نموذج رقم (1)

إقـــــرار

أنا الموقع أدناه مقدم الرسالة التي تحمل العنوان:

نظام عربي للاجابة على الأسئلة اعتمادًا على الأنتولوجي
Ontology - Based Arabic Question Answering System

أقر بأن ما اشتملت عليه هذه الرسالة إنما هو نتاج جهدي الخاص، باستثناء ما تمت الإشارة إليه حيثما ورد، وإن هذه الرسالة ككل أو أي جزء منها لم يقدم من قبل لنيل درجة أو لقب علمي أو بحثي لدى أي مؤسسة تعليمية أو بحثية أخرى.

DECLARATION

The work provided in this thesis, unless otherwise referenced, is the researcher's own work, and has not been submitted elsewhere for any other degree or qualification

Student's name: اسم الطالب: آلاء وليد عياد أبو طه

Signature: التوقيع:

Date: التاريخ: 2015/3/25

Islamic University-Gaza
Deanery of Graduate Studies
Faculty of Information Technology

An Ontology-Based Arabic Question Answering System

Submitted By:
Alaa W. AbuTaha

Supervised By:
Dr. Iyad M. Alagha

A Thesis Submitted as Partial Fulfillment of the Requirements for the Degree of Master in Information Technology

1436 H - 2015

بسم الله الرحمن الرحيم

الجامعة الإسلامية – غزة
The Islamic University - Gaza

مكتب نائب الرئيس للبحث العلمي والدراسات العليا **هاتف داخلي**: 1150

الرقم.ج س.غ/35/..... Ref

التاريخ 2015/02/03م Date

نتيجة الحكم على أطروحة ماجستير

بناءً على موافقة شئون البحث العلمي والدراسات العليا بالجامعة الإسلامية بغزة على تشكيل لجنة الحكم على أطروحة الباحثة/ **آلاء وليد عياد ابو طه** لنيل درجة الماجستير في ***كلية تكنولوجيا المعلومات*** **برنامج تكنولوجيا المعلومات** وموضوعها:

نظام عربي للإجابة على الأسئلة اعتمادًا على الأنتولوجيا

Ontology-Based Arabic Question Answering System

وبعد المناقشة التي تمت اليوم الثلاثاء 13 ربيع الآخر 1436هـ، الموافق 2015/02/03م الساعة الحادية عشرة صباحاً، اجتمعت لجنة الحكم على الأطروحة والمكونة من:

د. إياد محمد الأغا	مشرفاً و رئيساً	.....
د. ربحي سليمان بركة	مناقشاً داخلياً	.....
د. يوسف نبيل أبو شعبان	مناقشاً خارجياً	..... 03.02.2015

وبعد المداولة أوصت اللجنة بمنح الباحثة درجة الماجستير في ***كلية تكنولوجيا المعلومات*** **/ برنامج تكنولوجيا المعلومات.**

واللجنة إذ تمنحها هذه الدرجة فإنها توصيها بتقوى الله ولزوم طاعته وأن تسخر علمها في خدمة دينها ووطنها.

والله ولي التوفيق،،،

مساعد نائب الرئيس للبحث العلمي والدراسات العليا

C.15

أ.د. فؤاد علي العاجز

P.O. Box 108, Rimal, Gaza, Palestine fax: +970 (8) 286 0800 فاكس Tel: +970 (8) 286 0700 ص.ب. 108 الرمال، غزة، فلسطين هاتف
public@iugaza.edu.ps www.iugaza.edu.ps

Abstract

Question Answering (QA) is a field that has been widely explored in previous research. However, research in Arabic QA is still limited and has not reached the same level of English QA due to the Arabic language specific challenges. Most importantly, existing research in Arabic QA has not explored the field of QA on the Semantic Web, and has mainly focused on information retrieval from unstructured Arabic documents.

Nowadays, a huge amount of information is available on the Web in terms of RDF and OWL. The Web is evolving rapidly towards the notion of Linked Data where the data is linked by exploiting the Semantic Web technologies and standards. This information can be queried by using the standard SPARQL. However, naïve users who have no experience with Semantic Web cannot express their questions in SPARQL. This problem can be resolved by using Natural Language (NL) interfaces that translate NL queries to SPARQL. While many efforts explored the design of NL interfaces for the Semantic Web, none of these efforts, to our knowledge, considered the use of Arabic language which has unique characteristics.

This work aims to make a step towards supporting Arabic QA on the Semantic Web. It introduces the QA system that can interface to any Arabic ontology, get a NL user query as an input and retrieves an answer from a RDF knowledge base.

The core of the system is the approach we propose to translate Arabic NL queries to SPARQL. The approach makes intensive use of the ontology semantics to translate the user query to RDF triple patterns and infer any missing components to build up a complete SPARQL query. The proposed approach can process queries of different complexities and structures.

The proposed system has been preliminary tested with a sample ontology and a testing set consisting of 30 different questions. Results have shown that the system can correctly answer 28 out of the 30 questions.

Keywords: *Question Answering, Natural Language Processing, Ontology, RDF triple, SPARQ.*

الملخص

نشرت العديد من الأبحاث التي تتحدث عن مجال نظم إجابة الأسئلة Question Answering (QA)، ومع ذلك، فإن بحوث أنظمة إجابة الأسئلة العربية لا تزال محدودة ولم تصل إلى مثيلاتها في اللغة الإنجليزية واللغات الأخرى، وذلك بسبب الصعوبات التي تواجهها عملية معالجة اللغة العربية. والأهم من ذلك أن نظم إجابة الأسئلة العربية الموجودة حالياً لم تستخدم مفاهيم الويب الدلالي Semantic Web، وقامت بالتركيز بشكل أساسي على استرجاع المعلومات من الوثائق العربية الغير منظمة.

في الوقت الراهن، يوجد العديد من المعلومات المتاحة على شبكة الإنترنت متوفرة بتقنيات RDF و OWL. ومع التطور السريع في الشبكة العنكبوتية لوحظ نمو مفهوم البيانات المرتبطة Linked Data باستخدام تقنيات ومعايير الويب الدلالي، حيث يمكن الاستعلام عن هذه المعلومات باستخدام لغة SPARQL القياسية.

إن المستخدمين المبتدئين، الذين ليس لديهم أية خبرة في مجال الويب الدلالي، لن يتمكنوا من الاستعلام عن أسئلتهم باستخدام لغة SPARQL القياسية، لذلك كان لابد من استخدام اللغة الطبيعية Natural Language كواجهة من أجل ترجمة استعلامات المستخدمين المبتدئين إلى لغة SPARQL. وبالرغم من ان هناك العديد من الجهود التي وجهت لتصميم واجهات تمكن من استخدام اللغة الطبيعية لترجمة استعلامات للويب الدلالي، فإنه لا يوجد أي من هذه الجهود –على حد علمنا–، تأخذ بعين الاعتبار خصائص اللغة العربية الفريدة من نوعها.

إن الهدف الأساسي من وراء هذا العمل هو القيام بخطوة أولى نحو استخدام اللغة العربية في نظم إجابة الأسئلة مبنيةً على مفاهيم الويب الدلالي، حيث أن النظام المقترح قادر على التعامل مع أي انطولوجي عربية فهو يقوم باستقبال استعلامات المستخدمين باللغة الطبيعية ومن ثم تقديم الاجابات لهم.

