Chapter 4

Coverage

The method of stem rule reduces the number of ARs under the condition of certain minimum support. Minimum support is still a major factor that influences the number of ARs. Setting minimum support to a large value can reduce the number of ARs, but in the meantime causes the problem of coverage as discussed in chapter 2. To reduce ARs without causing the problem of coverage, a concept called *coverage* is discussed formally in this chapter. First, the formal definition of coverage and minimum coverage are given. Then a query refinement space based on coverage is defined. Finally, an algorithm generating query refinement space is proposed. All concept of query, query evaluation, and query refinement can be described within the query refinement space.

4.1 Preliminaries

The definition of *coverage* and so on, are given in this section. Meanwhile, notations given in section ?? are used here as well.

Definition 4.1 (Coverage):

Let $N \subset 2^{\mathcal{K}}$, $D \subset \mathcal{D}$. We say that N covers D, or N is a *coverage* of D, iff

$$D \subseteq \bigcup_{q \in N} \sigma(q)$$

In a similar fashion, N is also called a *coverage* of N' ($\subseteq 2^{\mathcal{K}}$) iff

$$\bigcup_{q \in N'} \sigma(q) \subseteq \bigcup_{q \in N} \sigma(q)$$

For example, in Table 2.2, let $q = \{k_1\}$ be the original query, then documents $D(q) = \{d_1, d_2, d_4, d_6\}$ will be retrieved. When minimum support is set to 2 and maximum confidence is set to 0.8, the set of refinement candidates based on stem rule is $\{\{k_2\}, \{k_6\}\}$.

Let $D = \{d_1, d_2, d_4, d_6\}$, $N = \{\{k_2\}, \{k_6\}\}$. Then

$$D = \{d_1, d_2, d_4, d_6\} \not\subset \bigcup_{q \in N} \sigma(q) = \{d_2, d_3, d_4, d_5, d_6\}$$

Here, $\sigma(\{k_2\}) = \{d_2, d_3, d_5, d_6\}$. $\sigma(\{k_6\}) = \{d_2, d_4, d_6\}$.

Therefore, $\{\{k_2\}, \{k_6\}\}$ is not the coverage of $\sigma(\{k_1\}) = \{d_1, d_2, d_4, d_6\}$.

$N = \{\{k_2\}, \{k_3\}, \{k_6\}\}$ is the coverage of $\sigma(\{k_1\}) = \{d_1, d_2, d_4, d_6\}$

because

$$D = \{d_1, d_2, d_4, d_6\} \subset \bigcup_{q \in N} \sigma(q) = \{d_1, d_2, d_3, d_4, d_5, d_6\}.$$

$N = \{\{k_3\}, \{k_6\}\}$ is also the coverage of $\sigma(\{k_1\}) = \{d_1, d_2, d_4, d_6\}$. A set of documents D or a set of keywords N might have more than one coverage.

Definition 3.2 (Minimum Coverage):

A set of keywords, N, is called a minimum coverage of D iff D can not be covered by any $N' \subset N$. That is, removing any keyword from N will lead to $D \not\subset \cup_{q \in N} \sigma(q)$.

In the example above, $N = \{\{k_3\}, \{k_6\}\}$ is a minimum coverage of $\sigma(k_1) = \{d_1, d_2, d_4, d_6\}$ but $N = \{\{k_2\}, \{k_3\}, \{k_6\}\}$ is not.

4.2 Query Refinement Space

Based on the *coverage* relationships among sets of keywords in the keyword space, a directed cyclic graph $\mathcal{G}$ can be defined as follows.

$$\mathcal{G} = (\mathcal{N}, \mathcal{E}, w, \gamma, C)$$

where

$$\mathcal{N} = \{q \mid q \in \bigcup_{d \in \mathcal{D}} 2^{\rho(d)} \wedge (0 < w(q))\}$$

$$\mathcal{E} = \{(q_1, q_2) \mid q_1, q_2 \in \mathcal{N} \wedge q_1 \subset q_2 \wedge 0 < w(e) \leq \theta_c)\}$$

$$w : \mathcal{N} \cup \mathcal{E} \rightarrow R^+$$

θ_c is maximum confidence.

w is weight of nodes and edges. Let $q, q_1, q_2 \in \mathcal{N}, e = (q_1, q_2) \in \mathcal{E}$. w is defined by σ as the follows.

