CHAPTER 3

QAAL SYSTEM WITH Q2Q ALGORITHM USING QUERY TEMPLATE METHOD

3.1 INTRODUCTION

The first phase of QA system is the question analysis phase based on query parsing & analysis and question classification & reformulation method. In this thesis, question classification is done with the proposed algorithm of improved question to query algorithm based on the set of rules proposed in this chapter in which semantic terms are taken for analysis of question. Before that, the improved query template method is proposed to classify the query pattern in knowledge base for a given question. In this thesis, query template is included with question to query method for question classification and reformulation. This proposed system performance is analyzed based on precision, recall, accuracy and recognition ratio.

Question analysis phase contains one part of question processing which is called question classification. Question classification is a task of assigning a boolean value to each pair. The derivation of expected answer type is carried out by means of machine learning approaches. If question classification is successful, the system might use different processing strategies to answer different types of questions.
Question for analyzing factoid based type of question is a keyword extraction. The extracted keywords are treated as topic in questions and are used for searching answers in question answering system by using information retrieval technique. Keywords in questions are like nouns, verbs, names entities. Sometime, questions are in complex manner because these questions are not represented for computers but are used for people. So, keyword extraction method is not sufficient for questions. Another important point in QA system is a category of question. These categories are listed below in Table 3.1.

**Table 3.1 Categories of Question Types**

<table>
<thead>
<tr>
<th>Question type</th>
<th>Information needs</th>
<th>Question Classification Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quantity</td>
<td>Answer is measurement</td>
<td>Factoid based type</td>
</tr>
<tr>
<td>Description</td>
<td>Answer needs description</td>
<td>Definition based type</td>
</tr>
<tr>
<td>Yes/No</td>
<td>Answer should be yes or no</td>
<td>Reason based type</td>
</tr>
<tr>
<td>Procedure</td>
<td>Answer should be a series of event for something</td>
<td>Reason based type</td>
</tr>
<tr>
<td>Definition</td>
<td>Answer is the definition of topic</td>
<td>Definition based type</td>
</tr>
<tr>
<td>Location</td>
<td>Answer is place</td>
<td>Factoid based type</td>
</tr>
<tr>
<td>Discussion</td>
<td>Answer with explanation of question</td>
<td>Explanation based type</td>
</tr>
<tr>
<td>Person</td>
<td>Answer about people</td>
<td>Factoid based type</td>
</tr>
<tr>
<td>Time</td>
<td>Answer is date or time for particular event</td>
<td>List / Factoid based type</td>
</tr>
</tbody>
</table>
Another one is query reformulation which is also called as surface pattern or answer pattern that tries to identify various ways of expressing answer given by natural language question. This reformulation is used in QA system for retrieving answer from large document collection only when the answer is not predicted at first time.

The Following items are needed for reformulation. First item is the use of syntax relations in words of question which are asked in sentence form. Second item is the use of semantic relations among words and third item is the use of existing information of previously asked question and answer.

The organization of the chapter 3 is as follows: QAAL system architecture is explained in Section 3.2. Analyses of question pattern and query formulation are described in Section 3.3 and Section 3.4. Question to query algorithm in QAAL system is explained in Section 3.5. Section 3.6 discusses the implementation of QAAL system with Q2Q algorithm. Sample query usage in QAAL system is explained in Section 3.7. In Section 3.8, the performance evaluation and discussion is based on precision, recall, accuracy and recognition ratio values. Finally concluding remarks as summary is made in Section 3.9.

3.2 QAAL SYSTEM ARCHITECTURE
answer is displayed to the user at the front end. This task is processed with various modules such as question analysis phase and question reformulation phase. Question analysis phase is divided into two sections as question parsing & analysis module and question reformulation & classification module. Searching technique is a challenging job in QAAL system. So, the proposed semantic based searching technique is implemented after query reformulation and classification module in this proposed QAAL system. In Figure 3.1, the general architecture of QAAL system with ontology and knowledge base is represented.
List of modules in the proposed architecture of QAAL system

The QAAL system has the following modules which are namely question as input, query parsing and analysis, reformulation and classification of query, semantic search, knowledge base, repositories, search engine, getting answer. All these modules already exist in QA system. Method of using these modules only differed in proposed QAAL system. These modules are supported with both front end and back end.

Query parsing and analysis: In this phase, the analytical operation of the question is found out. This analysis is responsible for processing natural language processing. It is a technique to identify the type of a question, type of an answer, subject, verb, noun, phrases and adjectives from the question. Tokens are separated from the question and the meaning is analyzed then the reformulation of query is sent to the next stage. The input is concerted into natural language and that is implemented using word segmentation algorithm. In word segmentation algorithm, the input query from the user is divided as keywords which is further subdivided and searched in knowledge base for getting correct answers.

Reformulation and classification of query: According to the user’s choice, the reformulation of query is generated with the help of WordNet which is implemented as semantic matching model.

Semantic search: At final stage, the given question is taken as a word format and the relevant concept is searched in ontology and knowledge base. Minimal answers, concept matching and computing interconnections are the three algorithms available for semantic search. Minimal answer is used to find the most specific answer. Concept
matching is based on tree matching model and computing interconnections identify related nodes in documents based on XML or RDF code. The search is carried out by using conceptual graph matching algorithm which is the best technique for matching question and query terms. Every conceptual graph has an entry and these entries are used for matching process. Each relation associated with the entry induces a sub graph. The matching is at the end when all the relations and concepts in query graph have been checked.

All the sentences in repository are framed as conceptual graph and the given question is framed as conceptual graph. The matching of question CG with given CG are checked out by using CG matching algorithms and the result is displayed at browser side of QAAL system. Graph patterns are important concept in semantic search. RDF model is organized and graph patterns are used to formulate and encode constraint queries for locating sub graph in RDF network.

Knowledge base: The knowledge base of this proposed system is domain specific. The storage of ontology is the necessary one to retrieve the relevant and correct answer from the knowledge base. In this proposed system, MySQL database is used which can be easily linked with protege. Knowledge base contains collection of classes, properties and relations in it.

