P- 2575

A RELATION-BASED SEARCH ENGINE IN SEMANTIC WEB

A PROJECT REPORT

Submitted by

D.ARUNKUMAR	71205205006
V.R.SURENDRAN	71205205058
V.THIRUMALAIVELU	71205205059

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

KUMARAGURU COLLEGE OF TECHNOLOGY, COIMBATORE

ANNA UNIVERSITY : CHENNAI 600 025

APRIL 2009

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report "**A RELATION BASED SEARCH ENGINE IN SEMANTIC WEB**" is the bonafide work of **D.ARUNKUMAR, V.R.SURENDRAN, V.THIRUMALAIVELU** who carried out the project work under my supervision.

SIGNATURE

Dr.S.Thangasamy ,B.E(Hons).,Ph.D

DEAN

Department of
Computer Science and Engineering,
Kumaraguru College Of
Technology ,
Coimbatore-641006.

SIGNATURE

Mr.K.R.Baskaran, B.E., M.S.

SUPERVISOR

Department of
Information Technology,
Kumaraguru College Of
Technology ,
Coimbatore-641006.

The candidates with University Register Nos. 7**1205205006, 71205205058 & 71205205059** examined by us in the project viva-voce examination held on

24/4/09

INTERNAL EXAMINER

27/4/09

EXTERNAL EXAMINER

DECLARATION

We,

D.ARUNKUMAR	**Reg.No: 71205205006**
V.R.SURENDRAN	**Reg.No: 71205205058**
V.THIRUMALAIVELU	**Reg.No: 71205205059**

hereby declare that the project entitled **"A RELATION BASED SEARCH ENGINE IN SEMANTIC WEB"**, submitted in partial fulfillment to Anna University as the project work of Bachelor of Technology (Information Technology) degree, is a record of original work done by us under the supervision and guidance of Department of Information Technology, Kumaraguru College of Technology, Coimbatore.

Place: Coimbatore

Date:

[D.Arun Kumar] [V.R.Surendran] [V.Thirumalai Velu]

Project Guided by,

[**L.S Jayashree M.E., Ph.D**] [**K.R.Baskaran., B.E,M.S**]

ACKNOWLEDGEMENT

We express our sincere thanks to our Chairman **Padmabhushaan Arutselvar Dr.N.Mahalingam B.Sc., F.I.E.,** Vice Chairman **Dr.K. Arumugam B.E., M.S., M.I.E.,** Correspondent **Shri.M.Balasubramaniam and** Joint Correspondent, **Dr.A.Selvakumar** for all their support and ray of strengthening hope extended. We are immensely grateful to our Vice Principal, **Prof.R.Annamalai** for his invaluable support to the outcome of this project.

We are deeply obliged to **Dr.S.Thangasamy, Dean**, Department of Computer Science and Engineering for his valuable guidance and useful suggestions during the course of this project.

We also extend our heartfelt thanks to our project co-ordinator **Mrs.L.S.Jayashree, Associate Professor.,** Department of Information Technology for providing us her support which really helped us.

We are indebted to our project guide **Mr.K.R.Baskaran, Assistant Professor** .,Department of Information Technology for his helpful guidance and valuable support given to us throughout this project.

We thank the teaching and non-teaching staffs of our department for providing us the technical support during the course of this project. We also thank all of our friends who helped us to complete this project successfully.

TABLE OF CONTENTS

ABSTRACT

ABSTRACT

With the development of the Web, an information "Big Bang" has taken place on the Internet. Search engines have become one of the most helpful tools for obtaining useful information from the Internet. However, instead of caring about the semantics of information, the machine on the current web cares about the location and display of information only. Because of this shortcoming of the current search techniques the search results by even the most popular search engines cannot produce satisfactory results. The development of the next generation web, Semantic web, will turn the situation around completely. This project proposes a prototype relation-based search engine, "OntoLook," which has been implemented in a virtual Semantic web environment and also present the architecture of the Semantic web and analyze the key algorithm.

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVATIONS

OWL	-	**Web Ontology Language**
RDF	-	**Resource Description Framework**
HTML	-	**Hyper Text Markup Language**
SQL	-	**Structured Query Language**

INTRODUCTION

1.INTRODUCTION

1.1 GENERAL

DATA MINING:

Data mining refers to extracting or "mining" knowledge from large amounts of data. The knowledge discovery process consists of an interactive sequence of the following steps.

1.Data cleaning

2. Data integration

3. Data selection

4. Data transformation

5. Data mining.

6. pattern evaluation

7. knowledge representation.

Today Data mining has attracted a great deal of attention in the information industry. The information and knowledge gained can be used for applications ranging from market analysis, fraud detection and customer retention, production control.

1.1.1. EXISTING SYSTEM:

In the existing system, the popular algorithm used in the search engines are

- The pagerank algorithm and
- HITs algorithm

In this approach, we directly query semantic information from ontology using some ontology query language. Because, in itself, Semantic web involves embedding semantics info into Web pages in the form of an RDF triple, the two mark languages, X(HTML) and RDF[3], appear in the same Web page. So, the compatibility of query languages becomes one of

the core principles. When designing a query language along with the development of the Web. There are three core principles in it. The first one is what we mentioned above: The same query language should provide convenient and efficient access to any kind of data expected to be found on Semantic Web. That is to say, the query language on Semantic Web cannot only query Web pages formed by X(HTML), but must also query the semantic description of the Web data formed by RDF[3]. Certainly, if the querying language addresses the semantic description formed by RDF, it can address the ontology formed by OWL[4] because OWL is a subset of RDF.

The second principle is that the query language should be based upon the principles of referential transparency and answer-closedness. Because of the decentralized and heterogeneous nature of the Web, the third principle is very important: It requires query languages that allow queries and answers to be incomplete. In the second approach, we translate ontology to some mature data store manner and the design of translation system becomes the key. Corcho and Gomez-Perez propose a Layered Model of the translation system. It divides the translation system to four layers: lexical, syntax, semantic, and pragmatic. The major advantage is easy construction of the translation system and convenient maintenance and reuse.

1.1.2 PROPOSED SYSTEM:

In our proposed system, each keyword specified by the user is associated with a relation from a pre-defined list of domains. There is nothing that can exist independently in the boundless universe. Everything is related to other things in various manners. When one tries to comprehend an entity, he comprehends it from the way it relates to other entities. In Semantic Web [2],[5], the semantics information is

presented by the relation with others and is recorded by RDF . Then, the relation is interpreted by OWL(Web Ontology Language) [4]. This enhances the search accuracy and provides a refined list of search results which match the keywords exactly.

Our proposed comprises of 5 modules,

- Construction of Microsemantic Web
- Creation of ontology RDBMS
- Architecture of relation-based search engine "OntoLook."
- Implemenatation of Cut arc algorithm.
- Performance Analysis

1.2 PROBLEM DESCRIPTION:

The problem is that when one enters the Keyword to the Search Engine it will display the results of the web in all the pages which contains the given keyword. It surprises users because they do input the right keywords and search engines do return pages involving these keywords, and, yet, the majority of the results are useless. In this type of Search the relation between the keywords which one tries to get is lost. This is main drawback of the existing system.

Relations lost—this is the key of the whole problem!

LITERATURE REVIEW

2.LITERATURE REVIEW

2.1 SEMANTIC WEB

The **Semantic Web** [2], is an evolving extension of the World Wide Web in which the semantics of information and services on the web is defined, making it possible for the web to understand and satisfy the requests of people and machines to use the web content. It derives from World Wide Web Consortium director Sir Tim Berners-Lee's vision of the Web as a universal medium for data, information, and knowledge exchange.

At its core, the semantic web comprises a set of design principles, collaborative working groups, and a variety of enabling technologies. Some elements of the semantic web are expressed as prospective future possibilities that are yet to be implemented or realized. Other elements of the semantic web are expressed in formal specifications. Some of these include Resource Description Framework (RDF), a variety of data interchange formats (e.g. RDF/XML, N3, Turtle, N-Triples), and notations such as RDF Schema (RDFS)[3], and the Web Ontology Language (OWL)[4], all of which are intended to provide a formal description of concepts, terms, and relationships within a given knowledge domain.