هذا النظام هو عبارة عن نهج مقترح لترجمة الاستعلامات باللغة العربية إلى لغة SPARQL القياسية ويعتمد بشكل كبير على دلالات ومعاني الانطولوجي، حيث أنه يترجم استعلامات المستخدمين إلى أنماط ثلاثية RDF triple pattern ويستطيع أن يستنتج أي عنصر مفقود لبناء استعلام بلغة SPARQL بصورة كاملة ، إن هذه المنهجية قادرة على معالجة استعلامات مختلفة في درجة التعقيد والتركيب وبصورة متميزة.

ولقد تم اختبار النظام المقترح باستخدام انطولوجي بسيطة، وكانت عملية التقييم بناء على اختبار ثلاثين سؤال مختلف، وقد أظهرت النتائج أنه تمت الإجابة على 28 سؤال من 30 سؤال.

Dedication

To my beloved parents

To my dear husband

To my sweet children…

Khalid and Zain

Acknowledgements

Thanks and praise to Allah Almighty for guidance and help to complete this thesis. This thesis would not exist without the help, advice, inspiration, and encouragement of many people.

I would like to thank my supervisor **Dr. Iyad Alaga** for his time, patience, and understanding. I would also like to thank him for his advice during the period of study and his support on the general direction of this thesis and for the many questions he asked me to verify that I'm still on the right side

I am very grateful to my dear husband without his encouragement I can't do this work.

Thanks, My Father, mother, brothers and sisters for all things you do for me, your pray, patience, motivation and continues support.

Lastly, but certainly not least, I want to thank my friends, for their moral support during this study.

Alaa W. AbuTaha
Jan, 2015

Table of Contents

ABSTRACT I

الملخص II

DEDICATION III

ACKNOWLEDGEMENTS IV

TABLE OF CONTENTS V

LIST OF FIGURES VII

LIST OF TABLES VIII

LIST OF ABBREVIATIONS IX

CHAPTER 1: INTRODUCTION 1

1.1 STATEMENT OF THE PROBLEM 3
1.2 OBJECTIVES 3
1.3 IMPORTANCE OF THE RESEARCH 4
1.4 SCOPE AND LIMITATIONS OF THE RESEARCH 4
1.5 RESEARCH METHODOLOGY 5
1.6 THESIS STRUCTURE 6
1.7 SUMMARY 7

CHAPTER 2: BACKGROUND 8

2.1 NATURAL LANGUAGE PROCESSING (NLP) 8
2.2 NATURAL LANGUAGE PROCESSING (NLP) IN THE ARABIC LANGUAGE 8
2.3 QUESTION ANSWERING (QA) SYSTEM 11
2.4 ARABIC QUESTION ANSWERING 12
2.5 RDF AND RDFS 13
2.6 ONTOLOGY 14
2.7 SPARQL 16
2.8 SUMMARY 17

CHAPTER 3: RELATED WORK 18

3.1 QA IN ENGLISH LANGUAGE 18
3.2 QA IN ARABIC LANGUAGE 20
3.3 SUMMARY 23

CHAPTER 4: ARABIC QUESTION ANSWERING SYSTEM (AQAS) 24

4.1 OVERVIEW 24
4.2 DATA PROCESSING 26
4.2.1 Building Arabic Ontology Domain *26*
4.2.2 Building RDF store to create and store instances data *33*
4.2.3 Developing Ontological Dictionary *33*

4.3 QUESTION PROCESSING 39
4.3.1 Tokenization *40*
4.3.2 Complete Example of preprocessing phases in Question Processing *41*
4.4 ONTOLOGY MAPPER 42
CHAPTER 5: INTERPRETATION OF NL QUERY INTO SPARQL 50
5.1 TRIPLE BUILDER 50
5.2 INTERPRETATION OF NL QUERIES THAT MAP TO COMPLETE TRIPLE PATTERNS 52
5.3 INTERPRETATION OF NL QUERIES THAT DO NOT MAP TO COMPLETE TRIPLE PATTERNS 54
5.3.1 Omitted Predicate *54*
5.3.2 Omitted Object *56*
5.4 INTERPRETATION OF NL QUERIES THAT CONSIST OF MULTIPLE QUERIES LINKED WITH CONJUNCTIONS. 57
5.5 SUMMARY 60
CHAPTER 6: EXPERIMENTAL RESULTS AND EVALUATION 61
6.1. IMPLEMENTATION OF ARABIC QA SYSTEM 61
6.2. TOOLS AND PROGRAMS 62
6.3 SYSTEM EXPERIMENTS 63
6.4 SYSTEM EVALUATION 70
6.5 DISCUSSION 70
6.6 SUMMARY 76
CHAPTER 7: CONCLUSION AND FUTURE WORK 78
REFERENCES: 79
APPENDIX A: OWL SOURCE CODE 82

List of Figures

Figure 1.1: RDF triple 2
Figure 2.1: A Classification of Arabic Words According to The Part of Speech 10
Figure 4.1: Arabic Question Answering (AQAS) Architecture 25
Figure 4.2. : Pathology ontology design 28
Figure 4.3:Ontology Classes in protégé 29
Figure 4.4:Data restriction 31
Figure 4.5: Ontology Individuals 32
Figure 4.6: Example of normalization in Ontological Dictionary 35
Figure 4.7: Example of stop word removal on terms in Ontological Dictionary 36
Figure 4.8: Examples of POST in Ontological Dictionary 37
Figure 4.9: Example of stemming on terms in Ontological Dictionary 39
Figure 4.10: The processing phases of NL query 40
Figure 4.11: Tokenization the question 41
Figure 4.12 :Question before and after preprocessing 42
Figure 4.13: Ontology Mapper 43
Figure 4.14: Mapping Process 44
Figure 4.15: An example of item and itemset 45
Figure 4.16: 1-gram matching 46
Figure 4.17: 2-gram matching 46
Figure 4.18: 3-gram matching 46
Figure 4.19: Ambiguity problem in mapping process 47
Figure 4.20: Mapping to ontology terms based on POST 48
Figure 4.21: Mapping to ontology terms based on POST 2 49
Figure 5.1: An excerpt of the disease ontology 51
Figure 5.2: Example No. 1 57
Figure 5.3: Example No. 2 58
Figure 5.4: Example No. 3 59
Figure 5.5 : Example No. 4 60
Figure 6.1: Arabic QA Components 61
Figure 6.2: Triple pattern of the question No. 25 73
Figure 6.3: Triple pattern of the question No. 30 74

List of Tables

Table 4.1: Classes, Object Properties and Data Properties in the ontology 28
Table 4.2 :Ontology Classes 29
Table 4.3: Object properties in Pathology ontology 30
Table 4.4 Datatype properties in Pathology ontology 31
Table 4.5: Examples of stop words 35
Table 6.1: The size of the ontology 63
Table 6.2: The number of individuals 63
Table 6.3: questions that have been tested 64
Table 6.4: questions that have been tested and the results 65
Table 6.5 The number of questions and their results 70
Table 6.6. Testing questions and rules that are applied 70

List of Abbreviations

QA	Question Answering
AQAS	Arabic Question Answering System
NL	Natural Language
NLP	Natural Language Processing
POST	Part Of Speech Tagging
OWL	Web Ontology Language
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SPARQL	SPARQL Protocol and RDF Query Language
URIs	Uniform Resource Identifiers
IR	Information Retrieval

Chapter 1: Introduction

Nowadays the internet is becoming a huge dump of documents, links and all other sorts of information (digital libraries, newspapers collections, etc.) stored in electronic format. Most common possibilities to explore this information are information retrieval applications like web search engines such as Google, Yahoo or MSN [1]. However, the problem with most of the search engine is that they provide you with the link of the web page where you can find the links with answers instead of exact answer.