Repositories: These repositories contain all the documents related to the closed domain ontology field. Proposed ontology is likely related to particular domain such as data structure ontology. The proposed document may be structured or in unstructured format which can be retrieved by the search engine.
Search engine: The user can search answers from ontology. If the concept exists in the knowledge base, the system will answer the question quickly. Otherwise, the user needs to apply web search as don’t know method. User can call meta search engine through web search interface.

Select relevant document: Using first order logic of projection algorithm, the conceptual graph matching is occurred. The transformation rules of conceptual graph matching are discussed by (Sowa 2008). The possibility of answer will be identified from various documents and the punctuation marks are removed. These documents are recovered and ordered in a specific manner.

Getting answer: This is a simple pattern matching technique to choose the appropriate response in terms of accuracy and simplicity. The proposed artificial intelligence with fuzzy logic concept is going to be implemented to get relevant answer for the given query.

Answer: The answer will be displayed in the text field of a browser after ranking. The user can accept the answer. It is needed to get more information than the query will be given to server once again. According to user satisfaction, the correct answer can be selected.

3.3 ANALYSIS OF QUESTION PATTERN

A question pattern for any type of question is defined as following term as “question-word headword”. In which Question Word (QW) is one of the terms like who/what/where/when/how and the HeadWord (HW). Headword is a word in which a set of related entries appear. It represents the words in question that reflect the intended answer. These set of chunks are described as subject and object. To determine the
exact question pattern for each question, the proposed work will be analyzed with a set of questions which are POS tagged and phrase-chunked. With the help of set of rules based on POS and chunk information, classification of questions can be carried out in an effective manner in this proposed work.

### 3.3.1 Types of Questions

Based on user’s point of view, questions are classified as factoid based, list based, definition based, reason based and explanation based type. Factoid based questions are like wh-form as what/ where/ when/which/who type. Answers are related to things, objects, location, time, date, etc. List based type expects the answer with more than one form. A collection of set of answer is listed. This format is looked as list type question and comparison is needed for this type of question.

Definition based question is expecting answer as a sentence form which is retrieved from the document and from the database. Question is like define/derive/expand form. Reason based questions are also like definition based question, but this type of question analyzes the reason for producing the answer. This type of question starts like how/why format. Explanation based questions are such complex one compared to reason based type. The situation should be analyzed from user point of view then the answer is displayed.

### 3.3.2 Creation of Rules for Question

There are two types of QA systems available nowadays. These are QA systems information retrieval with NLP and QA systems reasoning with NLP. In IR based QA systems, syntax processing and NER techniques are used in free text documents with extracted snippets
response model. But in reasoning based QA model, semantic analysis with high reasoning is used in knowledge base with synthesized responses. In this thesis reasoning based semantic analysis of QA is considered for implementation and rules are framed for this type of QA system. Rules are framed for all types of question with verb phrase, noun phrase and headword.

A question answering rule must contain a pattern that matches the question and a pattern that matches the answer for corresponding question. Another rule is a pattern to the answer which is available in given knowledge base. If a pattern defined in a rule matches a question sentence, then the rule will be applied to the sentence. To learn a QA rule, it is needed to determine the information which are common to question and an answer. So, the proposed model is in need to find maximum common graph.

3.3.3 Rule Generation

AQUALOG QA system contains a set of rule based on JAPE grammar. It provides finite state transduction over annotations based on regular expressions. These types of grammars are not suitable for all types of domain ontology. Complex questions are not get correct result while using this type of grammar. So, QAAL system suggests the simple rules for all types of question patterns.

Depending upon the phrase usage in question, totally six rules are proposed and framed for implementation in QAAL system. In the proposed QAAL system, the following rules are applied to extract words to form Question pattern ($Q_P$) with question word, headword, noun phrase and Prepositional Phrase (PP)
Rule 1 : \( Q_p = QW + HW \) in same chunk

Rule 2 : \( Q_p = QW + \) light verb + HW with NP or PP chunk

Rule 3 : \( Q_p = QW + HW \) with VP or NP chunk

Rule 4 : \( Q_p = QW + “to be” + HW \) in VP chunk

Rule 5 : \( Q_p = QW + “to be” + HW \) in NP chunk

Rule 6 : \( Q_p = QW \) when none of above rules is applicable.

Rule 1 suggests a question pattern \( Q_p \) with question word \( QW \) and the headword \( HW \). Rule 2 is used for a question pattern with question word, simple verb, headword with noun phrase or prepositional phrase. Rule 3 is used to find a question pattern in the form question word, headword with verb phrase or noun phrase type. Rule 4 is used for finding a question pattern in the form of headword with verb phrase type. If the question type is not in the above type, rule 5 is used for matching the question terms with headword in noun phrase type. Rule 6 is used for the remaining type of question pattern.

By using linguistic information of POS, it is easy to form the question pattern. These above rules are easy to understand and these are in general form. It is framed for gathering training data in an easier manner. These patterns indicate a preference of answer to be classified with proper nouns. Additionally, logic graph is used as a directed bipartite graph with two types of vertices, concepts and relations.

### 3.3.4 Sample Question Pattern Types

The Table 3.2 indicates some sample question patterns and question words. In this method, some common practice is proposed for keyword selection in query such as function words that are identified and
discarded. Then, proper nouns are treated as single search term with quotes. The proper nouns are already stored as a template format in ontology that will be checked when query is processed.

