Chapter 3

Stepwise Refinement by Stem Rule

As discussed in the previous chapter, one bottle neck of using AR for query refinement is that too much ARs are generated, because user will have to take the responsibility of browsing and choosing from a large number of refinement candidates. To address this problem, we introduced the concept of *stem rule* as mentioned in section 2.3. In this chapter we discuss formally the definition and generation of stem rules and the technique of *stepwise refinement* by using stem rules.

3.1 Preliminaries

In this section, we give the notations used in the following discussions in brief.

Let $\mathcal{D}$ and $\mathcal{K}$ be the universe set of documents and the universe

set of keywords, respectively.

Definition 3.1 (Operation ρ):

An operation ρ, which extracts keywords from a document $d \in \mathcal{D}$, is defined as

$$\rho(d) = \{k \mid (k \in \mathcal{K}) \wedge (k \text{ is a keyword included in } d)\}.$$

Furthermore, let $D \subset \mathcal{D}$, $\cup_{d \in D} \rho(d)$ is denoted by $\rho(D)$, and in particular $\rho(\mathcal{D}) = \mathcal{K}$.

Definition 3.2 (Evaluation σ):

Let q ($\subset \mathcal{K}$) be a set of keywords.

A conjunction evaluation $\sigma(q)$ will retrieve all the documents that contain all the given keywords in q, and is defined as

$$\sigma(q) = \{d \mid d \in \mathcal{D} \wedge q \subseteq \rho(d)\}.$$

Furthermore, let $Q \subseteq \mathcal{Q} = \cup_{d \in \mathcal{D}} 2^{\rho(d)}$. A disjunction evaluation σ retrieves a set of documents which is the union of all the results of each conjunction and is defined as

$$\sigma(Q) = \bigcup_{q \in Q} \sigma(q)$$

Here, a query is described as a disjunctive normal form of keywords. A conjunction form retrieves the documents each of which contains all the atom keywords in the conjunction. The query retrieves a set of documents which is the union of all the results of

each conjunction.

Definition 3.3 (Association Rule):

An AR is an implication of form $q_1 \Rightarrow q_2 - q_1$. Where $q_1 \subset q_2 \subseteq \mathcal{K}$.

The support and the confidence of a rule $q_1 \Rightarrow q_2 - q_1$, are calculated as follows.

$$Spt(q_1 \Rightarrow q_2 - q_1) = |\sigma(q_2)|$$
$$Cnf(q_1 \Rightarrow q_2 - q_1) = |\sigma(q_2)|/|\sigma(q_1)|$$

Here, q_1 is called original query. q_2 is called refined query. $q_2 - q_1$ is called refinement candidate.

In addition to the minimal confidence as used in [Fayy96, Han95, Sava95, Srik96], the maximum confidence is defined for the purposes of query refinement.

Definition 3.4 (Base Condition):

A rule, $q_1 \Rightarrow q_2 - q_1$, satisfies a base condition if and only if

$$MinSpt \leq Spt(q_1 \Rightarrow q_2 - q_1)$$
$$MaxCnf \geq Cnf(q_1 \Rightarrow q_2 - q_1)$$

Here, $MinSpt$, and $MaxCnf$ are minimum support and maximum confidence of the rule, respectively.

3.2 Stem Rules

In order to reduce the total number of rules that we need to store and to process, *stem rule* is introduced. Stem rules are the only rules we need to store in the rule base. All the other applicable association rules can be generated from the stem rules at run time instead of being generated from scratch.

Definition 3.5 (Stem Rule):

Given two rules r_1 and r_2. If r_1 satisfying the base condition implies that r_2 satisfies the same base condition, we say that rule r_2 is derived from rule r_1. A stem rule is a rule that can not be derived by any rules. In particular, for a stem rule, $q \Rightarrow p$, there exists $d \in \mathcal{D}$ such that $q \cup p \subset \rho(d)$.

This definition restricts the keywords of a rule to appear in at least one document. This restriction is natural because in keyword retrievals, if a set of keywords does not appear in any documents, then the retrievals by this set of keywords will get no hits.

Following properties are used for generating stem rules.

Property 3.1

a) $\forall q \subset \mathcal{K}, 0 \leq |\sigma(q)| \leq |\mathcal{D}|$;

b) $q \subset q' \subseteq \mathcal{K}$ implies $\sigma(q') \subseteq \sigma(q)$;

Association rule for query refinement is based on Property 3.1. User can refer to rule $q \Rightarrow q' - q$ to decide a new query q' that retrieves the result $\sigma(q')(\subseteq \sigma(q))$. That is, the size of result set will be reduced.

Property 3.2

a) Let $\Delta q \subseteq \mathcal{K}$. $Spt(q \Rightarrow p) \geq MinSpt$ implies $Spt(q - \Delta q \Rightarrow p \cup \Delta q) \geq MinSpt$;

b) $Cnf(q \Rightarrow p) \leq MaxCnf$ implies $Cnf(q - \Delta q \Rightarrow p \cup \Delta q) \leq MaxCnf$;

c) If $(q \Rightarrow p)$ satisfies the base conditions, then $(q - \Delta q \Rightarrow p \cup \Delta q)$ meets the base conditions.