We already know that web search engines flood their users with enormous amounts of data. Despite the fact that search engines are doing an excellent job, they still return too much inaccurate information and allow a user only to retrieve the relevant documents which (partially) match a given query [2]. The solution to this problem can be found in the form of Question Answering systems, which allow users to ask questions in a natural language and provide accurate answers to the user, rather than a series of possible results waiting for analysis. Therefore, the QA system is more convenient, faster and precise than traditional search engines. It can be better meeting the search-needs of users.

Question Answering (QA) system takes questions in natural language as input, searches for answers, and extracts concise results. Various QA systems contain at least the three following parts: Question processing, Document processing and Answer processing [3]. For a question to be answered correctly, a QA system first has to understand what the question is asking about. Most QA systems rely on Natural Language Processing (NLP) tools that perform linguistic analysis to help in understand the user's query and matching sections in documents. This is an important task of question and document processing. The most common NLP system contains: tokenization, part-of-speech tagging (POS), stemming, named entity recognition (NER), semantic relations, dictionaries, WordNet, etc. [4].

Many searches have been done for expanding English language QA systems. Also, some other works have been done on Chinese, Arabic, Spanish and … QA systems. QA systems for Arabic language are very few. Mainly, it is due to the lack of accessibility to linguistic resources, such as corpora and basic NLP tools (tokenizers, morphological analyzers, etc.). Moreover, the Arabic language has a very complex morphology (inflectional and derivational characteristics) and texts suffer from the scarcity of vowels as well as the absence of capitalization. These specificities of the Arabic language introduce many processing problems related to the word tokenization, the identification and categorization of named entities, etc. [5].

The purpose of a QA system is to find exact and correct answers for user's questions in both non-structured and structured collection of data. In the case of an ontology-based QA system, knowledge based data, where answers are sought, has a structured organization defined by an ontology. Ontology defined as *"a formal explicit description of concepts in a domain of discourse (classes). Properties of each concept describe various features and attributes of the concept (slots), and restrictions on slots (facets) ontologies together with a set of individual instances of classes constitutes a knowledge base"* [6].

Ontology is becoming the pivotal methodology to represent domain-specific conceptual knowledge in order to promote the semantic capability of a QA system [7]. The question answer retrieval of ontology knowledge base provides a convenient way to obtain knowledge for use, but the natural language needs to be mapped to the query statement of ontology.

Resource Description Framework (RDF) is a framework to annotate information resources in a machine-understandable way. It is used for making statements rather than language itself. RDF contains triple syntax to express annotations as subject, predicate and object. Information resources are commonly represented as Uniform Resource Identifiers (URIs). URIs are described by RDF [8]. RDF triples are visualized as directed labeled graph in which subject, objects are represented as nodes and predicates as arcs in Fig. 1.1

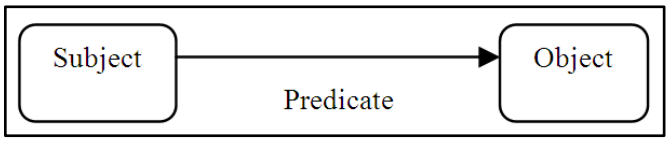


Figure 1.1: RDF triple

Web Ontology Language (OWL) is an ontology language for authoring ontologies or knowledge bases. The languages are characterized by formal semantics and RDF/XML-based serializations for the Semantic Web. RDF(S) and OWL are useful languages for representing ontologies and metadata on the semantic web. However, once this metadata has been published, query languages are required to make full use of it. SPARQL Protocol and RDF Query Language (SPARQL) aim to satisfy this goal and provides, as the name states, both a query language and protocol for RDF data on the semantic web. SPARQL offers powerful means to query RDF triples and graphs. SPARQL makes it possible to query information from databases and other diverse sources across the web [9].

Most of the research in the field of Arabic QA focused on morpho-syntactic approaches, while very little used semantic approaches [4]. We foresee that the

ontology-based question answering can offer several benefits over the traditional morpho-syntatic approach. Ontology based reasoning can help in resolving word disambiguation or identifying intelligent answers. Therefore, we have considered that an effort needs to be done in this direction.

Our work takes place in that field of research: how could we interpret a natural language (NL) query and translate it in SPARQL. We adopt a rule-based approach to capture syntactic relations and dependencies between the query terms. The process of constructing a SPARQL query from the NL query consists of procedures that are based on the complexity and completeness of the NL query.

In this research, we intend to develop an Arabic Question Answering system which is a domain-specific system, and it can work with any domain ontology represented in Arabic language. Our proposed system uses a combined approach of ontology-based reasoning and natural language processing to automatically answer questions posed by humans in a natural language.

1.1 Statement of the Problem

The field of Arabic question answering is not widely explored due to the challenges introduced by the NLP of Arabic language. Furthermore, it is unclear how the ontology based processing and reasoning can be integrated into the Arabic QA. Previous research has shown that the Arabic QA approaches which are solely based on syntactic and NLP were not very efficient. The field of Arabic QA on the Semantic Web remains largely unexplored, and this research aims to make a contribution to this field.

1.2 Objectives

- **Main objective**

The main objective of this research is to develop an ontology based Arabic Question Answering system that transforms NL queries in Arabic to SPARQL queries.

- **Specific objectives**

The specific objectives of this research are:

- Studying the current ontology-based QA systems approaches to determine the requirements of an approach for Arabic.
- Building a domain-specific Arabic ontology to be used in the development of the QA system.
- Using semantic relations in the ontology to add more intelligence to Arabic QA system.

- Developing ontology-based Arabic QA system model.
- Designing an approach to associate queries with their corresponding SPARQL expressions in order to retrieve the answer sought from the RDF database.
- Defining rules to capture syntactic relations and dependencies between the query terms.
- Implementing the system prototype that uses the ontology and NLP.
- Evaluating the system.

1.3 Importance of the Research

- This work aim to explore supporting Arabic QA on the Semantic Web.
- The proposed system aims to enable naïve users who have limited experience with the semantic web logic to query RDF backends through a natural language interface.
- The system can be easily configured to use any ontology as long as the Arabic translation of its content is supplied.
- Saving efforts and time by helping users to find accurate answers to their questions.