Table 3.2 Sample Question Patterns in QAAL System

<table>
<thead>
<tr>
<th>S.No</th>
<th>Types of sample question</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>What are applications of stack?</td>
</tr>
<tr>
<td>2</td>
<td>Who discovered AVL tree structure?</td>
</tr>
<tr>
<td>3</td>
<td>List out the type of sorting is used in graph.</td>
</tr>
<tr>
<td>4</td>
<td>Define linear data structure.</td>
</tr>
<tr>
<td>5</td>
<td>In which situation stack is empty?</td>
</tr>
<tr>
<td>6</td>
<td>List out the operations of linked list.</td>
</tr>
<tr>
<td>7</td>
<td>For what purpose front and rear variables are used in queue?</td>
</tr>
<tr>
<td>8</td>
<td>Explain the implementation of evaluation of postfix expression.</td>
</tr>
<tr>
<td>9</td>
<td>What are the applications of graph?</td>
</tr>
<tr>
<td>10</td>
<td>In which data structure Breadth First Search (BFS) is used?</td>
</tr>
</tbody>
</table>

In Table 3.2, second question is asked about the discovery of Adelson Velskii Landis (AVL) tree structure which is one of the types of binary tree in data structure. Additionally the question patterns are expanded on alignment and some considerations. It is mentioned as the question pattern which is identified with the given set of rules and that is expanded with disjunction of question patterns headword. The query will be the conjunction of expanded Q_p, proper names and remaining keywords. Except from Q_p, the other names and keywords will be the original in given question for retrieving better result.
In this proposed system, templates are created for factoid based questions which are also given as input. So the question template is created depending upon the different types of question. It is represented in Table 3.3 and the basic terms used in factoid model of question is who, whom, why, what, where, when, which and whose type of wh-form questions. This table is mainly used for construction of query pattern for the input type of question which is asked by the user. Templates are also generated for forming the query formulation to give better result.

Table 3.3 Types of Factoid Question and Template Representation

<table>
<thead>
<tr>
<th>Question type</th>
<th>Template Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Who, Whom</td>
<td>Person</td>
</tr>
<tr>
<td>Why</td>
<td>Reason</td>
</tr>
<tr>
<td>What, Which</td>
<td>Thing</td>
</tr>
<tr>
<td>When</td>
<td>Date, Time, Day</td>
</tr>
<tr>
<td>Where</td>
<td>Place, Location</td>
</tr>
</tbody>
</table>

3.4 QUERY FORMULATION

3.4.1 Need of Query Formulation

Query formulation is used to convert the question into a set of keyword queries that will be sent to inference engine for parallel evaluation. So, there is a need of deep analysis for checking the question related to common facts. Second important point of query formulation is the specificity of the query. Most queries contain only the most important keywords from the user’s question. So, the most specific queries are quoted as partial sentences. For increasing precision and reducing the search time, reformulation is needed when the expected answer is not returned. Query expansion technique is used for solving the above
constraints by adding additional related important terms with the query. For this, WordNet is used to find the attribute noun of adjectives. So, the adjective with its attribute noun query is replaced as new query by expanding this technique. In this thesis query reformulation and expansion is proposed and they are discussed in this following subsection.

3.4.2 Query Reformulation in QAAL System

The keyword terms used in question are first identified and the related terms are searched in ontology. If the terms are not related with the ontology, then related terms are identified by using WordNet and then query will be reformed based on relevant terms and again answer retrieved is processed. In this proposed model, sample terms and reformulated terms are noted in Table 3.4. First In First Out (FIFO) is taken in queue and Last In First Out (LIFO) is used in stack data structure for insertion and deletion operation.

<table>
<thead>
<tr>
<th>Terms</th>
<th>Related terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>stack</td>
<td>LIFO</td>
</tr>
<tr>
<td>queue</td>
<td>FIFO</td>
</tr>
<tr>
<td>implement</td>
<td>implementation</td>
</tr>
<tr>
<td>application</td>
<td>use, applications, usage</td>
</tr>
<tr>
<td>operations</td>
<td>operation, action, functionalities</td>
</tr>
</tbody>
</table>

3.4.3 Methods Used for Question to Query Form Transformation

Formulation of query for a given question is a challenging task because a correct query produces a correct answer.
Table 3.5 Example of Question to Query Form Transformation

<table>
<thead>
<tr>
<th>Pattern Type</th>
<th>Words in Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question</td>
<td>What are the applications of stack?</td>
</tr>
<tr>
<td>Question pattern</td>
<td>What</td>
</tr>
<tr>
<td>Proper noun</td>
<td>Stack</td>
</tr>
<tr>
<td>Keyword</td>
<td>Applications</td>
</tr>
<tr>
<td>Transformation</td>
<td>application, uses of, real implementation</td>
</tr>
<tr>
<td>Expanded query</td>
<td>“what” AND (“application” OR “applications” OR “uses of”</td>
</tr>
</tbody>
</table>

The type of patterns and words in questions are discussed in Table 3.5. This table shows the pattern of question, proper noun in question, keywords used, transformation type and expanded query form for a given question type. For example, the given question is asked like “What are the applications of stack”. In this question pattern “what” is the question term. Proper noun in the question is “stack” and remaining keywords are “applications”. The question will be asked with the keywords “application” or “applications”. The conjunction used here is OR and AND. OR is termed as disjunctive terms. If two terms are used, either one of the two terms will be treated for relation checking. So, in the table 3.5 OR symbol is used in expanded pattern type. Proper noun in the given question is “stack”. To search a term, this proper noun must be noted. So, AND term is used in Table 3.5 for expanded pattern type. AND is a term as conjunctive term. It returns true when both two terms conditions are checked. Otherwise, it will return false value. Expanded
query is used to merge the question pattern with proper noun and keyword terms with conjunction operators for finding the answer in knowledge base. If the expanded query is not suitable for finding out the answer, reformulation is possible with related terms for proper nouns.

3.5 QUESTION TO QUERY ALGORITHM IN QAAL SYSTEM

The input of improved Q2Q algorithm is a set of different question types such as factoid based, list based, definition based, reason based and explanation based types and the output is the triple set representation which contains subject, predicate and object types in it. The main functionality of Q2Q algorithm in the proposed QAAL system is shown in the Figure 3.2.