Property 3.2 is used to exclude non-stem rules.

According to property 3.2, the stem rules are those that have only one keyword in the right hand side of the rules because if there are more than one keyword in the right side of the rule, this rule can be derived by moving keywords in the left side of the rule to it's right side.

In Table 2.2, the following list is all ARs including $q = \{k_1\}$ when MinSpt is set to 2 and MaxCnf is set to 0.8.

$k_1 \Rightarrow k_6$, Spt=3, Cnf=0.75

$k_1 \Rightarrow k_2$, Spt=2, Cnf=0.5

$k_1 \Rightarrow k_2 k_6$ Spt=2, Cnf=0.5

$k_1k_6 \Rightarrow k_2k$ Spt=2, Cnf=0.66

In this example, $k_1 \Rightarrow k_2k_6$ is not a stem rule because it can be derived by the rule $k_1k_6 \Rightarrow k_2$.

3.3 Stepwise Refinement

By using stem rule, a technique called stepwise refinement is introduced in this section. The idea is to show only the refinement candidates appear in stem rules instead of any rules.

Property 3.3:

Let $q_0 \subset q_1 \subset \cdots \subset q_i \subset q_{i+1} \subset \cdots, q_{m-1} \subset q_m \subset \mathcal{K}$ and $Spt(q_m) \geq MinSpt$.

Let a path be $L = (q_0, q_1), \cdots, (q_i, q_{i+1}), \cdots, (q_{m-1}, q_m)$ which are based on ARs $q_0 \Rightarrow q_1 - q_0, \cdots, q_i \Rightarrow q_{i+1} - q_i, \cdots, q_{m-1} \Rightarrow q_m - q_{m-1}$ and $Cnf((q_i, q_{i+1})) = Cnf(q_i \Rightarrow q_{i+1} - q_i)$ be the weight of path (q_i, q_{i+1}). Suppose that the AR of (q_0, q_m) exists, then

$$\begin{aligned} Cnf((q_0, q_m)) &= Cnf((q_0, q_1)) \times Cnf((q_1, q_2)) \\ &\quad \times \cdots \times Cnf((q_i, q_{i+1})) \times \cdots \times Cnf((q_{m-1}, q_m)) \end{aligned}$$

and

$$\begin{aligned} Cnf((q_0, q_m)) \leq \max(&Cnf((q_0, q_1)), Cnf((q_1, q_2)), \cdots, \\ &Cnf((q_i, q_{i+1})), \cdots, Cnf((q_{m-1}, q_m)) \end{aligned}$$

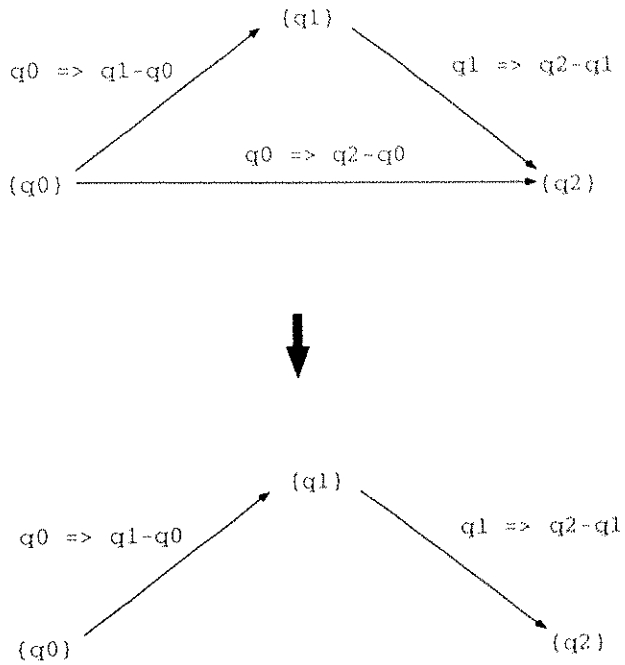

Figure 3.1: Stepwise

The following explains stepwise refinement based on the properties mentioned above.

Figure 3.1 shows the process of query refinement. Let q_0 be an original query. The refined query may be q_1 or q_2. The path (q_0, q_1), (q_1, q_2) and the path (q_0, q_2) are based on ARs $q_0 \Rightarrow q_1 - q_0$, $q_1 \Rightarrow q_2 - q1$ and $q_0 \Rightarrow q_2 - q_0$. According to Property 3.2, $q_0 \Rightarrow q_2 - q_0$ can be derived from $q_1 \Rightarrow q_2 - q_1$, and according to Property 3.3, the path (q_0, q_1), (q_1, q_2) can be used instead of the path (q_0, q_2).

Obviously, concentrating only on the paths corresponding to stem rules has two main advantages.

1. Reduce the time spent on displaying rules because non-stem

rules need not be derived.

