# Fast Computational Mining Technique for XML Query Answering Support

R.Brindhadevi, M.Tech
.J.Jabez, ME
Department of Information Technology, Professor of Information Technology,
Sathyabama University, TN. IndiaSathyabama University, TN.India

## ABSTRACT
The database research field has focused on the Extensible Mark-up Language (XML) because of its adaptable progressive nature which can use to represent to huge amount of data, likewise it doesn't have absolute and fixed schema, yet having possibly spasmodic and deficient structure. Quite hard undertaking to concentrate data from semi organized documents and is set to wind up more challenging as the measure of computerized data accessible on the Internet develops. Really, the data set returned as response to a query may be so enormous it is not possible pass on interpretable information, as documents are regularly so extensive. A methodology based on Tree- Based Association Rules (TARs), which furnish rough, intentionaldata about the structure and the contents of XML documents both, and additionally it might be saved in XML format. This mined information is utilized to give, a brief thought of both the structure and the content of the XML archive and snappy, inexact replies to queries at whatever point needed.

## Index Terms
Extensible mark-up Language (XML), query answering, data mining, intentionaldata, Tree-Based Association Rules.

## 1. INTRODUCTION
Data mining is generally used to concentrate intriguing learning from a lot of data saved in databases or data warehouses. This learning could be spoken to in numerous diverse routes, for example, bunches, choice trees, choice tenets and so forth. Around them, cooperation standards have been demonstrated compelling device to finding fascinating relations in huge measures of data. Throughout the later years, we have seen the sensational advancement of the extensible Mark-up Language (XML) as a significant standard for saving and trading information. The database research field has focused on XML as an expressive and adaptable progressive model suitable to speak to immense measures of data with no total and altered outline, and with a perhaps unpredictable and inadequate structure. In spite of its great development in ubiquity, XML is as of now needing proper systems to recover datasets accessible to easy clients. The first hails from the custom of information recovery [9], where most searches are performed on the literary substance of the archive; this implies that no preference is inferred from the semantics passed on by the report structure. With respect to Query-answering, since query dialects for semi organized data depend on the record structure to pass on its semantics, in place for query plan to be compelling clients requirement to know this structure ahead of time, which is regularly not the situation, actually, it is not required for a XML archive to have a characterized mapping: half of the reports on the web don't hold one [5]. The point when clients define questions without knowing the archive structure, they may neglect to recover information which was there, however under recover information which was there, yet under an alternate structure. This restriction is a critical problem, which finished not develop in the connection of social database administration frameworks. Regular, tragic conclusions of this circumstance are either the information overload problem, where an excessive amount of data are incorporated in the response in light of the fact that the set of keywords specified for the search catches an excess of implications, or the information deprivation problem, where either the utilization of in fitting keywords or the wrong definition of the query, avoid the client from accepting the right reply. As a result, when entering despite any precedent to the contrary a huge dataset, picking up some general information about its principle structural and semantic qualities helps examination on additional particular portions.

Uncovering repetitive examples inside XML (records) furnishes fantastic information about the report substance: Frequent examples are indeed deliberate data about the data held in the record itself, that is, they define the archive regarding a set of lands instead of by method of data. Rather than the point by point and exact data passed on by the data, this data is halfway and frequently inexact, however manufactured and concerns both the record structure and its substance. The thought of mining association rules[1],[2] to give outlined representations of XML records has been researched in numerous recommendations either by utilizing dialects, for example, XQuery, JQuery and so forth., and systems created in the XML connection or executing diagram or tree-based calculations. In this paper we present a proposal for mining and saving Tars (Tree-based Association Rules) as an intends to speak to deliberate information in local XML. Instinctively, a TAR speaks to deliberate learning as SB SH, where SB is the assemblage of tree and SH is head tree of the principle and SB is a sub tree of SH. The principle SBSH states that, if the tree SB shows up in a XML report D, it is likely that the "more extensive", tree SH likewise shows up in D. Graphically, we render the hubs of the assembly of a standard by method of black circles and the hubs of the head by empty circles.