2.1.1 SEMANTIC WEB SOLUTIONS

The Semantic Web takes the solution further. It involves publishing in languages specifically designed for data: Resource Description Framework (RDF), Web Ontology Language (OWL), and Extensible Markup Language (XML).

HTML describes documents and the links between them. RDF, OWL, and XML, by contrast, can describe arbitrary things such as people, meetings, or airplane parts. Tim Berners-Lee calls the resulting network of Linked Data the Giant Global Graph, in contrast to the HTML-based World Wide Web.

These technologies are combined in order to provide descriptions that supplement or replace the content of Web documents. Thus, content may manifest as descriptive data stored in Web-accessible databases, The machine-readable descriptions enable content managers to add meaning to the content, i.e. to describe the structure of the knowledge we have about that content. In this way, a machine can process knowledge itself, instead of text, using processes similar to human deductive reasoning and inference, thereby obtaining more meaningful results and facilitating automated information gathering and research by computers.

2.1.2 COMPONENTS:

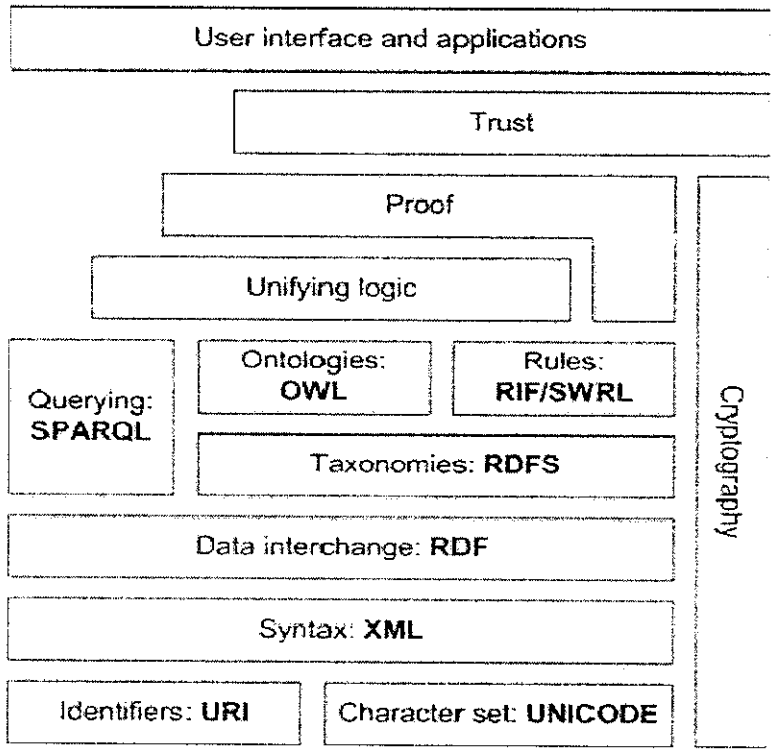


Fig 2.1 *The Semantic Web Stack*

The semantic web comprises the standards and tools of XML, XML Schema, RDF, RDF Schema and OWL that are organised in the Semantic Web Stack. The OWL Web Ontology language Overview describes the function and relationship of each of these components of the semantic web:

- XML provides an elemental syntax for content structure within documents, yet associates no semantics with the meaning of the content contained within.
- XML Schema is a language for providing and restricting the structure and content of elements contained within XML documents.
- RDF is a simple language for expressing data models, which refer to objects ("resources") and their relationships. An RDF-based model can be represented in XML syntax.
- RDF Schema is a vocabulary for describing properties and classes of RDF-based resources, with semantics for generalized-hierarchies of such properties and classes.
- OWL adds more vocabulary for describing properties and classes: among others, relations between classes (e.g. disjointness), cardinality (e.g. "exactly one"), equality, richer typing of properties, characteristics of properties (e.g. symmetry), and enumerated classes.

2.2 ONTOLOGY (INFORMATION SCIENCE):

An **ontology** [5], in computer science and information science is a formal representation of a set of concepts within a domain and the relationships between those concepts. It is used to reason about the properties of that domain, and may be used to define the domain.

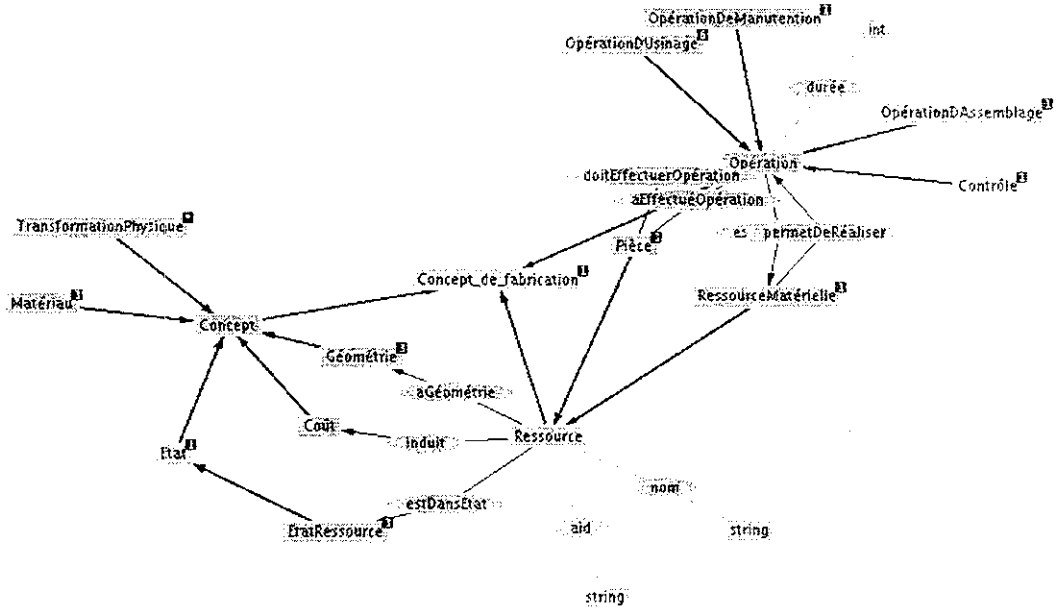


Fig2.2 Example of a ontology visualized: the Mason-ontology.

In theory, an ontology is a "formal, explicit specification of a shared conceptualization". An ontology provides a shared vocabulary, which can be used to model a domain – that is, the type of objects and/or concepts that exist, and their properties and relations.

Ontologies are used in artificial intelligence, the Semantic Web, software engineering, biomedical informatics, library science, and information architecture as a form of knowledge representation about the world or some part of it.

2.2.1 OVERVIEW

The term *ontology* has its origin in philosophy, and has been applied in many different ways. The core meaning within computer science is a model for describing the world that consists of a set of types, properties, and relationship types. Exactly what is provided around this varies, but this is the

essentials of an ontology. There is also generally an expectation that there be a close resemblance between the real world and the features of the model in an ontology.

What ontology has in common in both computer science and in philosophy is the representation of entities, ideas, and events, along with their properties and relations, according to a system of categories. In both fields, one finds considerable work on problems of ontological relativity (e.g., Quine and Kripke in philosophy, Sowa and Guarino in computer science) and debates concerning whether a normative ontology is viable (e.g., debates over foundationalism in philosophy, debates over the Cyc project in AI). Differences between the two are largely matters of focus. Philosophers are less concerned with establishing fixed, controlled vocabularies than are researchers in computer science, while computer scientists are less involved in discussions of first principles (such as debating whether there are such things as fixed essences, or whether entities must be ontologically more primary than processes).

2.2.2 ONTOLOGY COMPONENTS

Contemporary ontologies share many structural similarities, regardless of the language in which they are expressed. As mentioned above, most ontologies describe individuals (instances), classes (concepts), attributes, and relations. In this section each of these components is discussed in turn.