1.4 Scope and Limitations of the Research

- The proposed question answering model serves only Arabic language.
- Domain specific ontology is used for our Arabic question answering system related to the issue of "علم الامراض " (Pathology).
- We use handcrafted rules to translate the user query to RDF patterns.
- The proposed approach for transforming Arabic queries to SPARQL does not make intensive use of Arabic NLP tools. Instead, it relies on handcrafted rules and the ontology semantics. This decision was due to the limited support for Arabic NLP which is still far beyond the support for English NLP.
- The types of questions that the system will cover will be limited. Some types of questions, such as “why” and “how” questions are complex to be answered and require deep reasoning, and thus are not supported. Most existing QA system ignores these types of questions [10]. We emphasize that the main purpose of our system is not to cover and answer all sorts of questions, but rather to show how the ontology knowledge can be used to translate NL queries expressed in Arabic into SPARQL.

- The testing process of the system was limited to 30 questions and a sample domain ontology we designed for testing purposes. The assessment process was challenged by the lack of OWL test data in Arabic.

1.5 Research Methodology

Many researches and models have been proposed for developing ontology-based question answering system for the English language and other European and Asian languages. Arabic Question Answering still suffers from little attention. To accomplish the objectives of the research, the following methodology will be followed:

- **Research and survey:** Studying and analyzing current systems and applications that are related to Question Answering system in different languages (Arabic and English).
- **Design the model:** The structure of our proposed QA system is composed of four major components: Data Processing, Question Processing, Ontology Mapper and Answer Retrieval.

Data Processing: This component is responsible for maintaining the Knowledge Base which consists of the ontology and the RDF store. The ontology contains the schema and is often stored in a file. The RDF store contains the data annotated with the ontology which are in the format of RDF triples. The ontology and the RDF store are used to build the Ontological Dictionary in order to use it to get answers when building and testing the system. QA system will cover a specific domain of knowledge. We built an Ontology to represent that domain, and only questions targeting that domain will be answered. While we will rely on a specific domain of knowledge, this will be as a proof of concept, and what applies to a specific domain can apply to any other domain.

Question Processing: This component processes natural language query through five steps:

- Normalization: The process of transforming texts into a single canonical form that it might not have had before.
- Tokenizing: The tokenizer divides the user question into its separate words.
- Removing the stop words.
- Stemming: The process of segmenting and separating affixes from a stem to produce prefix, stem and suffix parts for each word.
- Tagging: We use tagger in order to determine the type of a word, verb or noun and obtain its root.

Ontology Mapper: In this component, we map between words in user's questions with all ontology terms (represents concepts/classes, instances/individuals, relations/properties in Ontological Dictionary).

Answer retrieval: In this component, we define set of rules and procedures to generate a SPARQL query from the NL query based on dependencies and complexity between the query terms.

- **Implementation of the model**: We integrate between techniques from ontology development and Natural Language Processing using java programming.
- **Testing the model**: We have tested 30 questions in the field of ontology domain.
- **Evaluation:** The main goal of our system is to get answers for questions by transforming NL queries in Arabic into SPARQL queries. So in the evaluation phase, we verify that the goal has been performed according to the rules and procedures that we have set in our thesis.
- **Results and discussions**: In this phase, we analyze the obtained results and justify our model feasibility.

1.6 Thesis Structure

The thesis consists of seven chapters organized around the objectives of the research.

Chapter 1 introduces the research, problem, objectives, importance, scope and limitations, and methodology followed in the work**.**

Chapter 2 focuses on the background and theoretical concepts related to the QA systems.

Chapter 3 is a literature survey of the Question Answering Systems and current approaches.

Chapter 4 presents the steps to execute the methodology and the architecture of the proposed model.

Chapter 5 presents approach of interpretation NL query into SPARQL.

Chapter 6 is devoted to the experiments, evaluation of the proposed model and discussing the results.

Chapter 7 concludes the thesis and states future work.

1.7 Summary

In this chapter, we have introduced the thesis by giving an introduction about question answering techniques and the terminologies related to it. We stated the research problem and the possibility of using ontology and NLP in the process of development Arabic question answering system. The main objective of this research is to build ontology based Arabic Question Answering system that transforms NL queries in Arabic into SPARQL queries. Additionally, the importance of this research is explained, which considers the system as a basis for designing more specialized and more advanced systems for answering questions in Arabic. We also stated the scope and limitation of this research. The limitation is that the system model depends on specific domain ontology, and we build a system prototype not a complete system. We presented the methodology that will be followed in this research including designing the model, implementing a prototype of the proposed model, testing and evaluating. In the sixth section, the thesis structure has been explained.

Chapter 2: Background

This chapter presents the background and theoretical concepts of the Natural Language Processing (NLP), NLP in the Arabic language, Question Answering (QA) system, Arabic QA, ontology development, RDF, RDFS and SPARQL.

2.1 Natural Language Processing (NLP)

Researchers in many fields of computer science are interested in making computer interfaces that are easier to use for people, since people will (hopefully) be able to talk to the computer in their own language, rather than learn a specialized language of computer commands. Natural language processing systems represent one of the most important fields of investigation to serve this interest [11].

Natural Language processing is a field of Artificial Intelligence and linguistic, and it is primarily concerned on interaction between computer and human language in terms of theoretical results and practical applications and on information sharing , now that information is exchanged as it never has been before and sharing information becoming the leading theme in the domain of NLP systems [12].

Morphological analysis techniques form the basis of most natural language processing systems. Such techniques are very useful for many applications, such as information retrieval, text categorization, dictionary automation, text compression, data encryption, automatic translation and computer-aided instruction. There is a couple of open source natural language processing software, such as OpenNLP, Stanford parser etc. [13].

2.2 Natural Language Processing (NLP) in the Arabic language

Arabic language is currently the sixth most widely spoken language in the world. It is the mother tongue of about 300 million of peoples. Arabic is an official language in more than 22 countries. Since it is also the language of religious instruction in Islam, many more speakers have at least a passive knowledge of the language. The direction of writing is from right-to left, and the Arabic alphabet consists of 28 letters. Most Arabic words are morphologically derived from a list of roots; most of these roots are three constants [14].

The Arabic language differs from other natural languages such as English language, its own features that are not found in other languages. NLP in the Arabic language is still in its initial stage compared to the work in the English language, which has already benefited from the extensive research in this area. There are some aspects that slow

down progress in Arabic NLP compared to the accomplishments in English and European languages [15]. These aspects include:

- The absence of diacritics in the written text creates ambiguity and therefore, complex morphological rules are required to identify the tokens and parse the text.
- The direction of the writing of the script is from right to left and some of the characters change their shapes based on their location in the word.
- Capital letters are not used in Arabic, which makes it hard to identify proper names, abbreviations., and this creates increased ambiguity and especially complicates such tasks as Information Extraction in general and Named Entity Recognition in particular.
- The major difference is that Arabic is mainly highly inflectional and derivational, which makes morphological analysis a very complex task while English and other languages are concatenate .

In addition to the above linguistic issues, there is also a lack of Arabic corpora, lexicons, and machine-readable dictionaries, which are essential to advance research in different areas.