$w(q)) \stackrel{def}{=} |\sigma(q)|, \quad w(e) \stackrel{def}{=} \frac{|\sigma(q_2)|}{|\sigma(q_1)|} = \frac{w(q_2)}{w(q_1)}$

Here, supports and confidences are a weight associated with a node and an edge, denoted $w(q_i)$ and $w((q_i, q_j))$, respectively.

γ maps a rule from the pair of nodes. $\gamma(q_1, q_2)$ returns $q_1 \Rightarrow (q_2 - q_1)$.

$C : \mathcal{N} \rightarrow \mathcal{N} \times 2^{\mathcal{N}}$ is lower cover , which will be explained in Definition 4.3.

When $\gamma(q_1, q_2)$ is a rule, q_1 is called upper node of q_2 or q_2 is called lower node of q_1. If q_1 is an upper node of q_2, then $q_2 - q_1 \neq \emptyset$, and hence

$$\begin{aligned} \sigma(q_2) &= \bigcap_{k \in q_2} \sigma(\{k\}) \\ &= (\bigcap_{k \in q_1} \sigma(\{k\})) \cap (\bigcap_{k \in q_2 - q_1} \sigma(\{k\})) \\ &\subseteq \bigcap_{k \in q_1} \sigma(\{k\}) \\ &= \sigma(q_1) \end{aligned}$$

This means that any node covers any of its lower nodes.

According to this property, user can choose the lower nodes of his/her original query to reduce the size of result set. But in order to cover a upper node, the union of low nodes has to be computed.

The coverage of query q of Definition 4.2 is essentially divided

Table 4.1: A sample of document database

D	Keyword				
d_1	k_1	k_2	k_3		
d_2	k_1	k_2			
d_3		k_2	k_3	k_4	
d_4				k_4	k_5
d_5	k_1	k_2			k_5

into two parts. One is the coverage of document set $\hat{D}_q \stackrel{def}{=} \{d | d \in \mathcal{D} \wedge \rho(d) = q\}$ which only contains query q itself, the other is the coverage of remained document set $\sigma(q) - \hat{D}_q$

Definition 4.3 (Lower Coverage):

Let $K \subset \mathcal{K}$ and $C(q) = \cup_{p \in K}(q \cup p)$ be minimum coverage of $\sigma(q) - \hat{D}_q$.

$(\delta(q),\ C(q))$ is a lower coverage of q, where

$$\delta(q) = \begin{cases} \emptyset, & \hat{D}_q = \emptyset \\ \{q\}, & \text{otherwise} \end{cases}$$

For example , in $\mathcal{D}$ of Table 4.1 , let $q = \{k_1, k_2\}$ and $\sigma(q) = \{d_1, d_2, d_5\}$, $C(q) = \{\{k_3\}, \{k_5\}\}$ covers $\{d_1, d_5\}$, $\hat{D}_q = \{d_2\}$ can be only covered by q itself. Therefore, the lower coverage of q is $(\{q\}, \{\{k_3\}, \{k_5\}\})$.

4.3 Generation of Query Refinement Space

This section introduces an algorithm generating query refinement space $\mathcal{G}$ from document database $\mathcal{D}$.

First, the algorithm generating query refinement space by usual methods [Agra93] is summed up as follows.

1. Specified thresholds of minimum support and minimum confidence are decided
2. 1-Itemsets that have support exceeding minimum support are generated from keyword set $\mathcal{K}$. Then generate 2-Itemsets that have support exceeding minimum support are generated from each pair of 1-ItemSets Until $|\mathcal{K}|$-Itemsets, the process is same.
3. For each Itemset, all the rules that have confidence exceeding minimum confidence are generated by checking $p \Rightarrow q - p$ of each item q.
4. The rules that can be derived by other rules are removed.
5. For each node, lower coverage is computed and the nodes that are not in lower coverage are removed.