![Figure 3.2 Question to Query Algorithm Model in QAAL System](image)

For a given user’s question, reformulation is processed and a syntax tree is constructed for the question. Before reformulation of user question, terms are extracted from it and stop words are removed. A triple
set is initialized as empty initially. Then, the noun word sets are found based upon the triple extraction rule. Noun word sets are nouns used in question term. Triple extraction rule form the query in <subject, predicate, object> form. Words are extracted and added to triple set. Then, the query is formed with the given set of triples which are stored in triple set. After construction of query from the given question, template matching is done in knowledge base for matching the terms of query with the terms in template model to retrieve answer from it. In this proposed QA system, ontology is constructed for Data Structure (DS) domain and the possible type of questions are stored in a template format. The system should match the template with the given query. Ontology contains collection of classes, property and relation among them. Properties are described for DS ontology.

**Table 3.6 Sample Class and Relations used in QAAL System Based Data Structure Ontology**

<table>
<thead>
<tr>
<th>Property Description</th>
<th>Object Property set in DS ontology</th>
<th>Individual</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementation type</td>
<td>&lt;imp_type&gt;</td>
<td>array, linked_list, cursor</td>
</tr>
<tr>
<td>Stack operations</td>
<td>&lt;stack_oper&gt;</td>
<td>push, pop</td>
</tr>
<tr>
<td>Queue operations</td>
<td>&lt;queue_oper&gt;</td>
<td>insert, delete</td>
</tr>
<tr>
<td>Variables used in stack</td>
<td>&lt;stack_var&gt;</td>
<td>top</td>
</tr>
<tr>
<td>Variable used in queue</td>
<td>&lt;queue_var&gt;</td>
<td>front, rear</td>
</tr>
<tr>
<td>Types of search</td>
<td>&lt;search_type&gt;</td>
<td>linear, binary</td>
</tr>
<tr>
<td>Types of sorting</td>
<td>&lt;sorting_type&gt;</td>
<td>internal, external</td>
</tr>
<tr>
<td>Types of tree</td>
<td>&lt;tree_type&gt;</td>
<td>AVL_Tree, B_Tree</td>
</tr>
</tbody>
</table>

In Table 3.6, sample object properties and individuals are mentioned. These properties are used in classes of data structure ontology. These properties are used to represent terms in query retrieval process. In
this proposed model, SPARQL based query processing is implemented. The terms in questions are changed with property name for easy accessing of answer from knowledge base. For this type of property and individual, all possibilities of queries are identified and these are categorized as follows. Questions can be related to class, property and instance. Questions related to concept (rdf:TypeOf) and it is related to domain of the concept. Questions can be a boolean type or it is related to relationship among the concepts. This type of questions is used to check the class and property with ontology by checking the rules mentioned in sub section 3.3.3.

### 3.5.1 Q2Q Algorithm Design in QAAL System

The proposed improved question to query algorithm is the backbone of QAAL system which is mainly used to convert the users question into query form for retrieving answer from ontology model. The output of Q2Q model is the triple set formation which is verified with the query template model for validation of answer in triple representation. The algorithm 3.1 describes Q2Q algorithm in QAAL system. Question Q is set by the user which is given as input in Q2Q algorithm. Input question is formed as question model and a syntax tree T is formed for the given question Q after removal of stop words from the question Q.

To split up words from the question Q, WSD technique is used. Then, the triple set t is formed. Initially syntax tree T is set as NULL. The Noun Word Set (NWS) of syntax tree T is found for a question Q.

<table>
<thead>
<tr>
<th>Q</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>Syntax Tree</td>
</tr>
<tr>
<td>NWS</td>
<td>Noun Word Set</td>
</tr>
<tr>
<td>t</td>
<td>Triple set</td>
</tr>
<tr>
<td>n</td>
<td>Node in syntax tree</td>
</tr>
</tbody>
</table>
Input: Questions such as factoid, list, definition, reason and explanation based type
Output: Triple set representation like (subject, predicate, object) form.

Steps:
1. Get a question Q from the user
2. Form a syntax tree T for a given question
3. Implement WSD technique to split-up words from Q
4. Create a triple set t
5. Set T as NULL
6. For n not visited, do
7. Find NWS of syntax tree for a question Q
8. Apply triple extraction rule for words and set triple set t based on rules
9. If t is NULL or T contain t, then
do nothing
10. Else
    add t to T
11. End If
12. End For
13. Display T

Algorithm 3.1 Question to Query Algorithm

Triple extraction rule for words are applied and a triple set t is created. If t is not in syntax tree T, add triple set t to T. This procedure is repeated until all the words in the given question Q is searched. Query template is a model which has a basic format of each possible question specified to ontology model. This template method is used to reduce the time for constructing a query and also used for verification of question
type. In this proposed system, data structure ontology is used and the Table 3.7 noted the sample query templates. In this proposed model, query reformulation is made automatically and the template query model is also specified in this proposed QAAL system.

3.5.2 Automatic Creation of Query Template in Q2Q

The proposed thesis work is adopted for asking questions in closed domain environment. Particular type of questions can be asked in it. So, query template method is proposed in this thesis. Query template is an abstract representation of frequently asked questions.

A query template can be used to represent many different questions terms which convey the same meaning. A query template model covers the conceptual model of the database and it is used to describe the concepts, attributes and relationships in form of natural language question. A question template contains an entity which is replaced by data instance in database. The system retrieves related data instances and question templates from database. It offers interpretations of original questions and user selects the interpretation for getting answers from the database.

A query template model captures user intention by matching the question to predefined template and returns answer corresponding to template query focus. Based on this idea, the terms of questions are categorized as the keywords in primary terms, secondary terms and irrelevant terms of question. Primary terms type are used to focus main keywords in question and the remaining terms are ignored. In secondary terms keyword type, the meaning of the sentence is analyzed and the related terms are extracted from WordNet and query formation is carried
out. In irrelevant terms keyword type, the terms which are not relevant to the question are ignored because it does not affect the meaning of the sentence. Table 3.7 represents the sample user questions, simple pattern of question, NL query for that question and with its corresponding query template model.