The importance of Arabic language processing tools has dramatically increased in the last decade because of the huge increment of Arabic digital content on the internet, and in internet users who speak Arabic. This fact increases the importance of creating language processing tools that can process this content, and interact with these users in better ways. Morphological analysis is an important step in Arabic language processing because of the complex morphological structure of Arabic where we have infixes along with prefixes and suffixes. In addition, each prefix or suffix may have its own syntactical tag; this means that we have to use the result of the morphological analysis stage in higher stages of Arabic processing like POS-tagging, syntactical analysis [16] .

The **stemming algorithm** is a computational process that gathers all words that share the same stem and have some semantic relation . The main objective of the stemming process is to remove all possible affixes and thus reduce the word to its stem. It is normally used for document matching and classification by using it to convert all likely forms of a word in the input document to the form in a reference document [13].

Arabic stemming algorithms can be classified, according to the desired level of analysis, as either stem-based or root-based algorithms. Stem-based algorithms, remove prefixes and suffixes from Arabic words, while root-based algorithms reduce stems to

roots . Light stemming refers to the process of stripping off a small set of prefixes and/or suffixes without trying to deal with infixes or recognize patterns and find roots [17].

Khoja and Garside [18] developed an effective stemmer depending on simpler linguistic rules, this approach (1) removes prefixes and suffixes, then (2) matches the remaining word against the patterns to extract the root, and finally (3) checks whether the extracted root is a valid root using an Arabic roots dictionary. Khoja stemmer is considered as a high performance stemmer.

Arabic grammarians traditionally analyze all Arabic words into three main parts-of-speech. The three major **part of speech** categories in the Arabic language i.e. nouns, verbs and particles (in Arabic, Ism (اسم), Fi'l (فعل) and Harf (حرف) respectively) , Figure 2.1 shows major part of speech categories [19].

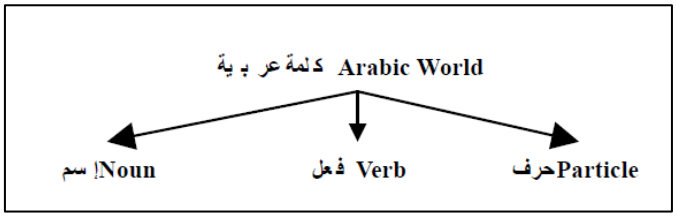


Figure 2.1: A Classification of Arabic Words According to The Part of Speech

Particle in Arabic is voice-based segment of excerpts of throat or tongue or lips. Such as: on ,in ,of (على ، في ، من). The Particle class include: prepositions, adverbs, Conjunctions, and interjections. Verb is a word that indicates an action or state with being connected with notion of time. Verb is divided into three Classes: Past tense (فعل ماضي), present tense (فعل حاضر), and ordered tense (فعل امر), such as: (قال، يقول ، قل). Noun or ism is a word that indicates meaning by itself without being connected with the notion of time, and that describes a person, location, or idea. Such as (Ali, Maca, and Bird), in Arabic (عصفور ، مكة ، علي) [14].

Tokenization is very important in natural language processing. It can be seen as a preparation stage for all other natural language processing tasks. Tokenization is the task of separating out words (morphemes) from running text. It (also sometimes called segmentation) refers to the division of a word into clusters of consecutive morphemes, one of which typically corresponds to the word stem, usually including inflectional morphemes. We can use blanks (white space) to tokenizes the sentence/input sequence on the basis of whitespaces, but there are hard cases. This definition is for English language but for Arabic the situation is deferent. In discussing tokenization, it is important to remember that there is no single optimal tokenization. What is optimal for

IR may not be true for SMT. Also, what is optimal for a specific SMT implementation may not be the same for another [20].

2.3 Question Answering (QA) system

In current years, there has been a marked increase in the amount of information available on the Internet. Users often have specific questions in their mind, for which they expect to discovering out the answers. They would like to find out the answers to be short and precise, and they always prefer to express the questions in their native language without being restricted to a particular query language, query formation rules, or even a particular knowledge domain. Question Answering system give the ability to answer questions posed by a human in natural language by extracting from a repository of documents, fragments of documents that contain material relevant to the answer [21].

The user of a question answering system is interested in a concise, comprehensible and correct answer, which may refer to a word, sentence, paragraph, image, audio fragment, or an entire document. Question answering is multidisciplinary. It involves information technology, artificial intelligence, natural language processing, knowledge and database management and cognitive science. From the technological perspective, question answering uses natural language processing, information retrieval, and knowledge representation [22].

The Question Answering system plays a major role in current era. It is needed when the user gets an in depth knowledge in a particular domain. QA system is classified as two types namely *closed domain* or *restricted domain* and *open domain* model. *Open domain* question answering deals with questions about nearly everything and can only rely on general ontology and world knowledge. On the other hand, these systems usually have much more data available from which to extract the answer. *Closed-domain* question answering deals with questions under a specific domain (for example medicine or weather forecasting and etc.) and can be seen as an easier task because NLP systems can exploit domain-specific knowledge frequently formalized in ontology. Alternatively, open-domain might refer to a situation where unlimited type of questions are accepted, such as questions asking for descriptive [3].

The QA competitions focus on open-domain systems that can potentially answer any generic question. In contrast, a QA system working on a specific technical domain can make use of the specific domain-dependent terminology to recognize the true meaning included in a segment of natural language text [7].

In QA systems two types of search is available namely keywords based search and semantic search. Normal search engines are working under keyword based

searching concept. But some time, there is a problem of getting wrong answer for different meaning of same word. So, semantic search is used to solve the above problem. Semantic search is used to improve the accuracy of search by understanding the intent of the user and the meaning of the terms in the searching sentence. Semantic Search uses semantics to produce highly relevant searching results. This Semantic Search technique can be used to retrieve the knowledge from the data source like ontology. Ontology is a technology used to enable the domain knowledge at a high level and improve the query time used in Question Answering system [23].

The QA task is a complex and challenging task both for building a QA system and for evaluating it. TREC4 (Text REtrieval Conference) and CLEF5 (Cross Language Evaluation Forum) are two international competitions allowing to the researchers in this area to compare their systems. Both Monolingual and Cross-Lingual QA tasks were organized in these competitions [24].

In general, Question Answering systems (QAS) consists of three distinct phases: question processing, document processing and answer processing. Question processing phase plays an important part in QA systems. If this phase doesn't work correctly, it will make problems for other sections. Moreover answer processing stage is an emerging topic in Question Answering, in which these systems are often required to rank and validate candidate answers. Most Question Answering systems follow these three phases; however they may differ in how they implement every phase. These techniques aiming at discovering the short and precise answers [25].

2.4 Arabic Question Answering

The goal of QA systems, as defined by Hirschman and Gaizauskas [26], is to allow users to ask questions in Natural Language (NL), using their own terminology, and receive a concise answer. Arabic is a rich language which adds many challenges to the problem of Arabic QA. Arabic is highly inflectional and derivational which causes sparseness of terms in Arabic text, leading to the inefficiency of many statistical IR and NLP techniques. The absence of diacritics in Modern Standard Arabic (MSA) also adds a lot of ambiguity to Question Analysis and Answer Extraction. Unlike English and other Latin-based languages, Arabic has no capital letters which makes Named Entity Recognition (NER) a lot harder [21].