2. Reduce the number of refinement candidates displayed so that users can browse easily.

3.4 Generation of Stem Rules

First, support of keywords in document is computed, which is a preparing of generating stem rules. Given a document database $\mathcal{D}$, Algorithm 3.1 retrieves all keywords contained and in the same time, calculates the support for each keyword.

Algorithm 3.1 (Counting Supports):
Input: a set of documents $\mathcal{D}$.
Output: a set of keywords and a support for each keyword in the set.

let $\mathcal{K} = \emptyset$;
foreach $d \in \mathcal{D}$ **do**
    calculate $\rho(d)$;
    foreach $k \in \rho(d)$ **do**
        if $k \in \mathcal{K}$ **then**
            $Spt(k) = Spt(k) + 1$;
        else
            $\mathcal{K} = \mathcal{K} \cup \{k\}$;

$Spt(k) = 1$;

endif

endforeach

endforeach

The following Algorithm 3.2 extracts a set of stem rules from a document database $\mathcal{D}$. First, a candidate set of rules, "Cand", will be generated in Algorithm 3.2. Second, a set of stem rules is generated from "Cand". An r is added to "Cand" again even if its confidence is smaller than $MaxCnf$ because r may derive other stem rules. In other words, r can not be excluded at this point of time because the inverses of Property 3.2 c) does not hold.

Algorithm 3.2 (Generating Stem Rules)

Input: A set of documents $\mathcal{D}$.

Output: A set of Stem Rules, R_s.

let Cand $= \emptyset$;

let $R_s = \emptyset$ be a set of Stem Rules;

foreach $d \in \mathcal{D}$ **do**

calculate $\rho(d)$;

foreach $X \subset \rho(d)$ **do**

if $MinSpt \leq Spt(X)$ **then**

Append $X \Rightarrow \emptyset$ to Cand;

endif

endforeach

endforeach

while Cand $\neq \emptyset$ **do**

remove a rule $X \Rightarrow Y$ from Cand;

foreach $k \in X$ **do**

let $r = (X - \{k\} \Rightarrow Y \cup \{k\})$

if $Cnf(r) \leq MaxCnf$ **and** r cannot be derived from R_s **then**

add r to R_s;

delete those rules which can be derived from r from Cand;

else

add r to Cand;

endif

endforeach

endwhile

In generating a candidate set of rules ("Cand"), unlike using Agrawal's algorithm in ([Agra93]) which computes 1-Itemsets, 2-Itemsets, ..., n-Itemsets from a set of items $\mathcal{K}$, we generate a set of stem rules from each document instead. At the step of computing n-Itemsets for each n, the complexity of using Agrawal's algorithm directly is $O({}_nC_{|\mathcal{K}|})$, because all combinations of items (keywords) $\mathcal{K}$ must be checked. However, on the other hand, the

time complexity of our algorithm is $O({}_nC_{\lfloor \rho(d) \rfloor})$, because we only need to check keywords in a document.

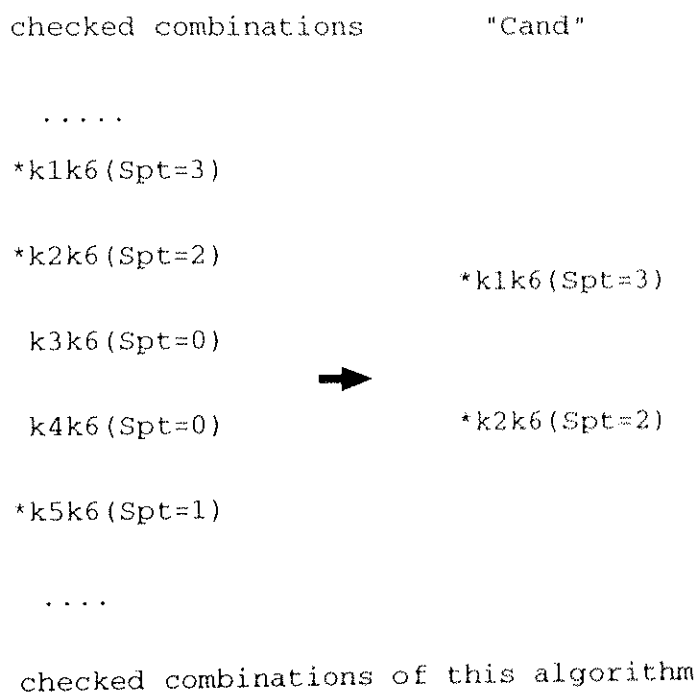

Figure 3.2: An Example of Algorithm

Figure 3.2 is an example of generating "Cand" from Table 2.2. This algorithm only checks the combinations having support exceeding 0. That is, the combinations exist in documents.

In this section, maximum confidence is proposed to be used instead of minimum confidence used in other researches about association rules. According to maximum confidence, refinement can-

didates have better effectiveness of screening. By using *stem rules* and *stepwise refinement*, the number of refinement candidates and the size of the rule base are reduced.