Summarizing, TARs are extracted for two main purposes: 1) To get a concise idea-the gist-of both the structure and the content of an XML document, and 2) To use them for purposeful query answering, that is, allowing the client to query the extracted TARs rather than the original document

### 1.1 Goalsand Contributions
In this method for determining intentional knowledge from XML documents as TARs, and storingthese TARs as an alternative, synthetic dataset to be queried for furnishing quick and summarized answers. OurStrategy is described by the following key aspects:

1. It works directly on the XML documents, without converting the data into any intermediate format.

2. It searches for general association rules, without the need to force what should be heldin the Antecedent and consequent of the rule.

3. It stores an association rules in xml format.

4.It converts the queries on the original dataset into queries on the TARs set.

The aim of our proposal is to give an approach to utilize deliberate information as a substitute of the first ever record throughout querying and not to enhance the execution time of the queries over the first XML dataset.

Accordingly, the project contributions are:

An improved version of the TARs extraction algorithm found in which was based on Path Join. The form utilizes the better performing CMTreeMiner to mine frequent sub trees from XMLDocument.Approach validation by means of experimental results, acknowledgingboth the previous and the current Algorithm and showing the changes

Automatic user-query change into "equivalent" queries over the mined deliberate knowledge.

As a formal corroboration of the accuracy of the procedure, the verification that our intentional-answering Process is sound and complete up to a Frequency threshold.

## 1. 2.Structure Of The Paper

Our paper is composed as accompanies. Section 2 characterizes Tree-based Association Rules (Tars) and presents their utilization, while section 3 presents how these rules are concentrated from XML archives. Section 4 displays the fundamental fascinating provision of Tars that is their utilization to furnish intentional replies to questions. Section 5 depicts a model that actualizes our proposal. Section 6 reaches the inference and future work.

## 2. TREE ASSOCIATION RULES

Association rule is a significance of the structure X=>Y, where the rule body X and head Y are subsets of the situated I of items (I= I1, I2… In}) inside a set of transactions D and X∩Y=0 guideline X=>Y states that the transaction T that hold the things in X are liable to hold likewise the terms in Y. Association rules are described by two measures: the support, which measures the rate of transactions in D that hold both things X and Y (XUY); the confidence, which measures the rate of transactions in D holding the things X that likewise hold the things Y (support (XUY) /support(X)). In XML connection, both Dand I are accumulation of trees. In this work we expand the idea of association guideline presented in the setting of socialdatabases to adjust it to thevarious level nature of XML documents. Emulating the Info set assemblies, we speak to a XML archive as a tree N, E, r, l, c where N is the situated of hubs, r N is the base of the tree, E is the situated of edges, l : NL is the mark capacity which furnishes a proportional payback of hubs (with L is the area of all tags) and c : N C { } is the content capacity which furnishes a proportional payback of nodes(with C the dominion of all contents). We think about the element-just Info set content model [14], where XML non-terminal tags incorporate just different elements and/or attributes, while the content is restricted to terminal elements. We are intrigued by discovering connections around sub trees of XML documents. Therefore, since both text based content of leaf elements and qualities of attributes pass on "content", we don't recognize them. As a result, for the purpose of coherence, we don't report the edge name and the hub sort mark in the name in the figures. Attributes and elements are portrayed by empty circles, although the printed content of elements, or the quality of attributes, is accounted for under the friendly edge of the elements or attributes.