Common components of ontologies include:

- Individuals: instances or objects (the basic or "ground level" objects)
- Classes: sets, collections, concepts, types of objects, or kinds of things.

- Attributes: aspects, properties, features, characteristics, or parameters that objects (and classes) can have
- Relations: ways in which classes and individuals can be related to one another
- Function terms: complex structures formed from certain relations that can be used in place of an individual term in a statement
- Restrictions: formally stated descriptions of what must be true in order for some assertion to be accepted as input
- Rules: statements in the form of an if-then (antecedent-consequent) sentence that describe the logical inferences that can be drawn from an assertion in a particular form
- Axioms: assertions (including rules) in a logical form that together comprise the overall theory that the ontology describes in its domain of application. This definition differs from that of "axioms" in generative grammar and formal logic. In these disciplines, axioms include only statements asserted as *a priori* knowledge. As used here, "axioms" also include the theory derived from axiomatic statements.
- Events: the changing of attributes or relations

Ontologies are commonly encoded using ontology languages

2.3 PROTEGE

Protégé[7], is one of the most widely use ontology Editors with currently about 10,000 registered users. Its extensible open-source Platform supports several ontology le formats including CLIPS (Protégé's native format),various XML dialects,

databases, DAML +OILand RDF(S).Very recently ,storage plug-ins for the Unied Modeling Language(UML)and the Web Ontology Language(OWL)have been added .Both plugins are not complete yet And will evolve during the following months. This document reports on a simple experiment with the UML and OWL[4] Plugins. We wanted to test whether Protégé can convert a given ontology into these formats and to get an idea of which information are getting lost during conversion. Our starting point is the Travel Ontology developed by Natasha F. Noyas described in her contribution to the previous EON workshop. A screenshot of this ontology(displayed in Protégé)is shown in figure ??. The experiment was performed using the most recental phare lease of Protégé 2.0(build42).Older versions(startingwithversion1.8)would expose the same Behavior for the UML conversion .However ,these versions do not support the OWL Plugin.

2.4 HARDWARE SPECIFICATIONS (MINIMUM REQUIREMENTS):

Processor	:	Intel Pentium IV 2.2 GHZ
Hard Disk Drive	:	20 GB
RAM	:	256 MB RAM minimum

2.5 SOFTWARE SPECIFICATIONS (MINIMUM REQUIREMENTS):

Operating System	:	Windows 2000/XP
Software		
Relation Tree	:	Protégé 2000
Front End	:	MS Visual Studio 2005
Back End	:	SQL Server 2005.

DETAILS OF THE METHODOLOGY

3. DETAILS OF METHODOLOGY INVOLVED:

3.1 Construction of Microsemantic Web :

To construct a Microsemantic Web[2], environment. First, many Web pages are downloaded, and then embedded semantic annotation into them. The first segment, that is, the content between label <**ontopath**> and </**ontopath**>, indicates the location that the ontology of the current Web page belongs to. The ontology will interpret the metadata in semantic annotation. We use the "travel" edited by prote´ge´ as our ontology in our Microsemantic Web. The second segment, that is, content between label <**rdf_description**> and </**rdf_description**>,[3], annotates the main content of the current Web semantically. The three level data model namely Web page, semantic annotation, and ontology, has been constructed and resulting in construction of the Microsemantic Web.

3.2Creation of Ontology RDBMS

We translate ontology to some mature data store manner i.e., in form tables to form RDBMS and the design of translation system becomes the key. It divides the translation system to four layers: lexical, syntax, semantic, and pragmatic. The major advantage is easy

construction of the translation system and convenient maintenance and reuse. Since there is no mature querying language and RDBMS is used extensively. We use the second approach to obtain the semantic information: mapping ontology to RDBMS first, and then querying the database.

3.3 Architecture of Relation-Based Search Engine "Ontolook."

"OntoLook," constructed in Semantic Web, can exclude the keywords-isolated pages from the result set. Different from the traditional keyword-based search engines, "Onto-Look" is a relation-based search engine. When "OntoLook" processes the keywords, not only are the keywords processed, but so is the relationship between the entities offered by the architecture of Semantic Web. A page will be returned to users only when it includes the relationship between keywords; and those pages with the keywords only and without the relationship are discarded.

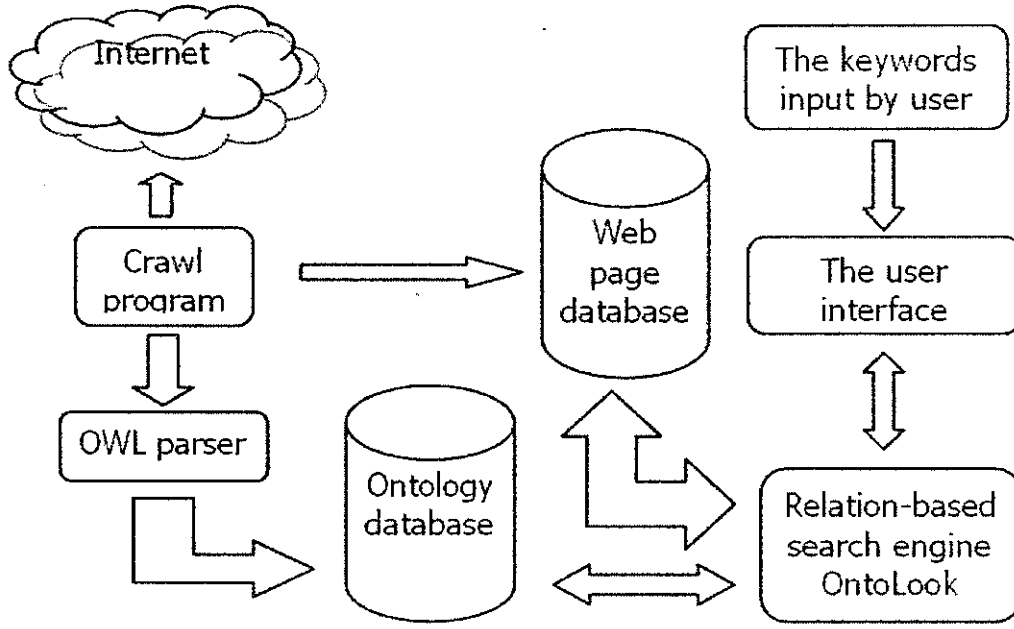


Fig 3.1 Ontolook System Architecture

3.3.1 Workflow Inside the Ontolook

The workflow of the architecture involves the relations between their keywords and their subgraph to evaluate a domain and their relations.