**Table 3.7  Sample User Question with Query Template Model for DS Ontology**

<table>
<thead>
<tr>
<th>S. No</th>
<th>User Question</th>
<th>Simple Question type after parsing</th>
<th>Proposed NL query template related to ontology</th>
<th>Equivalent SPARQL template</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>What are the operations performed in stack?</td>
<td>Operations of stack</td>
<td>[???]→operationof→[stack]</td>
<td>select arg1 from facts where relation=[operationof] and arg2=[stack]</td>
</tr>
<tr>
<td>2</td>
<td>When stack is empty?</td>
<td>Situation of empty stack</td>
<td>stack→hasemptyat→[???]</td>
<td>select arg2 from facts where relation=[hasemptyat] and arg1=[stack]</td>
</tr>
<tr>
<td>3</td>
<td>List out the major types of searching techniques in data structure</td>
<td>Types of search in data structure</td>
<td>[???]→istypeof→[search] AND [search]→operationin→[data_structure]</td>
<td>select arg1 from facts where relation=[istypeof] and arg2=[search]</td>
</tr>
<tr>
<td>4</td>
<td>What is the subtype of binary tree?</td>
<td>Types of binary tree</td>
<td>[???]→issubtypeof→[binary_tree]</td>
<td>select arg1 from facts where relation=[issubtypeof] and arg2=[binary_tree]</td>
</tr>
</tbody>
</table>
This section overviews the generation of natural language predicted questions and types of answers from domain ontology. The relations are specified in RDF form which is well suited for SPARQL query model. In SPARQL, each query should be terminated with full stop (.). Question mark (?) is used for setting a variable either for subject or predicate or object retrieval. The answer of a SPARQL query is retrieved from ontology and is stored in XML specification.

In Table 3.7, first question is asked about the operations of stack. NL query is identified as template model for this given question. The subject, predicate and object are identified and the SPARQL query template is framed for this question. The given SPARQL template is checked and matched with the collection of query template model in QAAL system. After matching, the corresponding tree will be returned as triple set type and the answer is displayed at user side. If the matching is not found, the answer with empty string will be returned and the user will be instructed that the answer for the given question is not in the QAAL system.

In Table 3.7, the expected answer from the NL query is noted with the symbol [??/]. It is used to represent the expected answer from QAAL system. The conjunction symbols AND is used to find the answer with more than one combination. In the given table, the question 3 has asked the types of searching techniques available in data structure. In DS ontology, two ways are used to represent the searching techniques. These two ways are framed with AND symbol in query template model for Q2Q algorithm.
3.6 IMPLEMENTATION OF QAAL SYSTEM WITH Q2Q ALGORITHM

In this proposed work, closed domain ontology is taken for implementation work. Closed domain ontology means concepts related to a particular topic or area of interest. For example, ontology can describe information technology domain or computer language domain or particular branches of science. Concepts are represented as classes in ontology. Features of concepts and attributes of concepts are represented as properties. It is also called as slots. Slots describe properties of classes and instances. Classes describe concepts in the domain. For example, a class DS represents the data structure domain in computer science field. DS contains several fields like linear data structure and non-linear data structure. A class can have subclasses that represent concepts which make more specific than the superclass. For example, in this thesis ‘sort’ is represented as subclass of ‘DS’ class which has different types of sorting methods that are implemented for collection of data.

In practical terms ontology includes class definition in ontology. Ontology arranges classes in subclass-superclass hierarchy and it is used to define slots and describe allowed values for slots. Slots are filled with values for instances. This thesis is proposed to design a data structure domain with concepts, properties and objects using protege 4.1 tool for data structure ontology. The properties are created as object properties and data properties. Object properties are used to relate two concepts as a link with them and data properties are used to set a value with basic data type. Initial value should be set for array implementation of stack and queue concepts. So, data properties are necessary for these concepts. In ontology, this concept is also called as class.
Protege is an extensible platform independent environment for creating and editing ontologies and knowledge bases. Ontology terms like classes, properties and individuals are represented in protege as classes, slots and instances. Instances are used to represent objects in the domain. Properties are used for representing binary relations on individuals. Properties can be limited to have single value as functional property. It can also be represented as transitive or symmetric. The representation of class with subclass-superclass relationship is called as taxonomy.

Three types of properties are set for all concepts in ontology. These properties are object properties, data type properties and annotation properties. Object properties are used for setting the relationships between two classes and data properties are used to set values for individuals. Annotation properties can be used to add additional information to classes, individuals and object or data type properties.

Annotation is a set of data properties for checking the maximum range for implementation of stack using array. Inverse property and disjoint properties are also set for classes. Individuals are used for selecting the classes for setting range for each one. In this thesis, data structure ontology is created with several classes, properties and individuals by using protege tool.

Figure 3.3 represents the annotation created for various concepts in DS ontology. Every individual in class are represented with annotations. This annotation property is created for data property “stack_defn’ of data structure ontology in this thesis.
By creating this annotation property for defining stack, RDF format is automatically generated in protege. The first line in Figure 3.4 represents the property type used in ontology with OWL format type. Line 2 represents the domain used by this annotation. Stack is a linear data structure and the definition of stack is stored in ‘stack’ class. It is represented in line 2 which is set with range and domain values.

```
<owl:AnnotationProperty rdf:about="http://www.owl-ontologies.com/DataStructure.owl#stack_defn">
</owl:AnnotationProperty>
```

Figure 3.4 RDF Format for Annotation

Properties are created for both object type and also data type. In Figure 3.5, the object property ‘has_variable’ is set for both stack and queue. Stack has a variable top and queue has two variables which are front and rear. Inverse property is set for ‘has_variable’ with ‘var_usedby’ property.
Figure 3.5  Object Properties with Inverse Property Setting in DS Ontology

The Figure 3.6 represents RDF format setting for linear data structure type in ontology. Queue comes under linear data structure and it is mentioned with the properties. Queue is implemented by array which is set as ‘is_implemented_by’ property in ontology. Queue has two operations which is set as push and pop. In Figure 3.6, ‘has_operation’ property is set for the operation type.