In this algorithm, combinations of ${}_{|\mathcal{K}|}C_n$ have to be checked to generate n-Itemset. In order to generate all Itemsets, combinations $2^{|\mathcal{K}|}$ need to be checked. Moreover, in the process of generating next rules, combinations 2^n of $q - p$ and p have to be checked for each

Table 4.2: Symbols for algorithm

Symbols	meaning	Symbols	meaning
$\mathcal{D}$	set of all documents	$\mathcal{K}$	set of all keywords
D	set of documents	K	set of keywords
q	query	σ	query evaluation
δ	coverage of $\hat{D}_q$	ρ	get keywords
$\hat{D}_q$	set of documents only contain q	$C(q)$	$\sigma(q)$-coverage of $\hat{D}_q$

item q of n-Itemsets and combinations of $2^n \times |n\text{-Itemset}|$ have to be checked for n-Itemsets. The calculation cost of all Itemsets may be computed as the following formula.

$$\sum_{t=1}^{2^{|\mathcal{K}|}} (2^t \times |t\text{-Itemset}|) = O(2^{2^{|\mathcal{K}|}}) \tag{4.1}$$

Moreover, this algorithm generates not only a large number intermediate rules but also causes the problem of coverage. In order to solve the problem above, we propose the following Algorithm that calculates only stem rule to form minimum coverage, which can reduce the number of rules. And minimum coverage will be used as the condition of generating ARs instead of minimum support.

Symbols used in the algorithm are listed in Table 4.2.

Algorithm (generate query refinement space($\mathcal{N}, \mathcal{E}$))

Input:$\mathcal{D}$, maximum confidence θ_c

Output:$\mathcal{N}$, $\mathcal{E}$

$\mathcal{N}_0 = \emptyset, \mathcal{N} = \emptyset$, $\mathcal{E} = \emptyset$

/* generate candidate notes */

forall $d \in \mathcal{D}$

begin

get $\rho(d)$

forall $q \subset \rho(d)$

$\mathcal{N}_0 = \mathcal{N}_0 \cup \{q\}$

end

/* generate $\mathcal{N}, \mathcal{E}$ */

1 **while** $\mathcal{N}_0 \neq \emptyset$ **do**

begin

2 select q that satisfies $|q| = \min_{p \in \mathcal{N}_0}\{|p|\}$

$D = \sigma(q) - \hat{D}_q$

/* generate lower coverage of q $(\delta(q), C(q))$*/

$(\delta(q), C(q)) = \mathrm{MC}(q, D, \theta_c)$

3 $\mathcal{N}_0 = \mathcal{N}_0 - \{q\} - \{q \cup p | p \in \delta(q)\}$

4 $\mathcal{N} = \mathcal{N} \cup \{q\} \cup \{q \cup p | p \in \delta(q)\}$

5 $\mathcal{E} = \mathcal{E} \cup \{(q,\ q \cup p) \mid p \in C(q) \cup \delta(q)\}$

end

function MC(q, D, θ_c)

begin

$C(q) = \emptyset,\ \delta(q) = \emptyset$

1 $q_0 = \{k \mid cnf(q \Rightarrow \{k\}) > 0\}$

2 **while** $D \neq \emptyset$ **do**

begin

3 select k_0 that satisfies $|D \cap \sigma(\{k_0\})| = \mathrm{Max}_{k \in q_0}\{|D \cap \sigma(\{k\})|\}$

.