In the field of QA, English and other Latin-based languages benefited a lot from the advancement in this field. However, Arabic Question Answering has not reached the same level due to the Arabic language specific challenges. In the CLEF and TREC conferences, the participating QA systems were systems performing on many languages (English, French, Spanish, Italian, Dutch, ...) but unfortunately, the Arabic language was

not one of them. However, some efforts were conducted to build QA systems oriented to the Arabic language [1].

Some attempts were made to reach an acceptable result in the Arabic Question Answering task. Most of these attempts suffered from over-fitting and subjective, nonrealistic evaluation. This is due to the scarceness of Arabic language resources, corpora, and test-beds [21]. Among the earliest attempts to tackle the Arabic Question answering problem was AQAS, an Arabic QA System developed by Mohamed et al. 1993. AQAS used a knowledge-based model that can only search for answers in structured data [27]. From 1993 till 2002 much advancement in the field of Arabic NLP and IR were done that led to the creation of QARAB which was used with unstructured documents written in Arabic for Al-Raya newspaper in Qatar [15]. Nevertheless, QARAB evaluation was biased as it only used 113 factoid questions as a testbed and QARAB creators themselves were the evaluating users for the system that they created. It is also skeptical that the results that they reached in their evaluation were much higher than the state of the art work done in English Question Answering [19].

However, as we mentioned above, there are no Arabic QA tasks which provide a test-bed allowing a general test for any Arabic QA system, so the reliability of the reported results keep on being very low since such precision and recall were not achieved in any other language [24].

2.5 RDF and RDFS

Resource Description Framework (RDF) is a framework for representing information about resources in a graph form. Since it was primarily intended for representing metadata about WWW resources, it is built around resources with Uniform Resource Identifier URI. RDF documents are written in XML, the XML language used by RDF is called RDF/XML. By using XML, RDF information can easily be exchanged between different types of computers using different types of operating systems and application languages. Information is represented by triples *subject-predicate-object* in RDF. All of the elements of the triple are resources with the exception of the last element, object, that can be also a literal. Literal in the RDF sense is a constant string value such as string or number [8] .

Many ontologies exist for RDF. They are usually defined using the Web Ontology Language (OWL) or, in a simpler fashion, using the RDF Schema (RDFS) system. RDF Schema is a semantic extension of RDF. It provides mechanisms for describing groups of related resources and the relationships between these resources. In RDFS, predefined Web resources rdfs: Class, rdfs: Resource, and rdf: Property can be used to declare classes, resources, and properties respectively These resources are used

to determine characteristics of other resources, such as the domains and ranges of properties [28].

2.6 Ontology

The term ontology can be defined in many different ways. Genesereth and Nilsson defined Ontology as an explicit specification of a set of objects, concepts, and other entities that are presumed to exist in some area of interest and the relationships that hold them. Ontology is a shared conceptualization with a clear hierarchy and a strong support for logical consequences. Ontology contains a set of specific and clearly described classes or concepts, property of the concepts, slot, restriction, facet and a series of instance related to one class, which combines to form the knowledge storage. Class is the core of ontology, which describes the concepts in some domain. Slot describes the property of the class and the instance [29].

Ontologies play an increasingly important role in knowledge management and is used as a standard knowledge representation for the Semantic Web. By ontology the users can connect with each other using a common understanding of a domain. This helps in understanding the concepts of the domain as well as helps the machine to interpret the definitions of concepts in the domains and also the relations between them [30]. For Accessing structured data in the form of ontologies requires training and learning formal query languages (e.g., SeRQL or SPARQL) which poses significant difficulties for non-expert users. Tools for creating, editing and querying ontologies are widely developed to date. However, an initial barrier for using these tools is in the required background knowledge of the field [31].

- **Ontology Building Methodologies**

Ontology building is not a simple task; it needs time, effort and expertise in the domain in which we want to build the ontology. A team of people, such as domain experts and ontological engineers, normally carries out the development of ontology. For manual construction of ontology we follow mainly Noy and McGuinnes methodology [6]. It includes the following:

1. Determine the domain and scope of the ontology

2. Consider reusing existing ontologies

3. Enumerate important terms in the ontology

4. Define the classes and the class hierarchy

5. Define the properties of classes-slots

6. Define the facets of the slots

7. Create instances

Step 1. Determine the domain and scope of the ontology

This step defines the domain and the purpose of the ontology. Developing an ontology is not an aim or a goal in itself but we build the ontology for a particular purpose. This stage includes multiple and basic questions to be answered: what is the domain that the ontology will cover? For what we are going to use the ontology? For what types of questions should the ontology provide answers? Who will use and maintain the ontology? [32].

Step 2. Consider reusing existing ontologies

This step is to ascertain if ontology has developed previously in the same subject area. If such ontology exists, it is easier to modify the existing ontology to suit ones needs than to create a new ontology [32]. Reusing existing ontology may be required if our system needs to interact with other applications that have been committed to a specific ontology or controlled vocabulary.

Step 3. Enumerate important terms in the ontology

This is step is consider as the first step or the actual definition of the ontology where we make a list of an expected terms that will be used on the ontology building. It is important to get a comprehensive list of these terms without fear of overlap between concepts they represent or relations among the terms.

Step 4. Define the classes and the class hierarchy

After the identification of the relevant terms, these terms have to be organized in a hierarchical way, There are several possible approaches in developing a class hierarchy: A top-down development process starts with the definition of the most general concepts in the domain and subsequent specialization of the concepts. For example, we can start with creating classes for the general concepts. Then we specialize the class by creating some of its subclasses and so on. A bottom-up development process starts with the definition of the most specific classes then leaves of the hierarchy with subsequent grouping of these classes into more general concepts. For example, we start by defining classes then we create a common superclass for these classes. A combination development process of the top-down and bottom up approaches is also possible [6].

Step 5. Define the properties of class-slots

In this step, the classes that are created in the previous step does not provide enough information alone. So once we have selected the defined classes in the list of terms we created in Step 3, most of the remaining terms consider properties (slots) of these classes [6]. Where for each property in the list, we have to show which class it describes.

Step 6. Define the facets of the slots

In this step we add facets to the properties these facets include value type, allowed values, the number of the values (cardinality), and other features of the values the slot can take.

Step 7. Create instances

The last step is creating individuals (instances) of classes in the hierarchy, defining an instance of a class requires: (1) choosing a class, (2) creating an instance of that class, and (3) filling in the slot values.

2.7 SPARQL

SPARQL (pronounced "sparkle", an acronym for SPARQL Protocol and RDF Query Language) is an RDF query language, that is, a query language for databases, able to retrieve and manipulate data stored in RDF format [33].

SPARQL is the standardized query language for RDF, the same way SQL is the standardized query language for relational databases. There are some similarities because it shares several keywords such as SELECT, WHERE, etc. It also has new keywords that you have never seen in a SQL world such as OPTIONAL, FILTER and much more [9].