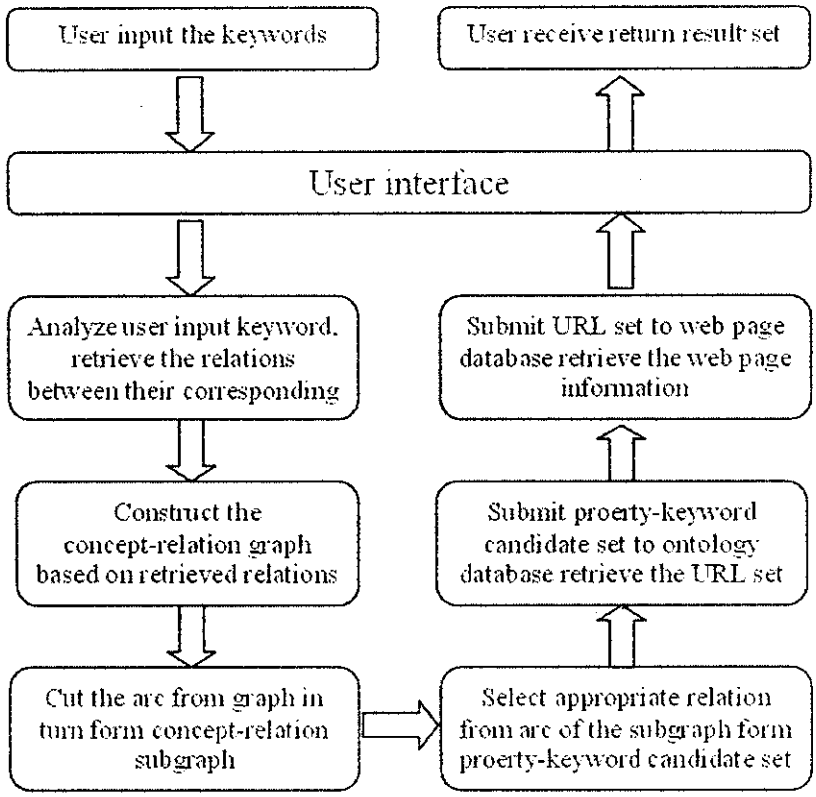


Fig 3.2 Workflow Inside the Ontolook

3.4 Implementation of Cut Arc Algorithm

We implement the Cut arc algorithm[1], to retrieve the exact webpages with matched keyword's relations from the Concept-Relation Graph formed. To cut arcs from graph G is to find and label the arc to be cut in the arc set R. From the definition of power set, we can determine that the time complexity of the cut arc algorithm is **O(2n)**. It seems to be ill-fitted in practice, but, fortunately, few people will submit large numbers of keywords to search engines. Form the Property-Keyword Pair Candidate Set. In subgraph **Gp**, we can take

out, in turn, an item from the property-keyword pair of each arc **Rij** to form property-keyword pair candidate set **CRKSetp**. Sending an item in CRKSetp in turn, and intersecting these result sets, we can obtain the Web page set users need, because it covers all of the relations among the keywords user input.

3.5.Performance Analysis

We evaluate the performance of the proposed Graph-Relation Based Cut arc algorithm based on the RDBMS ontology with that of the Non Semantic Web pages search. The quality and the quantity of the search result set is studied and evaluated. The relations based system and non relation based system are studied by giving many user search queries as the input to the system and the respective result sets are compared in detailed .But if the number of keywords is high ,then the time complexity also raises .so control it we can adding threshold limit to the sub graphs.

$$CRKSet_p \cdot \prod_{\substack{i=1 \\ j=1 \\ i>j}}^{n} |R_{ij}|$$

This comprises the Property-Keyword Candidate Set of the Sub graph .The below table deals the time complexity for the System performance.

Numbers of keywords or concepts	Numbers of relations among keywords	Numbers of subgraph be processed		Total numbers of property-relation Processed		CPU time millisecond	
		No threshold	add threshold When dealing	No threshold	add threshold When dealing	No threshold	add threshold When dealing
1	0	0	0	0	0	0	0
2	1	2	2	3	3	36.458	36.458
3	2	4	4	15	15	78.125	57.292
4	4	16	15	383	248	187.5	156.25
5	7	128	73	38399	2798	3125	500

Table 3.1 Testing the System Performance with Threshold

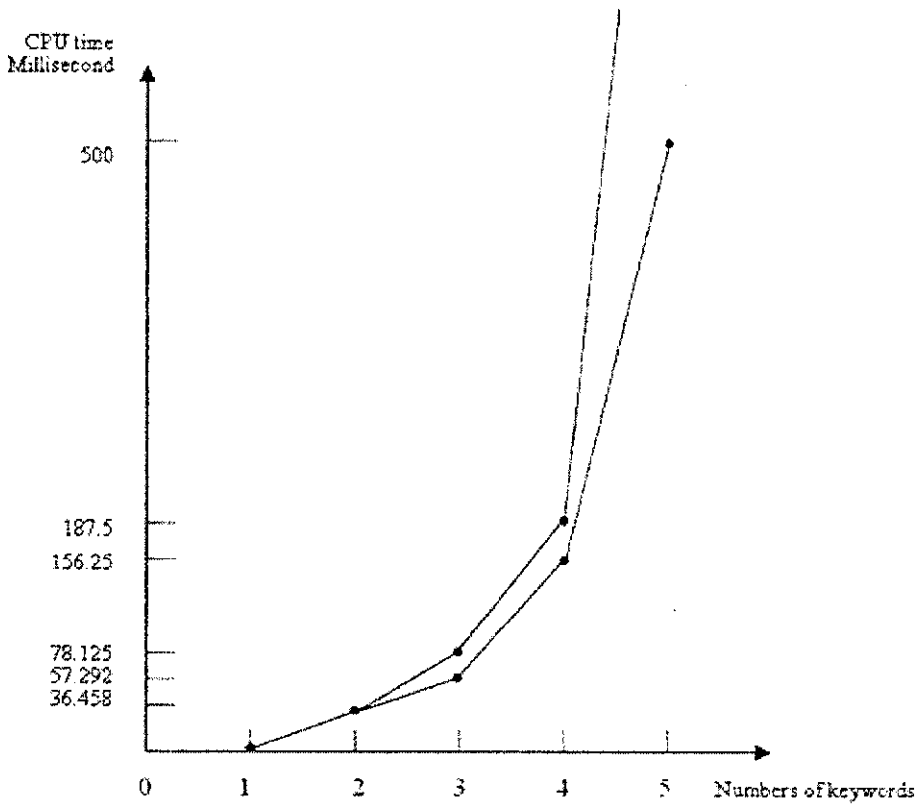


Fig 3.3 Comparison of CPU time after adding Threshold

PERFORMANCE EVALUATION

4 PERFORMANCE EVALUATION

4.1 Unit Testing

A program represents the logical elements of a system. For a program to run satisfactorily, it must compile and test data correctly and tie in properly with other programs. Achieving an error free program is the responsibility of the programmer. Program testing checks for two types of errors: syntax and logical. Syntax error is a program statement that violates one or more rules of the language in which it is written. An improperly defined field dimension or omitted keywords are common syntax errors. These errors are shown through error message generated by the computer. For Logic errors the programmer must examine the output carefully.

4.2 Functional Testing

Functional testing of an application is used to prove the application delivers correct results, using enough inputs to give an adequate level of confidence that will work correctly for all sets of inputs. The functional testing will need to prove that the application works for each client type and that personalization function work correctly.

Test case no	Description	Expected result
1	Test for all peers	All peers should communicate in the group.
2	Test for various peer in a distributed network framework as it display all users available in the group	The result after execution should give the accurate result.

Table 4.1 Functional Testing

When a program is tested, the actual output is compared with the expected output. When there is a discrepancy the sequence of instructions must be traced to determine the problem. The process is facilitated by breaking the program into self-contained portions, each of which can be checked at certain key points .

4.3 Non-Functional Testing

This testing used to check that an application will work in the operational environment. Non-functional testing includes:

- Load testing
- Performance testing
- Usability testing
- Reliability testing
- Security testing

4.3.1 Load Testing

Test case no	Description	Expected result
1	It is necessary to ascertain that the application behaves correctly under loads when 'Server busy' response is received.	Should designate another active node as a Server.

Table 4.2 Load Testing

4.3.2 Performance Testing

Test case no	Description	Expected result
1	This is required to assure that an application perforce adequately, having the capability to handle many peers, delivering its results in expected time and using an acceptable level of resource and it is an aspect of operational management.	Should handle large input values, and produce accurate result in a expected time

Table 4.3 Performance Testing

4.3.3 Reliability Testing

Test case no	Description	Expected result
1	This is to check that the server is rugged and reliable and can handle the failure of any of the components involved in provide the application.	In case of failure of the server an alternate server should take over the job

Table 4.4 Reliability Testing

4.3.4 Security Testing

It is necessary to check that the application's data is secured.

Test case no	Description	Expected result
1	Checking that the user identification is authenticated	In case failure it should not be connected in the framework
2	Check whether group keys in a tree are shared by all peers	The peers The peers should know group key in the same group

Table 4.5 Security Testing

4.4 White Box Testing

White box testing, sometimes called glass-box testing is a test case design method that uses the control structure of the procedural design to derive test cases. Using white box testing method, the software engineer can derive test cases.