## 2.1. Fundamental Concepts

Given two trees T = (NT, ET, rT, lT, cT) and S = (NS, ES, rS, lS, cS), S is an induced sub tree of T if and only ifthere exists a mapping $\theta : NS \rightarrow NT$ such that for each node $ni \in NS$, $lT(ni) = lS(nj)$ and $cT(ni) = cS(nj)$, where$\theta(ni) = nj$,induced subtree of t and rS = rT. Given a tree $T = \langle NT, ET, rT, lT, cT \rangle$, a subtree of T, $t = \langle Nt, Et, rt, lt, ct \rangle$ and a client-fixed support threshold smin: (i) t is frequent if its support is greater or at least equal to smin; (ii) t is Maximum if it is frequent and none of its proper super trees is frequent; (iii) t is closed if none of its proper a super tree has support greater than that of t.

A Tree-based Association rule (TAR) is a tuple of the form $Tr = \langle SB, SH, sTr, cTr \rangle$, where $SB = \langle NB, EB, rB, lB, cB \rangle$ and $\langle NH, EH, rH, lH, cH \rangle$ are trees and sTr, and cTr are real numbers in the interim [0, 1] representing the Support and confidence of the rule respectively. A TAR describes the co-event of the two trees SB and SH in an XML document. For the sake of readability we shall often use the short notation $SB \Rightarrow SH$; SB is called the body or antecedent of Tr while SH is the head or consequent of the rule. Besides, SB is a subtree of SH with an extra property on the node labels; the set of tags of SB is contained in the set of tags of SH with the addition of the empty label "ϵ": $\ell$ SB (NSB) $\subseteq \ell$ SB (NSB) $\cup$ {ϵ}.The empty label is introduced because the body of a rule may contain nodes with unspecified tags, that is, blank nodes. In addition:

1) **A rooted TAR (RTAR)** is a TAR such a SB is rooted subtree of SH

2) **An extended TAR (ETAR**) is a TAR such that SB is an induced subtree of SH.

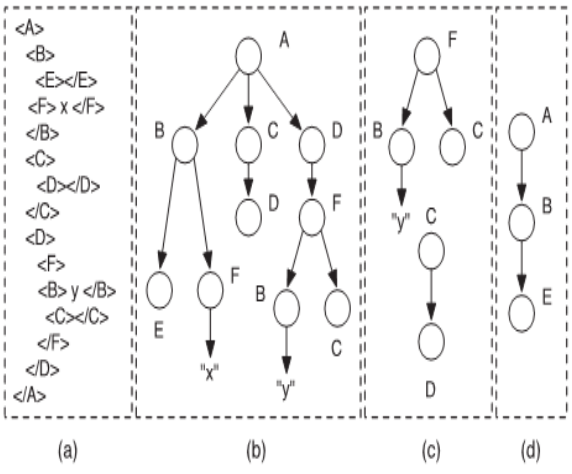



**Fig.1. TARs**

(a) An eg of XML document. (b) Tree-Based representation. (c) Two induced sub trees.(d) A rooted sub tree. Let count(S, D) indicate the number of occurrencesof a sub tree S in the tree D and cardinality (D) denote theNumber of nodes of D.

We formally define the support ofA TAR SB⇒ SH as count (SH, D)/cardinality (D) and its confidence as a count (SH, D)/count (SB, D)

Notice that TARs, in addition to association betweendata values, also provide information about the structure of successive parts of XML documents; thus they are more expressive than traditional association rules which onlyprovide frequent correlations of even values.

## 3. TAR EXTRACTION

TAR mining is a methodology made out of two steps:

1) Mining incessant sub trees, that is, sub trees with a support above a user defined threshold, from the XML document 2) Computing intriguing rules, that is, rules with a trust above a client characterized limit, from the continuous sub trees. Algorithm 1 exhibits our development to a non-specific regular subtree-mining calculation so as to process intriguing Tars. The inputs of Algorithm 1 are the XML record D, the threshold for the support of the incessant sub trees minsupp, and the threshold for the confident of the rule, minconf.