<!-- http://www.owl-ontologies.com/DataStructure.owl#queue -->


<is_implemented_by rdf:resource="http://www.owl-ontologies.com/DataStructure.owl#array"/>

< has_variable rdf:resource="http://www.owl-ontologies.com/DataStructure.owl#front"/>

< has_operation rdf:resource="http://www.owl-ontologies.com/DataStructure.owl#pop"/>

Figure 3.6 RDF Format for Property Setting of Linear Data Structure
In this proposed system DS ontology, data properties are used to set a value for array, tree and searching types of graph. ‘has_value’ is set as data property for Breadth First Search, Depth First Search (DFS), array and binary tree. It is represented and the usage of that variable in different classes is shown in Figure 3.7. Individuals are for all types of classes in DS ontology and the class hierarchy with its usage is viewed in Figure 3.7. In protege tool, ‘Thing’ class is set as a super class for all derived classes. From this class only, DS class is set and the subclasses are created manually.

Figure 3.7 Data Property Setting in DS Ontology

Figure 3.8 represents the types of classes used in DS ontology. ‘Thing’ is a base class for all types of user defined classes in ontology. The classes which are created in DS ontology are the subclasses of this ‘Thing’ type. Basic data structure concept like ‘implementation’, ‘operations’, ‘search’, ‘sort’, ‘linear’, ‘non-linear’ are created as classes in DS ontology.
Figure 3.8 Class Hierarchy and Usage of Class Hierarchy

Individuals are the term used to refer the value in class representation. Each individual is created and used in difference classes. BFS is an individual which is used in ‘search’ class and it is represented in Figure 3.9.

Figure 3.9 Individuals and its Usage
The overall tree view representation of DS ontology with the individuals of each class is represented in Figure 3.10. Circle represents types of classes and its subclasses. Diamond represents the individuals which are created for these classes. Arrow indicates the relations which are created for the classes and for other individuals in DS ontology. This Figure 3.10 gives only part of DS ontology hierarchy model in which data structure search type is represented in tree form.

![Figure 3.10 Tree View Representation of DS Ontology](image)

Figure 3.11 represents the types of operations used in data structure concept. Basic operations are insert, delete, display and search. These are represented as individuals in DS ontology with the class ‘operations’.
In Figure 3.12, data structure implementation type is represented in tree form. Three types of implementation mechanism are followed in data structure namely array, linked list and cursor implementation. These features are represented as individuals in
Figure 3.12 with the ‘implementation’ class of DS ontology. Figure 3.13 represents the RDF format created automatically for setting implementation types in DS ontology.

```xml
<!-- http://www.owl-ontologies.com/DataStructure.owl#array -->
  <rdf:type rdf:resource="http://www.owl-ontologies.com/DataStructure.owl#implementation"/>
  <has_value rdf:datatype="&xsd;string">array</has_value>
  <has_implementation rdf:resource="http://www.owl-ontologies.com/DataStructure.owl#queue"/>
  <has_implementation rdf:resource="http://www.owl-ontologies.com/DataStructure.owl#stack"/>
</owl:NamedIndividual>

Figure 3.13 RDF Format for Implementation of DS Ontology

Figure 3.14 shows the sorting types used in DS ontology. Internal sorting and external sorting are the two types of sorting used in DS ontology. Insertion, quick, bubble and selection sorting are the types of internal sorting. Merge sort and radix sort are the types of external sorting. These criteria are represented in DS ontology and Figure 3.14 shows the graphical form of relation with types of sorting in DS ontology.
RDFS is a schema which is used to represent the outline view of ontology. All the classes with their object properties, data properties with the individual are successfully created in DS ontology and the RDFS representation is shown in Figure 3.15.

```xml
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
  <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >

  <rdf:RDF xmlns=http://www.owl-ontologies.com/DataStructure.owl#
           xml:base="http://www.owl-ontologies.com/DataStructure.owl"
           xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
           xmlns:owl="http://www.w3.org/2002/07/owl#"

Figure 3.14 Tree View for Types of Sorting in DS Ontology

Figure 3.15 RDFS Representation for DS ontology
After generating ontology model for QAAL system, SPARQL queries are implemented with this ontology using protege tool. This proposed QAAL system is implemented using Jenawith java code based on Graphical User Interface (GUI) based model. This GUI model is used for giving input from the user in text box and retrieves the result from knowledge base with the help of Jena in java environment. The final result is published at user side in listed order. Using eclipse tool, the client/server environment was created.

Jena is a java API for RDF environment. It is used to parse, create and search RDF models in ontology. It is easy to use. RDF models are a collection of statements. A statement is represented as triple set form such as subject, predicate and object format. This Jena code can be used with Eclipse tool. Eclipse is a programming environment which is basically supports java code and certain packages related to ontology can be imported in it based on reference form.

3.7 SAMPLE QUERY USAGE IN QAAL SYSTEM

There are different types of queries asked by the user. It may also be the possibility of asking negative query type by the user like “Is stack a linear data structure?”, “Is Graph a type of tree?”. Another set of problem raised in users question is the wh-terms which may sometime have generic type, unknown terms in question and unknown relation from question to query model. So, there is a need of categorizing wh-terms with equivalent intermediate representation. It is possible with triple set model.

SPARQL is both a query language and a data access type which has the ability to become a major component in web 2.0 applications. It
contains query language specification which means for conveying query to query processor type with XML format in which query results will be returned. SPARQL also supports RDF model of triple set such as subject, predicate and object form. The following are some sample users questions and its corresponding SPARQL template which are generated for Q2Q reference type. The SPARQL query is like SQL query and the syntax of SPARQL query is defined as follows:

```
SELECT ?subject ?predicate ?object
FROM < rdf file name>
WHERE { ?subject rdffile:predicate?object. }
```

The above query is syntax for representing SPARQL form in which identifiers starting with question mark (?) is used to identify variables. The same format is used in QAAL system ontology model. A few questions with the queries are explained in following paragraphs. ":" symbol is used to refer the property value which is either set as object property or data property. The following examples are asked by the user.