Test case no	Description	Expected result
1	Exercise all logical decisions on their true and false sides	All the logical decisions must be valid
2	Execute all loops at their boundaries and within their operational bounds.	All the loops must be finite
3	Exercise internal data structures to ensure their validity.	All the data structures must be valid

Table 4.6 White Box Testing

4.5 Black Box Testing

Black box testing, also called behavioral testing, focuses on the functional requirements of the software. That is, black testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program. Black box testing is not alternative to white box techniques. Rather it is a complementary

approach that is likely to uncover a different class of errors than white box methods. Black box testing attempts to find errors in the following categories.

Test case no	**Description**	**Expected result**
1	To check for incorrect or missing functions	All the functions must be valid
2	To check for interface errors	All the interface must function normally
3	To check for errors in a data structures or external data base access.	The database updation and retrieval must be done
4	To check for initialization and termination errors.	All the functions and data structures must be initialized properly and terminated normally

Table 4.7 Black Box Testing

CONCLUSION

5.CONCLUSION:

The situation of having no way to process the information semantics due to the current Web system architecture will be improved considerably after the popularization of the next generation Web, the Semantic Web. In Semantic Web, the semantic information of the Web is recorded by RDF triple and is embedded in Web pages. In RDF triple, the concepts and their relationships are defined. We call the data defining the resource and its relations (concept and property) metadata. However, if it does not define metadata (concept and property) farther, then there is not enough information present about the semantics of the resource in the context of Web page retrieve the result set. Because the Web pages returned from the database not only include the keywords the user inputs, but also include the relations, some semantics of keywords are recorded by the form of RDF triples. So, the Web pages returned by "OntoLook" will be closer to the users' intention.

FUTURE ENHANCEMENTS

6.FUTURE ENHANCEMENTS:

The relation-based search engine is an important research field in search engines. Further work involves improving the environment of Microsemantic Web and the choice of cutting some arcs in concept-relation graph. The weight of relations in forming the property-keyword candidate set also needs to be considered. The priority ranking between concepts is an important study field. Because the number of relationships between concepts may be large, the priority ranking of relationships will affect the returned pages a lot. If one could combine the priority ranking technology and the page ranking technology to make a"relation-based page rank," it would be interesting.Because of the decentralized and heterogeneous Web, even on the same domain, it seems impossible for all Web pages to use the same ontology. So, study in semantic communication between ontologies will be needed.

APPENDICES

7.APPENDICES:

7.1 SOURCE CODE:

FrmAdminDomainRelationsTree.vb

```
Private Sub frmAdminDomainRelationsTree_Load(ByVal sender As
System.Object,ByVal e As System.EventArgs) Handles MyBase.Load
        Try
            txtRelationName.Enabled = False
            objX.clrText(Panel1)
            btnEnabled(True, False, True, False, False, True)
        Catch ex As Exception
            MsgBox(ex.Message.ToString, ,
"frmAdminDomainRelationsTree_Load")
        End Try
    End Sub

Sub btnEnabled(ByVal fbtnAddParent As Boolean, ByVal fbtnSaveParent As
Boolean, ByVal fbtnClear As Boolean, ByVal fbtnAddChild As Boolean,
ByVal fbtnSaveChild As Boolean, ByVal fbtnShowParent As Boolean)
        Try
            btnAddParent.Enabled = fbtnAddParent
            btnSaveParent.Enabled = fbtnSaveParent
            btnClear.Enabled = fbtnClear
            btnAddChild.Enabled = fbtnAddChild
            btnSaveChild.Enabled = fbtnSaveChild
            btnShowParent.Enabled = fbtnShowParent
        Catch ex As Exception
         End Try
 End Sub

 Sub addParent()
        Try
            TTPProcessGrid.Show("Processing...", btnAddParent, 300,200)
            objX.Connect()
            Dim qryString As String = ""
            Dim dSetMax As New DataSet
            dSetMax.Clear()
            qryString = "EXEC maxTreeDomain_index"
            dSetMax = objX.getDataset(qryString)
            If dSetMax.Tables(0).Rows.Count = 1 Then
txtDomainIndex.Text =
dSetMax.Tables(0).Rows(0).Item(0).ToString.Trim
                txtParentDomainIndex.Text = 0
                txtRelationName.Enabled = True
                txtRelationName.Focus()
            Else
                objX.clrText(Panel1)
            End If

            objX.disConnect()
            TTPProcessGrid.Hide(btnAddParent)

        Catch ex As Exception
```

```
            MsgBox(ex.Message.ToString, ,
"AddParentToolStripMenuItem_Click")
        End Try
    End Sub

    Sub saveParent()
        Try

            objX.Connect()
            Dim dSet2 As New DataSet
            Dim qryIs As String = "EXEC checkDomain_Name [" &
txtRelationName.Text.Trim & "]"
            Dim sqlAda As New SqlDataAdapter(qryIs, objX.sqlConStr)
            sqlAda.Fill(dSet2)
            objX.disConnect()
            If dSet2.Tables(0).Rows.Count > 0 Then
                MsgBox("Domain Name Already Found...",
MsgBoxStyle.Critical)
                txtRelationName.Text = ""
                txtRelationName.Focus()
            Else
                objX.Connect()
                Dim qryInsert As String = "EXEC SaveParent " &
txtDomainIndex.Text.Trim & " , [" & txtRelationName.Text.Trim & "] , "
& txtParentDomainIndex.Text.Trim
                objX.exeQuery(qryInsert)
                MsgBox("Record Saved...", MsgBoxStyle.Information)
                objX.disConnect()
            End If
            objX.clrText(Panel1)
            txtRelationName.Enabled = False
            btnEnabled(True, False, True, False, False, True)
            displayParent()
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "saveParent()")
        End Try
    End Sub
 Sub addChild()
        Try
            Dim selParentIndex As String = ""
            selParentIndex =
dgvDisplay.CurrentRow.Cells(0).Value.ToString

            If selParentIndex.Trim = "" Then
                MessageBox.Show("Please Select a Parent...", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Information)
            Else
                txtParentDomainIndex.Text = selParentIndex
                txtRelationName.Enabled = True
                txtRelationName.Focus()

                Dim dSetMax As New DataSet
                Dim qryString As String = ""
                dSetMax.Clear()
                qryString = "EXEC maxTreeDomain_index"
                dSetMax = objX.getDataset(qryString)
                If dSetMax.Tables(0).Rows.Count = 1 Then
```

```
                    txtDomainIndex.Text =
dSetMax.Tables(0).Rows(0).Item(0).ToString.Trim
                End If

                btnEnabled(False, False, True, False, True, True)
            End If
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "addChild()")
        End Try
    End Sub

    Sub clearText()
        Try
            objX.clrText(Panel1)
            txtRelationName.Enabled = False
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "clearText()")
        End Try
    End Sub


    Sub displayParent()
        Try
            TTPProcessGrid.Show("Searching...", btnShowParent, -400,
200)
            objX.Connect()
            Dim dSetParent As New DataSet
            dSetParent.Clear()
            dgvDisplay.DataSource = Nothing
            dSetParent = objX.getDataset("Select * from tbl_DomainTree
where Domain_Index_Ref = 0 ORDER BY Relation_Name")
            If dSetParent.Tables(0).Rows.Count > 0 Then
                dgvDisplay.DataSource = dSetParent.Tables(0)
            Else
                MessageBox.Show("Record Not Found", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation)
            End If

            objX.disConnect()
            TTPProcessGrid.Hide(btnShowParent)
            ' dgvDisplay.Columns(0).Visible = False
            'dgvDisplay.Columns(2).Visible = False
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "displayParent()")
        End Try
    End Sub


    Private Sub btnAddParent_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnAddParent.Click
        Try
            btnEnabled(False, True, True, False, False, True)
            addParent()
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "btnAddParent_Click")
        End Try
    End Sub
```

```
    Private Sub btnSaveParent_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnSaveParent.Click
        Try
            saveParent()
        Catch ex As Exception

        End Try
    End Sub

    Private Sub btnClear_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnClear.Click
        Try
            btnEnabled(True, False, True, False, False, True)
            clearText()
            displayParent()
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "btnClear_Click")
        End Try
    End Sub

    Private Sub btnAddChild_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnAddChild.Click
        Try
            addChild()
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "btnAddChild_Click")
        End Try
    End Sub
Private Sub btnShowParent_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnShowParent.Click
        Try
            displayParent()
        Catch ex As Exception
            MsgBox(ex.Message.ToString, ,
"ToolStripStatusLabel6_Click")
        End Try
    End Sub
Private Sub dgvDisplay_DoubleClick(ByVal sender As Object, ByVal e As
System.EventArgs) Handles dgvDisplay.DoubleClick
        Try
            Dim selChildIndex As Integer = 0
            Dim dsetChild As New DataSet
            Dim qryString As String = ""
            selChildIndex =
CInt(dgvDisplay.CurrentRow.Cells(0).Value.ToString)
            dsetChild.Clear()
            qryString = "Select * from tbl_DomainTree where
Domain_Index_Ref = " & selChildIndex
            dsetChild = objX.getDataset(qryString)
            If dsetChild.Tables(0).Rows.Count > 0 Then
                dgvDisplay.DataSource = dsetChild.Tables(0)
            Else
                MessageBox.Show("No Child(s) Found....", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Information)
            End If
```

```
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "dgvDisplay_DoubleClick")
        End Try
    End Sub


    Private Sub btnSaveChild_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnSaveChild.Click
        Try
            If txtDomainIndex.Text.Trim = "" Or
txtRelationName.Text.Trim = "" Or txtParentDomainIndex.Text.Trim = ""
Then
                MessageBox.Show("Input Not Found...", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Exclamation)
            Else
                objX.Connect()
                Dim qryIs As String = "Select Relation_Name from
tbl_DomainTree where Relation_Name = '" & txtRelationName.Text.Trim &
"' and Domain_Index_Ref = " & CInt(txtParentDomainIndex.Text.Trim)
                Dim dSetFound As New DataSet
                dSetFound.Clear()
                dSetFound = objX.getDataset(qryIs)
                If dSetFound.Tables(0).Rows.Count > 0 Then
                    MessageBox.Show("Child Already Found for This
Parent ...", "Error", MessageBoxButtons.OK, MessageBoxIcon.Information)
                Else
                    Dim qryInsert As String = "EXEC SaveParent " &
txtDomainIndex.Text.Trim & " , [" & txtRelationName.Text.Trim & "] , "
& txtParentDomainIndex.Text.Trim
                    objX.exeQuery(qryInsert)
                    MsgBox("Record Saved...", MsgBoxStyle.Information)
                End If
                objX.disConnect()
            End If

            objX.clrText(Panel1)
            txtRelationName.Enabled = False
            btnEnabled(True, False, True, False, False, True)
            'displayParent()
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "btnSaveChild_Click")
        End Try
    End Sub

    Private Sub dgvDisplay_CellClick(ByVal sender As Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles
dgvDisplay.CellClick
        Try
            btnEnabled(False, False, True, True, False, True)
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "dgvDisplay_CellClick")
        End Try
    End Sub

    Private Sub btnShowTree_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnShowTree.Click
        Try
```

```
            frmAdminTreeView.ShowDialog()
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "btnShowTree_Click")
        End Try
    End Sub

    Private Sub dgvDisplay_CellContentClick(ByVal sender As
System.Object, ByVal e As
System.Windows.Forms.DataGridViewCellEventArgs) Handles
dgvDisplay.CellContentClick

    End Sub
End Class
```