4 **if** $cnf(q \Rightarrow \{k_0\}) \leq \theta_c$ **then**

$C(q) = C(q) \cup \{\{k_0\}\}$

else

begin

5 $q' = q \cup \{k_0\}$

6 $D' = D \cap \sigma(q') - \hat{D}_{q'}$

7 **if** $\hat{D}_{q'} \neq \emptyset$ **then**

$\delta(q) = \delta(q) \cup \{\{k_0\}\}$

/* expand q' by reflexive */

8 $(\delta(q'), C(q'))$ =MC(q', D', θ_c)

9 $C(q) = C(q) \cup C(q')$

10 $\delta(q) = \delta(q) \cup \delta(q')$

end

11 $q_0 = q_0 - \{k_0\}$

12 $D = D - \sigma(\{k_0\})$

/* in order to assure minimum, remove
lower coverage of k_0 from $C(q)$. */

13 $C(q) = C(q) - \{p \mid p \in C(q) \wedge p \supset k_0\}$

end

return $(\delta(q), C(q))$

end

According to the definition, nodes of $\mathcal{G}$ are requested to have confidence under maximum confidence. In order to assure coverage, this algorithm expands nodes that have confidence exceeding maximum confidence instead of removing them.

Because both $\mathcal{K}$ and $\mathcal{G}$ are huge, the algorithm to exclude unnecessary nodes and edges in $\mathcal{G}$, in order to reduce the time complexity.

Proposition:

The algorithm above stops and generates $\mathcal{G}$ that satisfies the condition of the definition.

Proof: According to row 2, 3 in the main procedure, repetition of row 1 on $\mathcal{N}_0$ will stop iff MC stops.

On the other word, considering row 1, 2 of MC, first, it can be proved that $q_0 \neq \emptyset \iff D \neq \emptyset$.

Therefore, according to row 11, repetition of row 2 will stop because q_0 is limit.

By row 5, 6 of main procedure, obviously $(q_1, q_2) \in \mathcal{E} \iff q_2 \in C(q_1)$. Hence, all the lower nodes of q forms the lower coverage of q. The lower coverage of a leaf node q is $(\{q\}, \emptyset)$.

In this algorithm, $\mathcal{N}_0$ is generated from $\rho(d)$ instead of $\mathcal{K}$. This means that the number of combinations that have to be checked is average values of number of keyword sets in a document because combinations that exist in $d \in \mathcal{D}$ need be only calculated.

The calculation cost of generating $\mathcal{N}$ is

$$\sum_{d \in \mathcal{D}} 2^{|\rho(d)|} \approx |\mathcal{D}| \times 2^{|\rho(d)|}.$$

The cost of generating $\mathcal{N}$ is put down because combinations of $\mathcal{N}_0$ become less.

According to $|\rho(d)| \ll |\mathcal{K}|$, it can seen that calculation cost is dramatically reduced by comparing the formula above with formula (4.1)

The algorithm is explained using the document database in Table 4.1 as follows.

The coverage of $\{k_2\}$ with $\theta_c = 0.5$ is computed as same as the coverage of $D = \sigma(\{k_2\}) = \{d_1, d_2, d_3, d_5\}$. Figure 4.1 shows the generation of the part of $\mathcal{G}$. Exactly, the generation of a lower coverage of $\{k_2\}$ (solid line in the figure) is shown.

All the upper-lower relationship of $\{k_2\}$ is shown in the left side of the figure. Lower coverage of $\{k_2\}$, generated by this algorithm, is shown in the right side of the figure. (k2k1k3 is the omission of