**Example one: What is stack?**

SPARQL query is described as SELECT ?ds WHERE
{ ?subject :stack_defn ?ds . ?subject :has_value "stack". }

The definition of the stack is stored in data structure ontology. It is referred with the object name stack. So, the query is written in the above format. Here, the subject stack definition is set with the variable “ds” and the property is set as ‘stack_defn’ in ontology. Stack holds the object
property as ‘has_value’ in ontology. With this object property name only it will be referred in ontology and the definition will be extracted from it.

Example two: Which data structure is used to implement evaluation of postfix expression?

SPARQL query is described as

```
SELECT ?dsWHERE
{ ?subject :isan_applicationof ?ds . ?subject :has_value "postfix_eval" }
```

In the ontology, application of stack is stored as values which are referred using object property names. Here, ‘isan_applicationof’ property is set for stack with the value is set as ‘postfix_eval’ in ontology. Using two conditions, the query is written in above format.

Example thress: List out the implementation type and variables used in different data structure.

SPARQL query is described as

```
SELECT ?implementation ?DS ?varWHERE 
{ {?implementation :has_implementation ?DS .} 
UNION {?DS :has_variable ?var .} }ORDERBY ?DS
```

UNION property is used to combine more than one condition checking. ORDERBY type is used to arrange the answer in ordered manner. The order may be in ascending order for numeric result and alphabet order for text based result. For every data structure type, variables are declared with the object property ‘has_variable’ in ontology. Implementation mechanism of each data structure model is described with the object property name ‘has_implementation’. The possible list of user questions and its corresponding RDF schema are noted in Appendix 1.
3.8 PERFORMANCE EVALUATION AND DISCUSSION

The proposed improved Q2Q algorithm is implemented in the system with information retrieval technique. The experimental results are justified with the effectiveness of question analysis with query parsing & analysis and question classification & reformulation. In this proposed work, the POS tagger is used to perform shallow parsing of questions and answer types. The tagger is used from WordNet type.

Totally 420 possible query template models are created with 250 classes, properties and individuals in DS ontology. After conversion of query from question, there is a need of verifying the resultant query in correct format or not. So, sample SPARQL queries are generated and answer is verified with ontology. The performance evaluation is done based on precision, recall, accuracy of question and the recognition ratio. These metrics are evaluated in both AQUALOG QA system and the proposed QAAL system. These data are created using protege in RDF form. If the answer is retrieved correctly from ontology, those questions will be taken as predicted type questions. Questions may also be asked from outside domain or unpredicted type of question. Unpredicted type of question is identified only after five iterations of query reformulation and expansion type. Complex types of questions like reason based and explanation based questions are come under this unpredicted type of question pattern. Outside domain question type is suggested when the question is not asked relevant in particular domain.

3.8.1 Comparison of Question Types

Table 3.8 summarizes the type of questions and corrected answers for a given set of questions in this proposed QAAL system.
Totally 340 questions are posted in proposed system. Within this, 224 are predicted type of question in DS ontology, 47 questions are asked from outside domain which are not relevant in DS ontology and the remaining 23 questions are in unpredicted form. Then, the analysis is made for the answer which is in correct form or incorrect form. Out of 270 total questions 224 are correct and predicted questions type with the average performance percentage of 82.96%. Most of the correct answers are from the predicted type.

Table 3.8 Comparisons of Types of Questions used in QAAL System

<table>
<thead>
<tr>
<th>No.of Questions used</th>
<th>Predicted type</th>
<th>Outside domain</th>
<th>Unpredicted type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total questions</td>
<td>270</td>
<td>47</td>
<td>23</td>
</tr>
<tr>
<td>Correct questions</td>
<td>224</td>
<td>24</td>
<td>4</td>
</tr>
<tr>
<td>Incorrect questions</td>
<td>46</td>
<td>23</td>
<td>19</td>
</tr>
<tr>
<td>Percentage</td>
<td>82.96</td>
<td>51.06</td>
<td>17.39</td>
</tr>
</tbody>
</table>

Totally 47 questions are asked in outside domain and 23 questions are found as unpredicted type of questions out of 340 questions tested. Compared to outside domain model and unpredicted type of question, several questions are in predicted form with the average performance of 82.96% by implementing Q2Q algorithm in QAAL system which is shown in Figure 3.16.

The proposed system is compared with AQUALOG QA system and template matching is implemented in it. In AQUALOG QA system, two modules were used as linguistic component and relation similarity services. Linguistic component was used to convert natural language question into query triples format. But, the main limitation of linguistic
model was the association of syntactic annotation of input query. So, it could not resolve the ambiguity associated with the natural language query. This query triples format was given as input to the relation similarity service module. With the help of user feedback the ambiguity was solved.

![Figure 3.16 Performance Analysis of QAAL System](image)

**Figure 3.16 Performance Analysis of QAAL System**

In this proposed QAAL system, annotations are used in semantic form and the usage of user feedback is less compared to AQUALOG QA system.

### 3.8.2 Precision and Recall Ratio

The system performance is also analyzed depending upon the recall and precision ratio. Figure 3.17 contains precision and recall values
which are represented as 10 sets and the corresponding values are shown in query template model for QAAL system and AQUALOG QA systems.

50 possible types of questions are framed as one test set. 10 possible sets of question with 50 questions in each set is framed for analyzing performance of QA system. Totally 10 set of recall values are measured in proposed QAAL system and AQUALOG system. Recall values are noted from 0 to 1 and the corresponding precision value range is from 0 to 1. Q2Q algorithm with query template method in QAAL system gives a better result of high precision and it is approximately high in each recall values. If recall increases then the precision value will be reduced gradually.

