \{g\}, \{b, c, f\}, \{d\}>$ because $\{a\} \subseteq \{a, d\}, \{b, c\} \subseteq \{b, c, f\}$, and $\{d\} \subseteq \{d\}$. Hence, $s_2 \subseteq s_1$.

Given a sequence set $V$ and a sequence $s$, if there exists a sequence $s' \in V$ such that $s \subseteq s'$, we write $s \models V$. In words, $s \models V$ if $s$ is contained in some sequence of $V$. Further, a sequence $s \models V$ is maximal if $s$ is not contained in any other sequence in $V$. That is, $s$ is maximal if there does not exist $s'$ such...
Bulk Construction of Geo-Textual Indices
www.igi-global.com/article/bulk-construction-of-geo-textual-indices/116891?camid=4v1a