**Algorithm 1 Get Interesting-Rules (*D, minsupp, minconf*)**

1: *// frequent sub trees*

2: *FS* = Find Frequent subtrees (*D, minsupp*)

3: rule Set = ∅

4: for all *s* ∈ *FS* do

5: *// rules computed from s*

6: tempset = Compute-Rules(*s, minconf*)

7: *// all rules n*

8: ruleSet = ruleSet U tempSet

9: end for

10: return ruleSet

**Function 1 Compute-Rules(*s, minconf*)**

1: ruleSet = ∅ ; blackList = ∅

2: for all *cs*, subtrees of *s* do

3: if *cs* is not a subtree of any element in blackList then

4: *conf* = *supp*(*s*) / *supp* (*cs*)

5: if conf≥minconf then

6: newRule = *{cs, s, conf, supp*(*s*)}

7: ruleSet = ruleSet ∪ *{*newRule*}*

8: else

9: blackList = blackList ∪ *cs*

10: end if

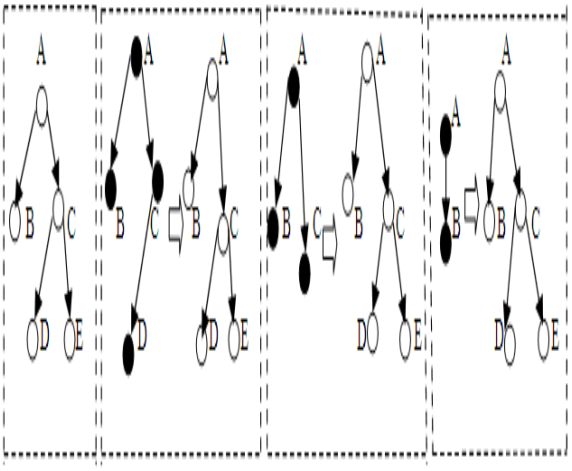
11: end if

12: end for

13: return ruleSet

Algorithm 1 finds regular sub trees and then hands each of them over to a capacity that registers all the conceivable rules. Contingent upon the amount of incessant sub trees and their cardinality, the measure of rules created by a naive Compute Rules function may be quite high. Given a subtree with n hubs, we could produce 2n-2 rules, making the algorithm exponential. This eruption occurs in the relational connection too, accordingly, based on comparable contemplations, it is conceivable to state the accompanying property that permits us to propose the upgraded adaptation of register rules demonstrated in function 2.

**Remark 1.** If the confidence of a rule SB⇒SH is below the established threshold minconf then the confidence ofevery other rule SBi⇒SHi, such that its body SBi is an induced subtree of the body SB, is no longer thanminconf. Consider Fig 2, which shows a frequent sub tree (Figure 2a) and three possible TARs mined from thetree; all the three rules have the same support k and confidence to be calculated. Let the support of the bodytree of rule (1) be *s*. Since the body trees of rules (2) and (3) are sub trees of the body tree of rule (1), theirSupport is at least *s*, and potentially higher. This means that the confidences of rules (2) and (3) are equal otherwise less than the confidence of rule (1).




**(a) (1) (2) (3)**

**Fig.2. Rule examples for Property**

1In Function 2 Tars are mined misusing Remark 1 byproducing first the principles with the most noteworthy number of hubs in the body tree. Think about two principles Tr1 and Tr2 whose body trees hold one and three nodes separately; assume both rules have confidence underneath the settled threshold. In the event that the algorithm recognizes standard Tr2 first and foremost, all runs whose figures are prompted subtrees of Tr2 will be disposed of when Tr2 is eliminated. Thusly, it is more helpful to first produce principle Tr2 and when all is said in done, to begin the mining methodology from the principles with a bigger figure. Utilizing this result, we can bring down the unpredictability of the calculation, however insufficient to make it perform superior to exponentially. Then again, perceive that the procedure of determining Tars from XMLarchives is just completed intermittently. Since intensional learning speaks to continuous data, to redesign it, it is attractive to perform such transform after huge measures of overhauls have been made on the first archive. In this manner, on account of stable records (that is, those that are infrequently upgraded) the calculation must be connected few times or once (for reports that don't change). When the mining procedure has completed and continuous Tars have been concentrated, they are archived in XML format.

