Mining Generalized Term Associations: Count Propagation Algorithm

Jonghyun Kahng, Wen-Hsiang Kevin Liao, and Dennis McLeod

Computer Science Department
University of Southern California
Los Angeles, CA 90089-0781
{jkahng, whliao, mcleod}@usc.edu

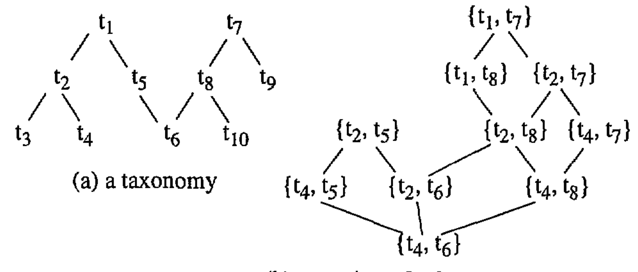
Abstract

We present here an approach and algorithm for mining generalized term associations. The problem is to find co-occurrence frequencies of terms, given a collection of documents each with relevant terms, and a taxonomy of terms. We have developed an efficient Count Propagation Algorithm (CPA) targeted for library applications such as Medline. The basis of our approach is that sets of terms (termsets) can be put into a taxonomy. By exploring this taxonomy, CPA propagates the count of termsets to their ancestors in the taxonomy, instead of separately counting individual termset. We found that CPA is more efficient than other algorithms, particularly for counting large termsets. A benchmark on data sets extracted from a Medline database showed that CPA outperforms other known algorithms by up to around 200% (half the computing time) at the cost of less than 20% of additional memory to keep the taxonomy of termsets. We have used discovered knowledge of term associations for the purpose of improving search capability of Medline.

Introduction

As the processing of business records, documents, and scientific experiments has been automated, discovery of interesting patterns from a huge amount of existing data (data mining) has attracted much attention recently. In particular, the problem of mining association rules from transaction records of department or grocery stores, with item taxonomies (Han and Fu 1995; Srikant and Agrawal 1995) or without them (Agrawal and Srikant 1994; Park, Chen, and Yu 1995; Savasere, Omiecinski, and Navathe 1995), has been studied extensively for the last few years. The most computationally expensive step in this mining process is to count co-occurrence frequencies of two or more items.

We have developed the *Count Propagation Algorithm (CPA)* for mining associations among terms in the presence of a term taxonomy (generalized term associations), and used discovered term associations to improve search capabilities of Medline (a medical information retrieval system). Medline at USC provides extensive information about over 5 million research articles in bio-medical areas. It maintains a taxonomy of 18,000 MeSHs (Medical Subject Head-



(b) a portion of a 2-termset taxonomy

Figure 1: An example of taxonomies (the direction of edges is assumed to be upward).

ings), and each article is assigned a set of relevant MeSHs. CPA has been applied to efficiently determine associations among MeSHs. We were motivated by the observation that Medline data is different from typical transaction records in several aspects. For example, interconnection among terms in the taxonomy is denser, and groups of several terms tend to strongly associate with each other. This is in contrast to transaction records, in which two or three item associations are dominant (Srikant and Agrawal 1995).

The basis of our approach is that the set of terms (termset) can be put into a taxonomy. By exploring this taxonomy, the frequency count of termsets can be propagated to their ancestors in the taxonomy. This is distinct from other approaches which count frequencies of individual termset separately. We found that CPA is more efficient than other algorithms, particularly for counting termsets of large sizes. Our benchmark showed that, for Medline data, CPA outperforms other known algorithms by up to around 200% (half the CPU time) at the cost of less than 20% of additional memory to keep the taxonomy of termsets.

Problem Statement

First, we define terminology and notation, and formally state the problem. $T = \{t_1, t_2, ..., t_m\}$ is the set of terms involved in the application, and X is the taxonomy of those terms (*the taxonomy* in short). X is represented by a directed acyclic graph (DAG), where a node is a term and an edge is directed from a child to its parent term (see Fig-

1. Copyright © 1997, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

ure 1(a) for an example). A term may have multiple parents, implying that multiple classification schemes are represented by the taxonomy.

A *termset* refers to a set of terms, and *n-termset* is a set of n terms. A termset s is *trivial* if it contains one or more ancestor/descendant pairs (e.g., $\{t_2, t_4, t_5\}$). For the rest of this paper, a termset will refer to a non-trivial termset, unless otherwise mentioned. A termset s_1 is an ancestor of a termset s_2 if s_1 and s_2 have the same number of terms and any term in s_1 is either a term in s_2 or an ancestor of one or more terms in s_2. s_1 is a parent of s_2 if s_1 is an ancestor of s_2 and there does not exist a termset s_3 such that s_1 is an ancestor of s_3 and s_3 is an ancestor of s_2 (e.g., parents of $\{t_4, t_6\}$ are $\{t_2, t_6\}$, $\{t_4, t_5\}$, and $\{t_4, t_8\}$). Descendants and children of a termset are similarly defined. *A termset taxonomy* is represented by a DAG, where a node is a termset and an edge is directed from a child to its parent termset. Figure 1(b) shows a portion of a 2-termset taxonomy, including all ancestors of $\{t_4, t_6\}$.