Frm MappingRelationship.vb

```
Private Sub btnSearch_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSearch.Click
        Try
            Dim file As System.IO.FileStream
            Dim files As String = ""
            Dim sw As StreamReader
            Dim input As String = ""
            Dim words() As String
            Dim word As String = ""
            Dim downloadPath As String = ""
            Dim cnt As Integer = 0

            Dim grid0 As String = ""
            Dim grid1 As String = ""
            Dim allWords() As String
            Dim firstWords As String = ""
            Dim temp() As String
            Dim MatchRes1 As Integer = 0
            Dim MatchRes2 As Integer = 0
            Dim MatchResult As Integer = 0
            Dim MatchCount As Integer = 0
            Dim colURL As New Collection

            colURL.Clear()
            objX.exeQuery("Truncate Table SearchedResultURL")
            If DataGrid1.RowCount > 0 Then

downloadPath = System.IO.Path.GetFullPath("..\..\db\Websites")
                For Each files In
System.IO.Directory.GetFiles(downloadPath)
TTPProcessGrid.Show("Searching...", btnSearch, -100, -100)
file = New System.IO.FileStream(files.Trim, IO.FileMode.Open,
IO.FileAccess.Read)
                    sw = New StreamReader(file)
                    sw.BaseStream.Seek(0, SeekOrigin.Begin)
                    While sw.Peek() <> -1
                        input = sw.ReadLine
                        words = input.Split(" ")
                        For Each word In words
                            If InStr(word, "<rdf:Description>") Then
```

```
Do Until input = "</rdf:Description>"
 If InStr(input, "</rdf:Description>") Then
                                        Exit While
                                    End If

input = sw.ReadLine

If input.Trim <> "" Then

For cnt = 0 To   DataGrid1.RowCount - 1

grid0 = DataGrid1.Item(0, cnt).Value.ToString.Trim

grid1 = DataGrid1.Item(1, cnt).Value.ToString.Trim

allWords = input.Split("<")

If allWords.GetLength(0) > 1 Then

temp = allWords.GetValue(1).ToString.Trim.Split(" ")

firstWords = temp.GetValue(0).ToString.Trim
    If grid1.Trim = firstWords.Trim Then
      MatchRes1 = 1
    End If
 End If
    allWords = input.Split("#")
 If allWords.GetLength(0) > 1 Then
  temp = allWords.GetValue(1).ToString.Trim.Split("""/>")
  firstWords = temp.GetValue(0).ToString.Trim
  temp = firstWords.Trim.Split(" ")
  firstWords = temp.GetValue(0).ToString.Trim
   If InStr(grid0, firstWords, CompareMethod.Text) Then
      MatchRes2 =
   End If
 End If
If MatchRes1 = 1 And MatchRes2 = 1 Then
    MatchResult = 1
    MatchCount += 1
End If
    MatchRes1 = 0
    MatchRes2 = 0
 Next
End If
Loop
End If
Next
End While
If MatchResult > 0 Then
  colURL.Add(file.Name.ToString.Trim)
     objX.Connect()
     objX.exeQuery("Insert into SearchedResultURL values ('" &
file.Name.ToString.Trim & "', " & MatchCount & ")")
      objX.disConnect()
                        'MsgBox("File : " & file.Name.ToString.Trim &
"MatchCount : " & MatchCount)
```

```
   End If
   MatchCount = 0
   MatchResult = 0
   TTPProcessGrid.Hide(btnSearch)
     Next
     If colURL.Count = 0 Then
      MessageBox.Show("Webpage Not Found....", "btnSearch_Click")
     Else
      frmDisplay.fromForm = "frmMappingRelationShip"
       Dim qrySend As String = "Select * from  SearchedResultURL order
by MatchCount desc"
                    frmDisplay.getRecords(qrySend)
                    frmDisplay.ShowDialog()
                End If

            Else
                MessageBox.Show("Keyword Not Found....",
"btnSearch_Click")
            End If

            TTPProcessGrid.Hide(btnSearch)
        Catch ex As Exception
            MsgBox(ex.Message.ToString, , "btnSearch_Click")
        End Try
    End Sub
```