SPARQL is powerful, flexible, and allows the use of RDF, with all of its advantages over traditional databases. However, SPARQL query construction has been described as “absurdly difficult”, and even experienced users may struggle with it. For this reason, various methods have been suggested for aiding in SPARQL query generation, including assisted query construction [34].

SPARQL is built upon the concept of triple pattern, which is written as subject, predicate, and object, and has to be terminated with a full stop. A SPARQL triple pattern can include variables: any or all of the subject, predicate, and object values in a triple pattern can be a variable [30].

2.8 Summary

In this chapter, we have presented a background for this research. We discussed the concept of NLP and its technologies including tokenization, stemming, POS and the importance of them in the field of Question Answering system. We explained the difference between Arabic and English NLP. Additionally, we defined the ontology and explained the steps that must be followed to build it. We also identified the terminology of RDF, RDFS and SPARQL.

Chapter 3: Related Work

In this chapter, different related works are studied and investigated. The related works introduced and analyzed Question Answering Systems. Most of presented related works in English language because for Arabic language there is a few published work in this field.

3.1 QA in English language

A plenty of research has explored question answering systems using different approaches. However, this section focuses only on ontology- based approaches.

Querix [35]: is an ontology-based question answering system which relies on clarification dialogs in case of ambiguities. This system contains user interface, ontology manager, query analyzer, matching center, query generator, dialog component and ontology access layer. NL queries are converted into SPARQL query form and using Wordnet the synonym is identified. Standford parser is also used in this system which provides a syntax tree for NL query. Querix doesn't exploit the logic based semantic techniques.

PANTO [36]: is a Portable nAtural laNguage inTerface to Ontologies which accepts input as natural language form and the output is in SPARQL query. It is based on triple-based model in which parse tree is constructed for the data model using the off-the-shelf Standford parser. Logic rules are applied for natural language queries as negation, comparative and superlative form. For mapping WordNet and String metric algorithms are used. The parse tree forms the intermediate representation as Query Triples Form. Then PANTO converts Query Triples form into OntoTriples form which are represents as entities in ontology. OntoTriples are finally interpreted as SPARQL form. The performance of PANTO is analyzed by using FMeasure type. At the maximum 88.05% Precision is achieved for Geography domain with tested queries. So this system helps to bridge the gap between the real world users with the semantic web based on logic model.

AquaLog [37] : is An ontology-driven Question Answering system as an interface to the Semantic Web that capable of learning the user's jargon in order to improve his experience by the time. Their learning mechanism is good in a way that it uses ontology reasoning to learn more generic patterns, which could then be reused for the questions with similar context. In this system two major models are used as Linguistic Component which is used to convert the NL questions into Query-triple format and Relation Similarity Service (RSS) which takes Query Triple form into Onto-Triple form. The data model is triple based like <Subject, Predicate, Object> type. The Performance is based

on Precision, Recall and also failure types are referred separately. At average 63.5 % of successive answers are retrieved from ontology with closed domain environment.

ORAKEL [38]: is an ontology-based question answering system that used for computing intentional answers of user query. It computes wh-based questions as logical query form and knowledge is represented with F-Logic and Ontobroker form. This system is used to convert question into query form and the given query is fed to bottom-up generalization model for getting intentional answer to the user. Inference engine is used to evaluate queries to knowledge base form. Customization is performed through the user interaction, using software called Frame Mapper, where the linguistic argument structures, such as verbs or nouns with their arguments, are mapped to the relations in the ontology.

QACID [39] : is an ontology-based Question Answering system applied to the CInema Domain that relies on the ontology, a collection of user queries, and an entailment engine that associates new queries to a cluster of existing queries. Each query is considered as a bag of words, the mapping between words in NL queries to instances in a KB is done through string distance metrics and an ontological lexicon. Prior to launching the corresponding SPARQL query for the cluster, the SPARQL generator replaces the ontology concepts with the instances mapped for the original NL query. This system is at the domain-specific end of the spectrum because the performance depends on the variety of questions collected in the domain, the process is domain-dependent, costly and can only be applied to domains with limited coverage.

QuestIO [40]: In this system NL queries are translated into formal queries but the system is reliant on the use of gazetteers initialized for the domain ontology. In QuestIO users can enter queries of any length and form. QuestIO works by recognizing concepts inside the query through the gazetteers, without relying on other words in the query. It analyzes potential relations between concept pairs and ranks them according to string similarity meas-ures, the specifity of the property or distance between terms. QuestIO supports conjunction and disjunction.

FREyA [41]: a Feedback Refinement and Extended Vocabulary Aggregation system is the successor to QuestIO and combines syntactic parsing with knowledge in ontology for reducing customization effort. The rules are not used in this system instead of that knowledge encoded in ontology is given for understanding the user's question. Then the syntactic parsing is used to get a precise answer. In this model the ontology concepts are identified and verified initially. Then the SPARQL query is generated and the answer type is identified. Syntactic parse tree is generated using Stanford Parser. Mapping of user query with ontology concept is implemented in two ways as automatically and by

the help of user. Ranking model is used by string similarity type. Answer type of this system is in graph form. JIT library is used for graph visualization. The Precision and Recall value for the tested data is reached high as 92.4% this is achieved only by the system returns the answers always as correct/partial/incorrect form. For measuring the performance of the system the Mean Reciprocal Rank (MRR) algorithm is implemented. It is a statistic for evaluation the process to a query. MRR value is achieved to 0.81. It supports high precision and recall.

3.2 QA in Arabic language

AQAS (1993) [27]: is a knowledge-based QA system that extracts answers only from structured data and not from raw text (not structured text written in natural language);to our knowledge, no evaluation results have been published for AQAS system.

QARAB (2004) [15]: is an Arabic QA system that uses Information Retrieval (IR) and Natural Language Processing (NLP) techniques. QARAB is first treats the incoming question as a *"bag of words"* against which the index file is searched to obtain a list of ranked documents that possibly contain the answer. The question processing begins by performing tokenization to extract individual terms. Then, the stop-words are removed. The remaining words are tagged for part-of-speech in an attempt to highlight the main words that should appear in the hypothesized answer. Then the set of relevant documents that may contain the answer are retrieved by the IR system. The answer exists in unstructured documents written in Arabic for Al-Raya newspaper in Qatar. In QARAB system great effort should be spent on identifying proper names to identify the possible answer. Also it uses shallow language understanding to process questions and it does not attempt to understand the content of the question at a deep, semantic level. QARAB system reached a precision of 97.3% and also a recall of 97.3%. The evaluation was done directly by four native Arabic speakers who presented 113 questions to the system and judged themselves the correctness of the answers.

ArabiQA (2007) [1]: is an Arabic QA system that employed the Java Information Retrieval System (JIRS), a passage retrieval system to search the relevant passages .Also it uses an Arabic Named Entity Recognition (NER) system called ANERsys to identify and classify named entities within the passages retrieved. The test-set consists of 200 questions and 11,000 documents from Wikipedia Arabic version. They reached a precision of 83.3%.