The database in the application is denoted by O, and called a set of term occurrences (or a set of occurrences in short); each *occurrence* is a subset of T. An occurrence corresponds to the set of items in a department transaction record, and the set of terms (MeSHs) assigned to an article in Medline. For a given X, an *extended occurrence* of an occurrence o consists of all terms in o and all of their ancestors.

An occurrence o *supports* a termset s if and only if every term in s is either a term in o or an ancestor of one or more terms in o. For example, $o = \{t_4, t_6, t_{10}\}$ supports $\{t_4\}$, $\{t_6\}$, $\{t_{10}\}$, $\{t_4, t_6\}$, $\{t_4, t_{10}\}$, $\{t_6, t_{10}\}$, $\{t_4, t_6, t_{10}\}$, and all ancestors of these termsets. The *support* of a termset is the percentage of occurrences that support it. A termset with a support over a certain threshold (*a minimum support*) is called a *frequent termset*. It follows from definitions that a termset s is supported by o if and only if s is a subset of the extended occurrence of o; and that all ancestors of a frequent termset are also frequent termsets.

With this terminology, the problem of mining generalized term associations is formally defined as:

Given a term taxonomy, a set of occurrences, and a minimum support, find all frequent termsets.

Count Propagation Algorithm

We now examine the new concepts and mechanism that are necessary for a count propagation scheme. Given a taxonomy X, the *reduced taxonomy* X' is a taxonomy derived from X such that X' has the same terms as X, a term in X' has at most one parent and at most one child, and a term that has multiple parents or multiple children based on X is either a root or a leaf in X'. Figure 2(a)-(b) shows an example of this derivation. In the initial stage of the frequency counting, X' will be heavily fragmented because most terms in X are likely to have either multiple parents or multiple children. In the later stage, however, if terms that are unnecessary for further counting are removed from X, the average height of trees in X' will increase.

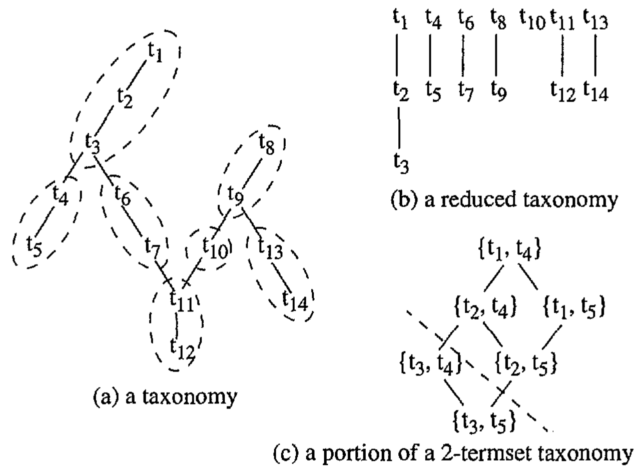


Figure 2: An example of taxonomies.

Given X and X' derived from X, the *augmented occurrence* o' of an occurrence o is derived from o as follows. First, add to o' all terms in o and all their ancestors based on X. Then, if more than one term in the same tree of X' are present in o', keep only the one with the greatest depth and remove all the others. For example, $o' = \{t_3, t_7, t_9, t_{10}, t_{11}\}$ results from $o = \{t_{11}\}$, and $o' = \{t_3, t_5, t_7, t_9, t_{10}, t_{11}\}$ results from $o = \{t_5, t_{11}\}$.

The transformed problem given by X' and O' has an interesting property:

Theorem The set of frequent termsets F' computed from X' and O' includes the set of frequent termsets F computed from X and O, and F' - F consists of termsets that are non-trivial based on X' but trivial based on X.

The extended occurrence of o in O (based on X) is the same as that of the corresponding o' in O' (based on X') so that o and o' supports the same termsets. The only difference is that, among those supported termsets, trivial ones based on X are different from those based on X': the latter is included in the former. This is because X' misses some of ancestor/descendant relationships present in X. For example, t_3 is an ancestor of t_6 in X, but not in X' so that $\{t_3, t_6\}$ is trivial based on X, but non-trivial based on X'. Therefore, X' and O' will result in the same frequent termsets as X and O as far as termsets that are trivial based on X are excluded.