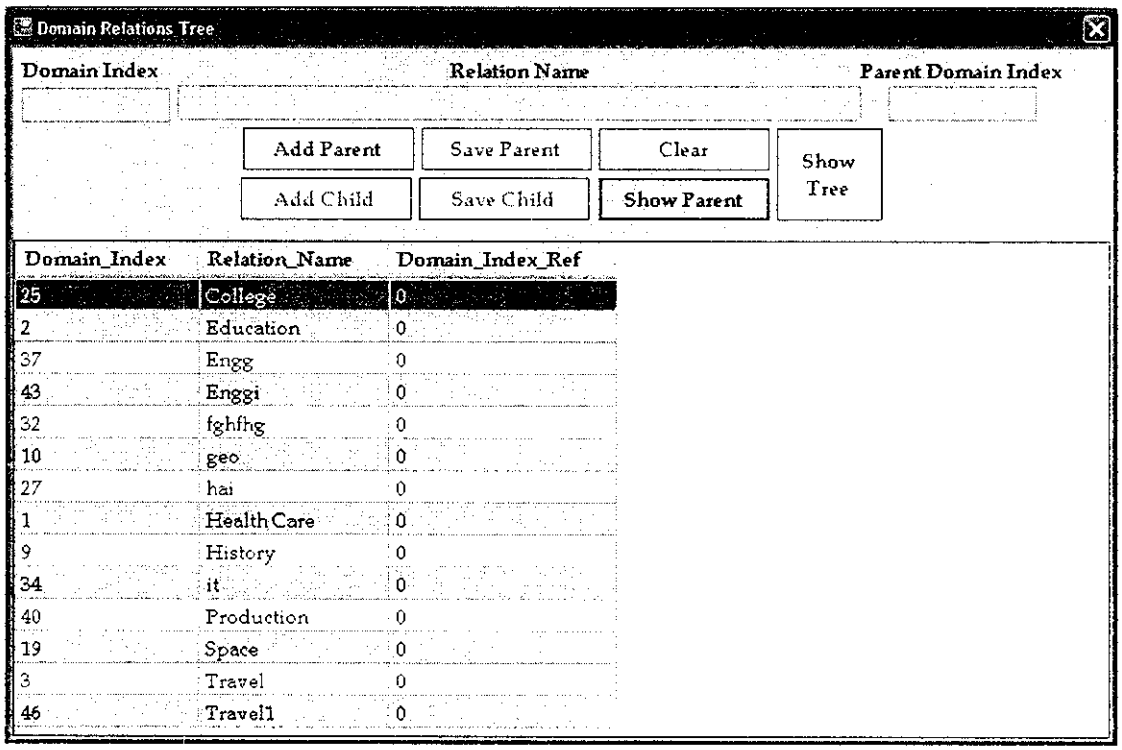
CutSearched.vb

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Try
            TTPProcessGrid.Show("Searching...", Button1, 0, 0)
            Dim downloadPath As String = ""
            Dim files As String = ""
            objX.Connect()
            objX.exeQuery("Truncate Table KeyRelation")
            objX.exeQuery("Truncate Table CutSearched")
            objX.exeQuery("Truncate Table RelationCount")
            objX.disConnect()
            downloadPath =
System.IO.Path.GetFullPath("..\..\db\Websites2")

            For Each files In
System.IO.Directory.GetFiles(downloadPath)
                'objX.exeQuery("Truncate Table KeyRelation")
                search(files)
            Next
            updateResult()


            frmDisplay.fromForm = "frmMappingRelationShip1"
            Dim qrySend As String = "Select * from  CutSearched"
            frmDisplay.getRecords(qrySend)
```

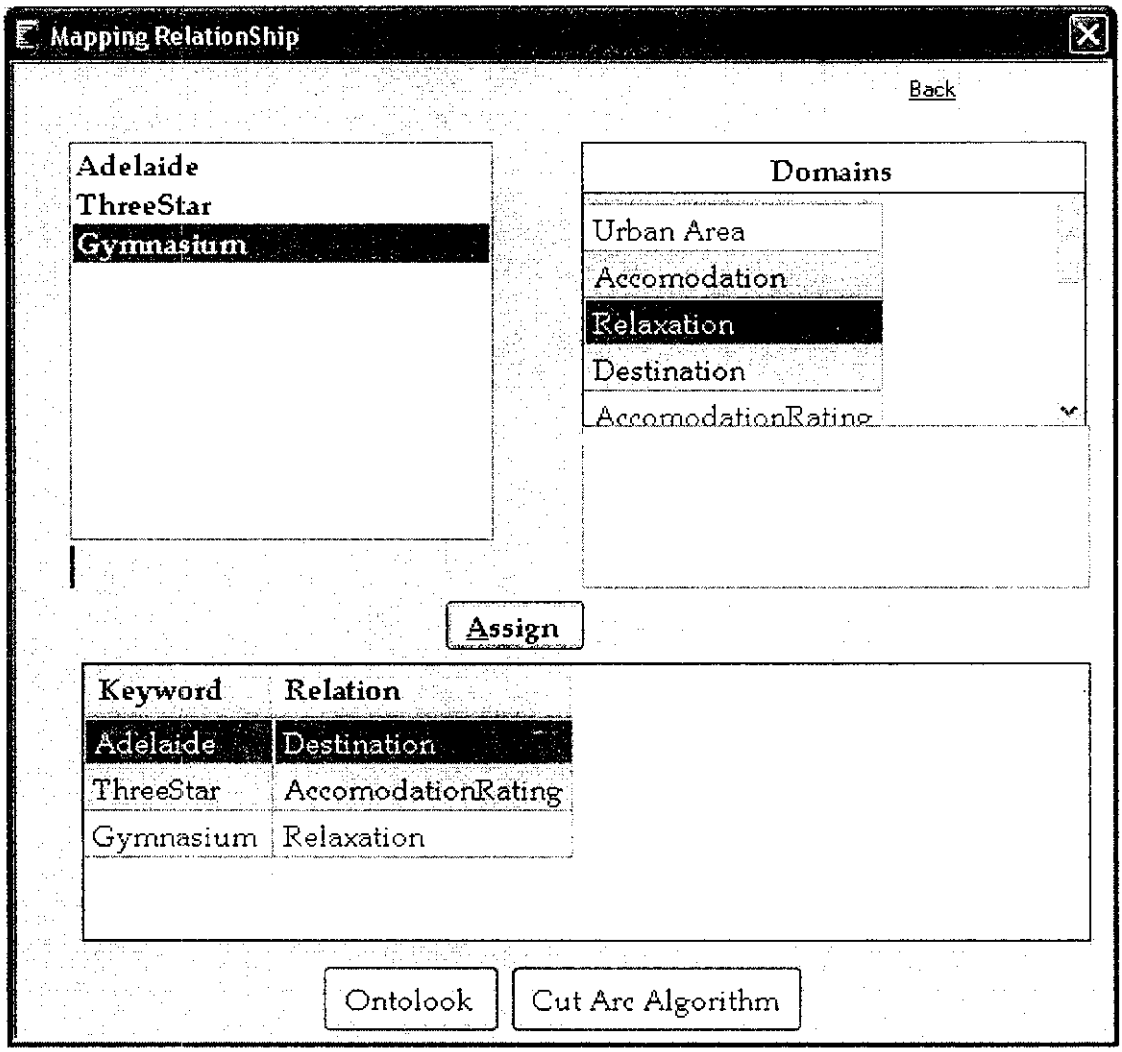
```
        frmDisplay.ShowDialog()
        TTPProcessGrid.Hide(Button1)
    Catch ex As Exception