This conclusion has been taken to allow the use of the same language for querying both the original dataset and the mined

rules. Each rule is saved inside a <rule> element, which contains three, characteristic for the ID, support and confidence of a rule. Follows the list of elements, one for each hub in the rule head. We misuse the fact that the body of the rule is a subtree of the head, and utilize a Boolean attribute in each node to indicate if it also belongs to the body. Every blank node is described by an element <blank>.

At last, the rules in the XML file are sorted on the amount of hubs of their predecessor; this is an essential characteristic that is utilized to improve the answering of queries holding a check operator. One of the (obvious) explanations behind utilizing TARs rather than the original document is that handling the archive. To exploit this, we acquaint records on TARs with further accelerate the right to gain entrance to mined trees - and all in all of purposeful query answering. As a rule, path indexes are proposed to rapidly answer queries that accompany some incessant path pattern, and are fabricated by indexing just those ways having greatly visit questions. We begin from an surrogate point of view: we need to furnish speedy, and frequently inexact, answers likewise to casual queries.

Given a set R of rules, the index connects, with every path p available in at least one rule of R, the references to rules that have p in SH. An index is an XML document containing a set of trees T1…..Tn such that each node n of each tree Ti contains a set of references to the rules containing in SH the path from the root of Ti to n. A TAR-index contains references both to iTARs, sTARs, and is constructed by Alg2.

**Algorithm 2 Create-Index** (D)

1: for all *Di* ∈ *D* do

2: for all *dj* ∈ *Di* with *j*∈ *{*2, 3 . . . *n}* do

3: references (root (*d*1)) = references (root (*d*1)) ∪ references (root (*dj)*)

4: sumChildren (*d*1, *dj)*

5: end for

6: end for

7: return D

**Function 2** sumChildren (*T*1, *T2*)

1: for all x ∈ children (root (*T*2)) do

2: if □ □ ∉ children (root (*T*1)) | c = x then

3: references (root(c)) = references (root(c)) □ □ references (root(x))

4: c = sumChildren(c, x)

5: else

6: addChild (root (*T*1), x)

7: end if

8: end for

9: return

# 4. INTENSIONAL ANSWERS

ITARs (deliberate Tree-based Association Rules) furnish an approximate intensional perspective of the content of an XML record, which is when all is said in done more succinct than the extensional one since it depicts the data in termsOf its properties, and since just the properties that are checked by a high number of items are concentrated. A Client query over the first dataset might be immediately changed into a query over the concentrated iTARs.

The response will be anintentional, on the grounds that, instead of giving the set of data fulfilling the query, the framework will reply with a set of properties that these data "frequently fulfilled", along with support and confidence. There are two major advantages:

i) querying iTARs requires less time than querying the original XML document;

ii) Approximate, intensional answers are in some cases more useful than the extensional ones (see theIntroduction). Not all queries lend themselves to being transformed into queries on iTARs; we list three classesof queries that can be transformed by maintaining the soundness; Additionally, we describe how such transformationcan be automatically done.

**Algorithm 3 Class1-Query** (*vF ,VW, CONN, vOB*)

1: *// the intentional query is empty*

2: IQ=ϵ

3: if *VW* ≠ ∅ then

4: *// get instance rule for paths with a constraint*

5: IQ = IQ • get iTARs (*vF, VW, CONN, false*)

6: else

7: *// structure rule for the path without constraint*

8: IQ = IQ • get sTARs (*vF)*

9: end if

10: *// order the results*

11: IQ = IQ • "for $r in $Rules/RuleOrder by $r/*vF* /*vOB*

return $r"