![Comparison of Q2Q Method in QAAL Vs AQUALOG in Test Set Five](image)

*Figure 3.17  Comparison of Q2Q Method in QAAL Vs AQUALOG in Test Set Five*
In Figure 3.17, recall with precision ratio is compared with QAAL system and AQUALOG QA system by 10 sets of values. The Figure 3.17 shows the results for the test set five. This test set contains more number of complex questions like reason and explanation based type in it. AQUALOG QA system produces 0.26 for the 0 recall. But, QAAL system produces 0.37 for the same recall value. The precision ratio for the remaining set of recall value is high in QAAL system compared with AQUALOG QA system. For 0.9 recall value, precision in QAAL system reached 0.02 value and AQUALOG system reached 0.01 precision value.

With this performance evaluation, it is concluded that the performance of the QAAL system with query template based Q2Q algorithm is improved precision values by 2% compared to AQUALOG QA system. So, if the precision value is high then the performance of QA system will be improved automatically. In addition to these two types of metrics used in Q2Q algorithm evaluation accuracy and recognition ratio are also taken for evaluation.

### 3.8.3 Accuracy and Recognition Ratio

The additional metrics used to measure the performance of QAAL system are accuracy and recognition rate type. Accuracy is defined as the ratio between the number of questions matched correctly by the number of identified questions.

\[
\text{accuracy} = \frac{\text{number of questions matched correctly}}{\text{number of identified questions}}
\]

In Equation (3.1), identified questions are represented as the number of questions matched by the template in the system. Accuracy is
the important indicator for the quality of collection of query templates as well as matching score formula.

Recognition ratio is the ratio of the number of identified questions by the total number of test questions which reflect the domain knowledge by the template collection. It is noted in Equation (3.2).

$$\text{recognition ratio} = \frac{\text{number of identified questions}}{\text{total number of questions}}$$

(3.2)

If the accuracy and recognition ratio is more than 0.5, the system performance is good. Otherwise, the system is considered in under performance. The Figure 3.18 gives the evaluation values of accuracy ratio with the given set of query template type for six test sets in both predicted and unpredicted question set.

![Figure 3.18 Accuracy Ratio in QAAL Vs AQUALOG](image)

Figure 3.18 Accuracy Ratio in QAAL Vs AQUALOG
Totally 10 sets of evaluation is done and in Figure 3.18 the last five test sets with the accuracy values are shown. Test set five contains more complex types of questions like reason and explanation based type. Test set 10 contains more factoid based questions in it. Compared to unpredicted question set accuracy performance, predicted question set produced high performance in QAAL system.

In AQUALOG QA system, the last five test set values are examined which are occurred within the range from 0.52 to 0.76. If the accuracy value is greater than 0.5 then the performance of QA system will be taken as good. So, AQUALOG QA system performance is good for the given test sets.

The performance of accuracy is also tested in the proposed QAAL system by using question template model with improved Q2Q model. Figure 3.18 shows the accuracy rate which has been increased to 5.83% (approximately 6%) in QAAL system compared with AQUALOG QA system. The minimum accuracy ratio from the figure shows 0.58 in QAAL system whereas 0.52 accuracy is produced in AQUALOG QA system in test set five because that test set contains maximum number of complex questions in it. At the maximum, 0.81 accuracy ratio is reached in proposed QAAL system by using query template with Q2Q algorithm. This analysis shows that the performance of proposed QAAL system is high than AQUALOG QA system.
Recognition ratio has been taken as the last parameter for performance evaluation in this proposed work. In this proposed system, the initial question template set is evaluated from 6 questions onwards. Figure 3.19 shows the recognition ratio measured with different test sets in QAAL system and AQUALOG QA system. Recognition ratio is calculated for all 10 test sets but in Figure 3.19, the last six test sets performance is noted.

The recognition ratio value for AQUALOG QA system is reached 0.72 where QAAL system produced 0.79. It shows the ratio value for the QAAL system is high in test set five. The same range is produced in the remaining test sets six to ten. Recognition ratio is also considered for the performance of QA system. If the value of recognition ratio is greater than 0.5 then, the system will be treated as good. If the value of recognition ratio is greater than 0.8 then, the system will be considered as excellent. Among these six different test sets, the recognition ratio for
QAAL system reaches a high value as 0.91 whereas 0.87 value is reached in AQUALOG QA system. So, depending upon the recognition ratio metric the performance of QAAL system by using query template with Q2Q algorithm is excellent for the given DS ontology environment and it is concluded that the system produces correct answers mostly.

It is concluded that compared with AQUALOG QA system, recognition ratio is improved to 3.16 % in QAAL system by using query template with Q2Q algorithm.

3.9 SUMMARY

The proposed QAAL system is designed to understand the given question with the semantic search model and the template is used for correcting and reformulating the question into query form. Related with these criteria, this proposed an improved Q2Q algorithm which is implemented and tested in QAAL system based on query template model which is used to convert the given question to query model. It is used to find respective answer model prescribed in data structure ontology in a successful manner. In the existing AQUALOG QA system, syntactic annotations were used and implemented for conversion of natural language query into query triple form. But, ambiguity occurred and it could not be solved in this model. This problem was noted and it is solved in this proposed QAAL system by using Q2Q algorithm model with query template model.

Precision and recall are the main metrics used for evaluation of question answering system. By using query template model with Q2Q algorithm in the proposed QAAL system, the QA system performance is
increased to 2% in precision and recall values compared to existing AQUALOG QA system. Accuracy and recognition ratio are taken as another two performance metrics in query template method. Accuracy is increased 6% approximately in QAAL system and recognition ratio is increased to 3.16% in QAAL system compared with AQUALOG QA system.

The semantic relations are checked by using text based method. But, for effective retrieval of an answer semantic search is needed. Searching is different in graph model compared to searching in linear representation data structure model. So, it is proposed to use a graph based searching mechanism with semantic search model and it is discussed in next chapter.