    End Try
End Sub
```

7.2 Screen Shots

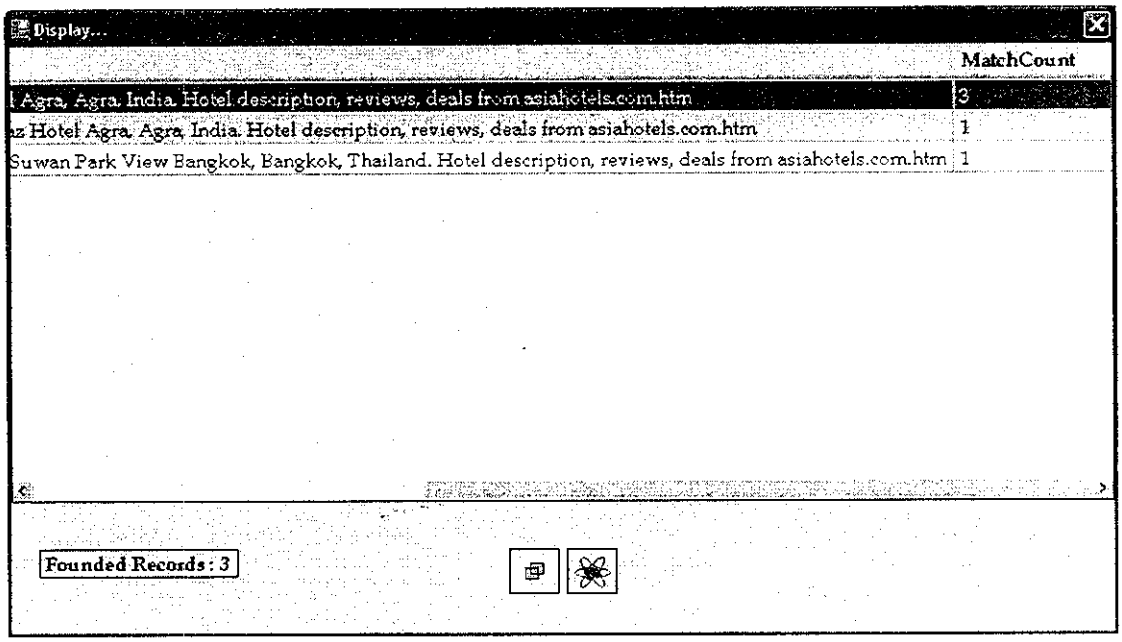
7.2.1 DomainRelationTree:



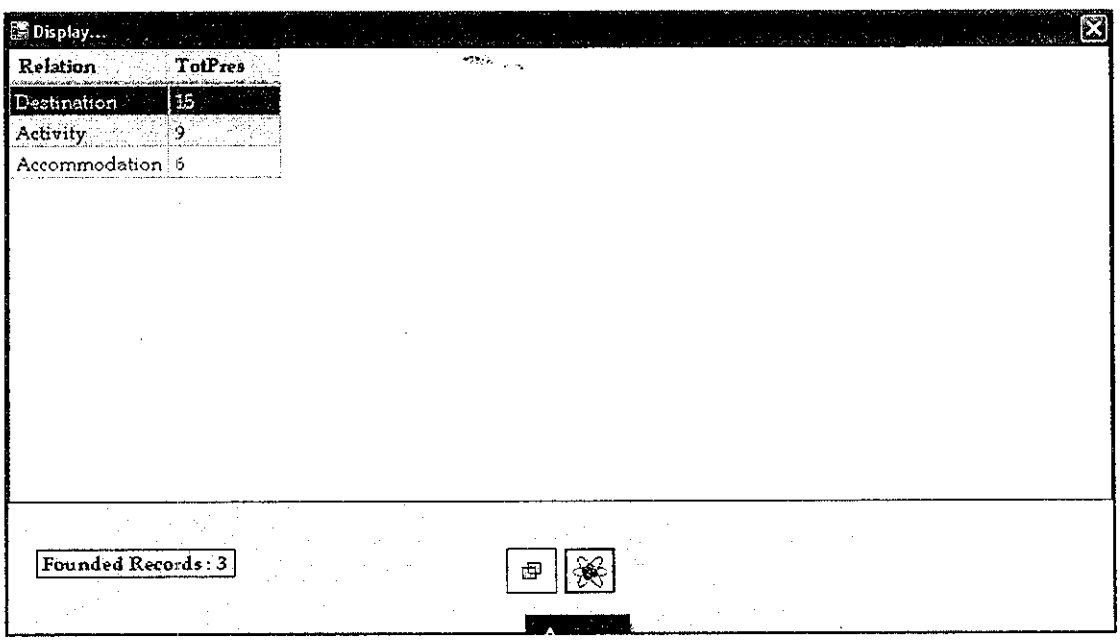
7.2.2 Mapping Relationship:



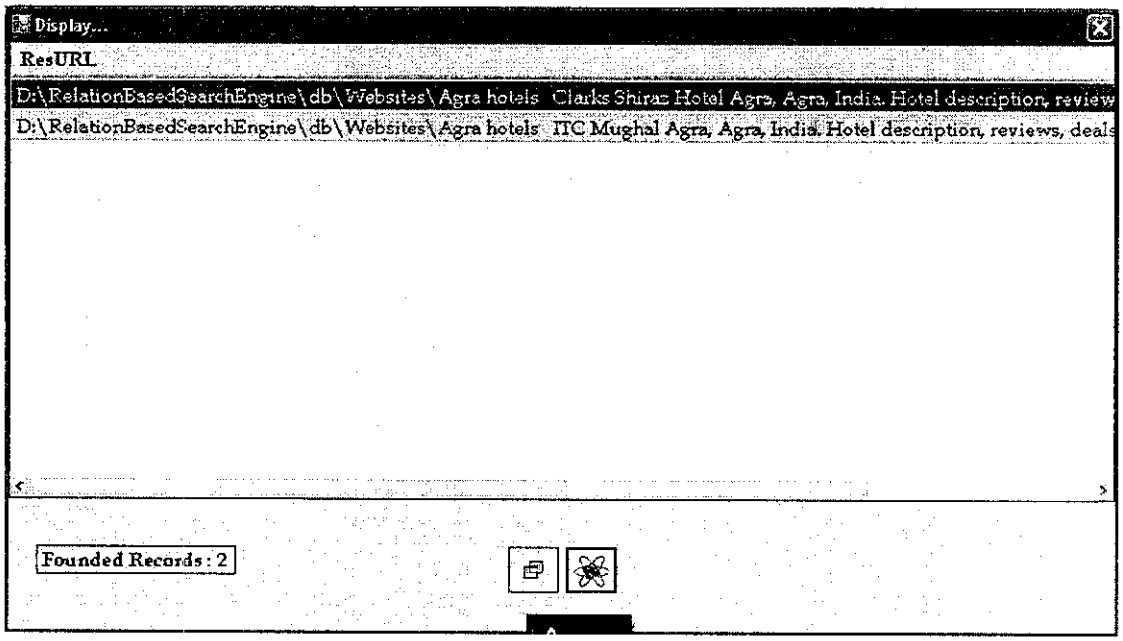
7.2.3 Retrieving URL's with Match Count:



7.2.4 Checking the Relations using Cut Arc Algorithm:



7.2.5 Revised URL After Cut-Arc Algorithm:



REFERENCES

8.REFERENCES

[1] Yufei Li, Yuan Wang, and Xiaotao Huang," A Relation Based Search Engine in Semantic Web", IEEE Transactions on Knowledge and data Engineering, vol. 19, no. 2, February 2007

[2] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web," Scientific Am., vol. 284, no. 5, pp. 34-43, 2001.

[3] D. Beckett, "RDF/XML Syntax Specification (Revised)," http:// www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/, 1994.

[4] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, and L.A. Stein, "OWL Web Ontology Language Reference," http://www.w3.org/TR/2004/REC-owl-ref-20040210/, 2004.

[5] A. Go´mez-Pe´rez and O. Corcho, "Ontology Languages for the Semantic Web," IEEE Intelligent Systems, vol. 17, no. 1, pp. 54-60, Jan.-Feb. 2002.

[6] The Google.com search engine, http://www.google.com/, 2004.

[7] N.F. Noy, M. Sintek, S. Decker, M. Crubezy, R.W. Fergerson, and M.A. Musen, "Creating Semantic Web Contents with Protege-2000," IEEE Intelligent Systems, vol. 16, no. 2, pp. 60-71, Mar.-Apr. 2001.