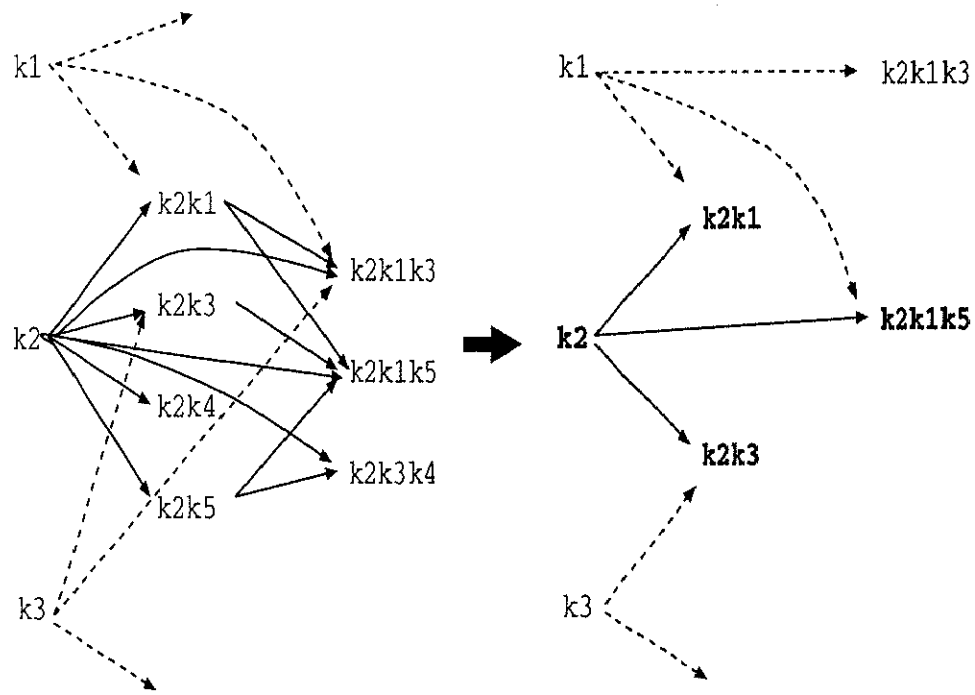

Figure 4.1: Example of the Algorithm

$\{k2, k1, k3\}$).

First, k_1 is chose from $q_0 = \{k_1, k_3, k_4, k_5\}$. Because the confidence of $\{k_2\} \Rightarrow \{k_1\}$ is $\frac{3}{4}$ which is large than $\theta_c = 0.5$, $\{k_2, k_1\}$ is expanded. That is,

$\mathrm{MC}(\{k_2, k_1\}, D \cap \sigma(\{k_2, k_1\}) - \{d|\rho(d) = \{k_1, k_2\}\}, 0.5)$

$=\mathrm{MC}(\{k_2, k_1\},\ \{d_1, d_5\},\ 0.5)$, is called.

The calling of MC results in $C(\{k_2, k_1\}) = \{\{k_3\}, \{k_5\}\}$ and D becomes $D - \sigma(\{k_1\})$, that is $\{d_3\}$.

Second, as k_3 in q_0 covers $\{d_3\}$, $\{k_3\}$ is added to $C(q) = \{\{k_1, k_3\}, \{k_1, k_5\}\}$ and lower node $\{k_1, k_3\}$ of $\{k_3\}$ is removed from $C(q)$. Now D becomes empty, so finally $C(\{k_2\}) = \{\{k_1, k_5\}, \{k_3\}\}$.

Then $\delta(\{k_2, k_1\}) = \{\{k_2, k_1\}\}$ is merged to $\delta(\{k_2\})$. With $\delta(\{k_2, k_1\})$ and $C(\{k_2\})$, the coverage of k_2 which is shown in the graph in the right side of figure 4.1, is generated according to row 4, 5 of the algorithm.

Some observations can be made on this algorithm. First, the number of nodes in $\mathcal{G}'$ will never exceed $\sum_{d \in \mathcal{D}} 2^{|\rho(d)|}$. This number is considerably small in comparison with that generated by those algorithms that will generate Itemsets from $2^{\mathcal{K}}$ where $\mathcal{K}$ is the set of keywords (see [Agra93]). In our experiment with a medium-size document base, the number of nodes in $\mathcal{G}'$ is only about several times of $O(|\mathcal{K}|)$. Second, from the definition, we know that $\mathcal{G}'$ contains no derivable nodes. Unnecessary edges have also been pruned.

In this chapter, minimum coverage is used as the condition of generating ARs instead of minimum support, which is different with stem rule. A query refinement space based on coverage still uses concept of maximum confidence and stepwise refinement. The algorithm generating query refinement space assures the coverage under the condition of maximum confidence.