CPA transforms the problem given by X and O into another problem given by derived X' and O', and computes frequent termsets for the transformed problem. Extra care is necessary, of course, for termsets that are trivial based on X. But, for the moment, let's focus on the problem given by X' and O' with no other constraints. X' and termset taxonomies built from it have some simple and nice properties:

- A term t in X' is uniquely identified by an *index* defined as <r, d> where r is the root of the DAG that contains t and d is the depth of t in the DAG (e.g., the index of t_1, t_2, t_3 is <t_1, 0>, <t_1, 1>, <t_1, 2>, respectively). The index makes it easy to identify relationships between two terms. For t_1 with an index <r_1, d_1> and t_2 with an index <r_2, d_2>, t_1 is an ancestor (or a descendant) of t_2 if and only if $r_1 = r_2$ and $d_1 < d_2$ (or $d_1 > d_2$).
- Termset taxonomies built from X' has a simple structure. As for terms, a termset s in such termset taxonomies can be uniquely identified by an *index* which is the set of indices of the terms in s. A termset s with an index {<u_1, d_1>, ..., <u_n, d_n>} is an ancestor of a termset q with an index {<v_1, e_1>, ..., <v_n, e_n>} if and only if s and q are not identical and, for any i, there exists j such that $u_i = v_j$ and $d_i <= e_j$. That is, any ancestor of a termset q can be obtained by simply replacing one or more terms in q with their ancestors. Note that this is not true if X' allowed multiple parents or multiple children because thus obtained termsets may turn out to be trivial.

CPA keeps only candidate termsets in termset taxonomies, and use them for top-down count propagation. Suppose that a termset s is a subset of an augmented occurrence o' so that s and its ancestors are supported by o'. CPA first checks if the root of the DAG that contains s is present in the taxonomy. If it is not, no ancestor of s are present in the taxonomy so that no further counting is necessary. But, if it is present, CPA traverses down the DAG while counting all ancestors of s on the way. Indices of termsets are used to identify the root and to determine whether termsets in the DAG are ancestors of s or not. Figure 2(c) shows a portion of the 2-termset taxonomy constructed from X' in Figure 2(b). Suppose that the termsets below the dotted line are not candidate termsets so that they are not stored in the taxonomy. If an augmented occurrence o' contains a termset {t_3, t_5}, the counting begins at the root {t_1, t_4} and propagates down to {t_2, t_4}, {t_1, t_5}, and {t_2, t_5}.

Coming back to the problem of termsets that are trivial based on X, but non-trivial based on X', it is possible to discard them after frequent termsets are computed. This strategy will however waste a lot of computing time and memory. Therefore, those termsets should rather be removed from termset taxonomies before the counting begins. This operation is simple enough, but it would break down the count propagation scheme just described, if some ancestors of a termset s were removed while s remained. Such a problematic situation would not happen because, if a termset s in the termset taxonomy is trivial based on X, all termsets in the DAG that contains s are also trivial based on X so that the whole DAG drops out of the termset taxonomy. In other words, if a termset s remains in the termset taxonomy, so do all of its ancestors.

In summary, CPA partially extends occurrences O into augmented occurrences O', and partially relies on count propagation using termset taxonomies built from the reduced taxonomy X'. It would be especially efficient when the average height of trees in X' is large so that a large portion of counting is covered by propagation. This is likely to happen when the given taxonomy X is deep and the counting procedure reaches later iterations. (Due to the space limitation, details of the algorithm will be reported elsewhere.)

Performance Evaluation

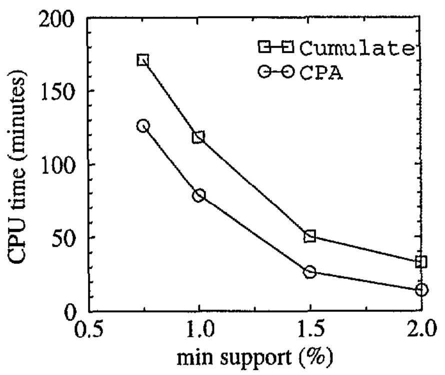
We have implemented CPA and evaluated its performance in comparison with other algorithms. Among the two algorithms, Cumulate and EstMerge, developed by IBM for mining generalized association rules (Srikant and Agrawal 1995), the latter showed only a marginal performance improvement over the former. Han and Fu (1995) reported another algorithm whose strategy is similar to EstMerge. We therefore chose and implemented Cumulate for comparison with CPA. Our benchmark was run on a Sun SPARC-Station 20 running SunOS 4.1.3 with 128 MB of real memory and 439 MB of virtual memory.

We have compared CPA with Cumulate for data sets extracted from a Medline database at USC. Each article in Medline has major and minor MeSHs (Medical Subject Headings) which are selected from the 18,000 MeSH taxonomy. We picked the major MeSH field, and applied the algorithms to discover associations among MeSHs.