12: return IQ

**Function 3: get iTARs** (for, variables, connectives, count)

1: Q = ϵ

2: for all *VJ* ∈ variables do

3: if count = true then

4: *// for count queries match only in antecedent*

5: Q=Q•"let $RefI_j:=references A (for, *VJ)*"

6: else

7: *// for queries without include match both in antecedent and consequent*

8: Q=Q • "let $RefI_j:=references (for, *VJ)*"

9: end if

10: end for

11: Q=Q • "let $Rules: ="

12: for all *vj* ∈ variables, *j* ∈ *{*1 . . . *n}* do

13: Q=Q • "ruleset ($RefI_j) *connectivej"*

14: end for

15: return Q

**Function 4:** get sTARs (variable)

1: Q="let $RefS: =references (variable, "")

let $Rules: = ruleset ($RefS)"

2: return Q

**Syntax**
for variable in path

[Where condition [(and/or) condition]]

[order by element [asc|desc]] return variable

Class 2: count- queries: Used to number the amount of elements having a particular content. The query makes a set holding the elements which fulfil the conditions and then gives back the number of elements in such set. For instance, "Recover the amount of occurrences".

This class of queries is revamped utilizing Algorithm 4. The effect is a query qi that determines the iTARs, which fulfil the first query conditions, and gives back the support of the first rule, which has been found, partitioned by its confidence. Perceive that, since rules are requested as per the amount of hubs in their precursor, the first rule will be either the one, which fulfils all, and just the asked for conditions or its best rough guess (that is, a rule whose antecedent fulfils all the coveted conditions and holds the minimum number of nodes).

**Algorithm 4 Class2-Query (***vF, VW, CONN***)**

1: *// the intensional query is empty*

2: IQ =ϵ

3: *// get instance rule for path with a constraint*

4: IQ = IQ• get iTARs (*vF, VW, CONN, true*)

5: IQ = IQ • get count ()

6: IQ= IQ • "return $supp div $conf"

7: return IQ

Class 3: top-k queries: Used to select the best k responses satisfying a counting and amassing condition. The query counts the occurrences of each dissimilar value of a variable in a desired set; then orders the variables with respect to their occurrences and returns the most frequent k. For example, ""Recover those k most utilized sorts of nation".

This class of queries is rewritten using Alg 5. The result will be a query *qI* that for each distinct value of a variable finds the comparing sTARs and uses them to compute the number of occurrences of each value; ranks the values according to the computed count and returns all the rules associated with the first k stacked up values.

**Function 5:** get count **()**

1: Q ="let $supp:=$Rules/Rule[1]@support

let $conf: =$Rules/Rule [1] @confidence"

2: return Q

**Syntax**
let $set: = (class *1 query)*

Return count ($set)

**Algorithm 5 Class3-Query (***vDV, vF, VW, CONN***)**

1: IQ =ϵ *// the intensional query is empty*

2: IQ = IQ • get sTARs (*vDV)// get instance rules for paths with a constraint*

3: IQ = IQ • "for $v in distinct-values ($Rules/*vDV)*"

4: IQ = IQ • get iTARs (*vF, VW, CONN, true*)

5: IQ = IQ • get count ()

6: IQ = IQ • "order by $supp div $conf descending

return $Rules) [position () <=k]"

7: return IQ

**Syntax**
(for *variable* in distinct-values (*path*)

Let $set: = (class *1 query)*

Order by count ($set) desc

Return *variable) [*position () <= k]

Perceive that, in all classes of queries, conditions might be forced on the descendants of the element that is returned and not on its ancestors.

That is, a query holding conditions on the contents of an element should be as portrayed in (where x is the element returned by the query).

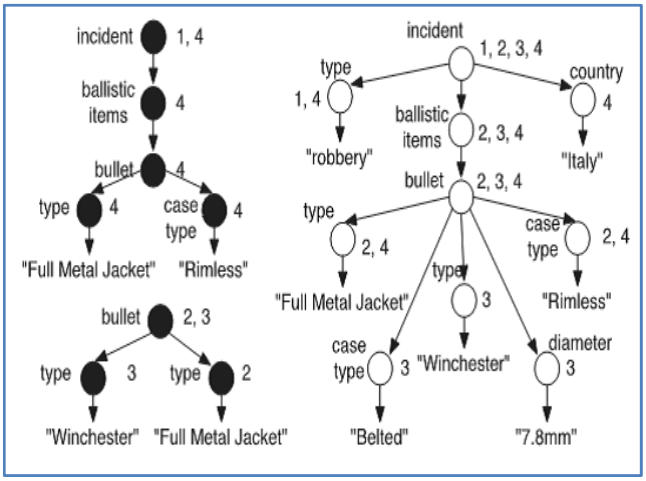



**Fig.3 A Graphical Representation**

<index>

<antecedent>

<bullet><ref>2</ref><ref>3</ref>

<type> Winchester

<ref>3</ref>

</type></bullet>

</antecedent>

<Consequent>

<incident>

<ref>1</ref><ref>2</ref>

<ref>3</ref><ref>4</ref>

<type> robbery

<ref>1</ref><ref>4</ref>

</type>

<country> Italy

<ref>4</ref>

</country>

</incident>

</consequent>

</index>

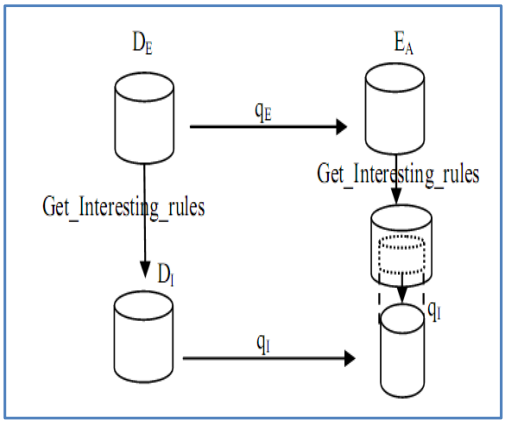



**Fig.4 Intentional query answering**

## 5. EXPERIMENTAL RESULTS

### 5.1. TheTree Ruler Prototype

Tree Ruler is a prototype tool that incorporates all the functionalities proposed in our methodology. Given an XMLDocument, the tool is able to extract intentional knowledge and permits the user to form traditional queries asWell as queries over the intentional knowledge.

Figure 1 shows the design of the tool. Specially, given the XML document, it is possible to extract Tree-based rules and the corresponding indexed.

The user figures XQuery articulations on the data and these queries are immediately interpreted to be executed on the deliberate information. The response is given regarding the set of Tree-based rules which re and so forth the search criteria. It is com postured by some tabs for performing distinctive tasks. Specifically, there are three tabs:

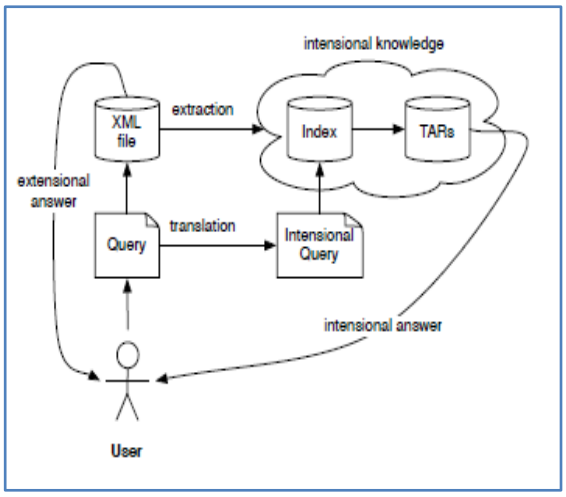



**Fig. 5: Tree Ruler Architecture**

- Get the Gist allows intentional information extraction from an XML file, given support, confidence and the files where the extracted TARs and their index are to be stored.
- Get the Idea allows showing intentional information as well as the original document, to give users the possibility to compare the two class of information.
- Get the Answers allows querying anintentional knowledge and the original XML document. Users have to compose an extensional query.