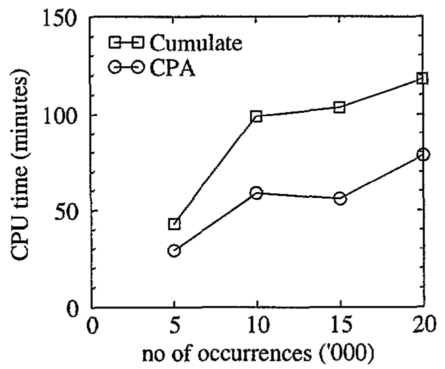
In all cases that we checked on different sets of research articles from Medline, CPA showed better performance than Cumulate, the ratio of the total CPU times varying between 35% and 75% (Figure 3(a)-(b)). Figure 3(c) shows the breakdown of the CPU times into each iteration (the n^{th} iteration computes frequent n-termsets) for a typical case of our benchmark. As expected, the performance gain of CPA over Cumulate becomes higher for later iterations. Memory usages of the two algorithms are compared in Figure 3(d) for the same data set as in Figure 3(c). The figure indicates that CPA uses less than 20% of additional memory to store termset taxonomies near the peak use of the memory (iterations five through nine).

Conclusions

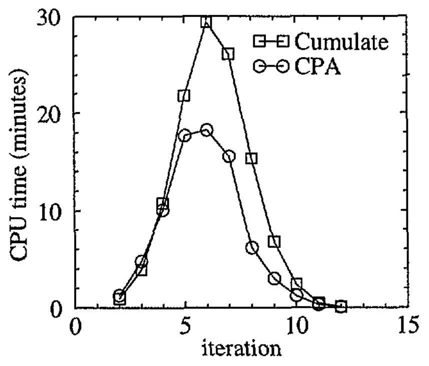
An algorithm CPA for mining generalized term associations was presented, and it was shown that, for Medline data,



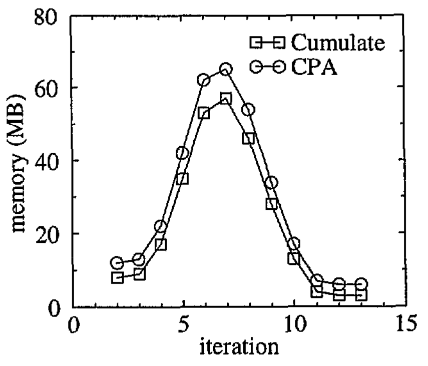
(a) no of occurrences = 20K



(b) min support = 1%



(c) no of occurrences = 20K, min support = 1%



(d) no of occurrences = 20K, min support = 1%

Figure 3: A benchmark on Medline data.

CPA performs better than a known algorithm, Cumulate. Other algorithms for generalized association rules, including Cumulate, have been designed and tested for department transaction records. We found that our data sets from Medline have characteristics very different from the transaction records. We expect that other library data would behave more like Medline data than department transaction records so that other library applications will also benefit from the count propagation technique of CPA. More investigation is on the way to confirm this conjecture.

We have used discovered term associations to improve the search mechanism of Medline, within the context of USC Brain Project (Arbib et al.). In particular, we have developed a query interface that progressively guides users to refine queries by providing MeSHs that might be relevant to their interests. Further, we found that the knowledge of term associations is very useful for measuring relevance of documents to a given query.

Acknowledgments

We would like to express our gratitude to Drs. Ramakrishnan Srikant and Rakesh Agrawal for providing their synthetic data for our experiment.

This research has been funded in part by NIMH under grant no. 5P01MD/DA52194-02, and in part by the Integrated Media Systems Center, a National Science Foundation Engineering Research Center with additional support from the Annenberg Center for Communication at the University of Southern California and the California Trade and Commerce Agency.

References

Agrawal, R., and Srikant, R. 1994. Fast algorithms for mining association rules. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487-499, Santigo, Chile.

Arbib, M. et al. USC Brain Project. Refer to http://www-hbp.usc.edu/.

Han, J., and Fu, Y. 1995. Discovery of multi-level association rules from large databases. In *Proceedings of the 21st International Conference on Very Large Data Bases*, pages 420-431, Zurich, Swizerland.

Park, J. S., Chen, M.-S., and Yu, P. S. 1995. An effective hash-based algorithm for mining association rules. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 175-186, San Jose, CA.

Savasere, A., Omiecinski, E., and Navathe, S. 1995. An efficient algorithm for mining association rules in large databases. In *Proceedings of the 21st International Conference on Very Large Data Bases*, pages 432-444, Zurich, Swizerland.

Srikant, R., and Agrawal, R. 1995. Mining generalized association rules. In *Proceedings of the 21st International Conference on Very Large Data Bases*, pages 407-419, Zurich, Swizerland.