## 6. CONCLUSIONS AND FUTURE WORK

The main objectives we have attained in this work are:

1) Mine all continuous association rules without forcing any apriority restriction on the structure and the contentof the rules.

2) Store extracted information in XML format.

3) Use mined knowledge to gain information about the original datasets.

Casual clients can search the data by a keyword without any data base knowledge from XML storage media.

We have not examined the updatability of both the document storing TARs and their index's an progressing work, we are studying how to incrementally update mined TARs when the original XMLDataset change and how to further upgrade our mining efficient algorithm; additionally, for the moment we dealwith a (substantial) fragment of XQuery; we would like to find the exact section of XQuery, which lends itselfto interpretation into an intentional queries.

## 7. REFERENCES

[1] Agrawal.R and Srikant.R, "Fast Algorithms for Mining Association Rules in Large Databases," 2004, Proc. 20th Int'l Conf. Very Large Data Bases, pp. 478-499.

[2] Baralis.E, Garza.P, Quintarelli, and Tanca.L, "Answering XML Queries by Means of Data Summaries," vol .25, 2007 ACM Trans. Information Systems, p.no. 3, p. 10.

[3] Barbosa.D, Mignet.L, and Veltri.P, "Studying the XML Web: Gathering Statistics from an XML Sample," World Wide Web, vol. 8, no. 4, 2005, pp. 413-438.

[4] Braga.D,.Campi.A, Ceri.S, Klemettinen.M, and Lanzi.P, "Discovering Interesting Information in XML Data with Association Rules," 2003,Proc. ACM Symp. Applied Computing, pp. 450-454.

[5] Chi.Y,Yang.Y, Xia.Y, and Muntz.R.R, "CMTreeMiner: Mining both Closed and Maximal Frequent Subtrees," 2004, Proc. Eighth Pacific- Asia Conf. Knowledge Discovery and Data Mining, pp. 63-73.

[6] Evfimievski.A, Srikant.R, Agrawal.R, and Gehrke.J, "PrivacyPreserving Mining of Association Rules,"2012, Proc. Eighth ACM Int'l Conf. Knowledge Discovery and Data Mining, pp. 217-228.

[7] Gasparini.S and Quintarelli.E, "Intensional Query Answering to XQuery Expressions," 2005, Proc. 16th Int'l Conf. Database and Expert Systems Applications, pp. 544-553.

[8] Mazuran.M, Quintarelli.E, and Tanca.L, "Mining Tree-Based Association Rules from XML Documents," technical report, 2009, Politecnicodi Milano, http://home.dei.polimi.it/quintare/Papers/MQT09-RR.pdf,

[9] Paik.J, Youn.H.Y and Kim.U.M, "A New Method for Mining Association Rules from a Collection of XML Documents,"2005, Proc. Int'l Conf. Computational Science and Its Applications, pp. 936-945.

[10] Termier.A, Rousset.M, and Sebag.M, "Dryade: A New Approach for Discovering Closed Frequent Trees in Heterogeneous Tree Databases,"2004, Proc. IEEE Fourth Int'l Conf. Data Mining, pp. 543-546

[11] World Wide Web Consortium, XML Schema, http://www.w3c.org/TR/xml schema,2001.www.w3C.org/xml-info set/, 2001.

[12] W3C XML Schema,2001. http://www.w3C.org/TR/xmlschema-1/.

[13] W3C. XML information Set, 2001. http://www.w3C.org/xml-infoset/.

[14] W3C. XQuery 1.0: An xml query language, 2007. http://www.w3C.org/TR/xquery.

[15] Wang.K. and Liu. Discovering typical structures of documents: a roadmap approach. In *Proc. of the 21st Int. Conf. on Research.*