QASAL (2009) [42]: is an Question Answering system for Arabic Language that developed a prototype to build an Arabic factual Question Answering system using Nooj platform to identify answers from a set of education books. NooJ is a linguistic environment that includes large-coverage dictionaries and grammars, and parses corpora

in real time. It includes tools to create and maintain lexical resources, as well as morphological and syntactic grammars. Dictionaries and grammars are applied to texts in order to locate morphological, lexical and syntactic patterns and tag simple and compound words. QASAL consist of three components: Question analysis, Passage retrieval, and Answer extraction module . In Question analysis: First, they apply the set of linguistic resources to the input question in order to annotate it. Next, they carry out a question analysis step based on some local grammars built as a set of Augmented Transition Networks (ATNs) using the NooJ's graph Editor. These grammars could translate each question into one or more Regular Expressions (RegEx) in order to represent its corresponding answer pattern. In Passage retrieval: The first task of this step could be the selection of one or more regular expressions between those automatically generated. After that, these expressions are applied to the answer text in order to identify the potential answer. In Answer Extraction: this last step uses the displayed concordance table to automatically extract the answer of the input question.

Arabic QAS (2009) [43]: is an a new Arabic QA system (QAS) using techniques from IR and NLP to answer short questions in Arabic. Similar to QARAB, QAS excludes the questions "ماذا , كيف"(How and Why) citing the same reason cited by Hammo et al. [7] Both stated that (How and Why) " require long and complex processing." Furthermore, the authors reported a test reference collection consisting of 25 documents gathered from the Internet, 12 queries (questions) and some relevant documents provided by the authors. Kanaan et al. reported different recall levels (0, 10 and 20%) where the interpolated precision was equal to 100% and at recall levels 90 and 100% it was equal to 43%. In VSM a document is conceptually represented by a vector of keywords extracted from the document, with associated weights representing the importance of the keywords in the document and within the whole document collection. Similarly, a query is modeled as a list of keywords with associated weights representing the importance of the keywords in the query.

DefArabicQA (2010) [10]: is an Arabic Definition Question Answering System that tackled the definition type of questions. They first identified the candidate definitions using manual lexical patterns of sequence of words, letters and punctuation symbols. Then they used some heuristic rules that they deduced from observing the form of some correct and incorrect definitions. After they extracted the candidate definitions, they ranked them according to three criteria which are Pattern weight of the pattern that matched the candidate definition and Snippet position of the snippet that contains the candidate definition in the snippets collection and the sum of word frequencies in the candidate definition. However, their evaluation was not good enough as they tested on only 50 organization definition questions and the answers were assessed by only one Arabic speaker .

AQuASys (2011) [44]: a stand-alone Arabic QA system for factoid questions. It extensively utilizes NLP techniques to analyses questions and retrieves answers from an Arabic corpus that has been developed by the authors. Retrieved answers are scored and presented based on their relevance.

Al-Bayan (2014) [45]: An Arabic Question Answering System specialized for the Holy Quran. The system accepts an Arabic question about the Quran, retrieves the most relevant Quran verses, then extracts the passage that contains the answer from the Quran and its interpretation books (Tafseer). Evaluation results on a collected dataset show that the overall system can achieve 85% accuracy using the top-3 results.

JAWEB (2014) [46]: An Arabic web-based question answering system. It is a web-based application, so it can be accessed at any time and from anywhere. Evaluating JAWEB showed that it gives the correct answer with 100% recall and 80% precision on average. When compared to ask.com, the well-established web-based QA system, JAWEB provided 15-20% higher recall. These promising results give clear evidence that JAWEB has great potential as a QA platform and is much needed by Arabic-speaking Internet users across the world.

As noticed from the above systems, the first and almost only attempt for restricted domain Arabic question answering was AQAS, an Arabic QA system that was specialized in the restricted domain of radiation and its effects [27]. However, this system was more or less a natural language interface to a structured database of frames about this area. Almost all of the systems that followed after that were open domain QA systems that were either used to search a group of documents from a newspaper for example [15] or used to search the Internet as an open domain. Research in restricted domain QA makes semantic tasks like word sense disambiguation easier. It is also very common to find domain rules affecting how the question is posed and how the answer is formulated.

Working on the field of Arabic QA is very limited which mandates a deep look in the future trends of this area. Most of researches in the field of Arabic QA is focused on open domain question answering while very few attempts approached restricted domain question answering. It is important to give more attention to semantics in approach Arabic QA where most research concentrated on the morphological and syntactic aspects of Arabic. The ontology based reasoning can help in translating the human question, identify the key terms and the relationships between them and determine related answers if there is no matching. These benefits are difficult to achieve by using the syntactic approach.

3.3 Summary

In this chapter we presented an overview about question answering system. This chapter divided into two sections. In the first section we focuses only on ontology-based approaches in question answering system for English language because a plenty of research has addressed question answering systems using different approaches. In the second section we studied Arabic question answering system. Research and development in the area of Arabic QA is a few and not of high quality compared to similar work on English systems.

Chapter 4: Arabic Question Answering System (AQAS)

4.1 Overview

This chapter discusses the design of our Arabic question answering system. Our proposed system is based on an Ontology which models the target domain of knowledge. The core of our system is the approach we propose to transform NL queries in Arabic to SPARQL queries. The proposed system will be described using flowcharts, algorithms, figures and tables.

Our model will be used to allow users to raise questions in natural Arabic language, and the system will return accurate answers to users directly after question analyzing and answer extraction depending on the domain of the Arabic Ontology. By an Arabic ontology we mean an ontology that is built in Arabic language, or that is populated with Arabic translations of English terms. An important feature of our system is that it is designed to be ontology-portable. A thing that means it can be interfaced to any ontology, as long as the ontology has Arabic translation of the terms used.

The process of building Arabic Question Answering System (AQAS) has four components, which are **Data Processing**, **Question Processing**, **Ontology Mapper** and **Answer Retrieval**.. Each component will be explained in a separate section. The main required components are shown in Figure 4.1 and stated below:

1) **Data Processing**

   This component is responsible for maintaining the Knowledge Base which consists of the ontology and the RDF store. The ontology contains the schema and is often stored in a file. The RDF store contains the data annotated with the ontology which are in the format of RDF triples. Note that our system is designed to keep the data separate from the ontology which should remain intact and reusable.

   After creating instance and storing them in RDF store, we apply an ontology reasoner on the ontology and RDF data. This is necessary to identify new relations from existing ones.

   The ontology and the RDF store are used to build the Ontological Dictionary in order to use it to get answers quickly. Rather than referring to the ontology file and the RDF store every time the system needs to answer a NL query. All the contents of the ontology and the RDF store are retrieved, preprocess and stored in the Ontological Dictionary. The contents of the Oncological Dictionary are preprocessed through normalizing, stop word removal, stemming and POS.

2) **Question Processing**

In this component we process a natural language query using normalizing, tokenizing, stop word removal, stemming and part of speech tagging. This makes the question words comparable with the content of the Ontological Dictionary.

3) **Ontology Mapper**

This component is used to match words in the user query with the content of the Ontological Dictionary. Terms that match with the query words will be used later to build the RDF triple patterns of the SPARQL query.

4) **Answer Retrieval**

This is the core component of the system, and it is responsible for translating the NL query to SPARQL. Then it executes the SPARQL query to retrieve results from the RDF store. This will be explained later in the next chapter.

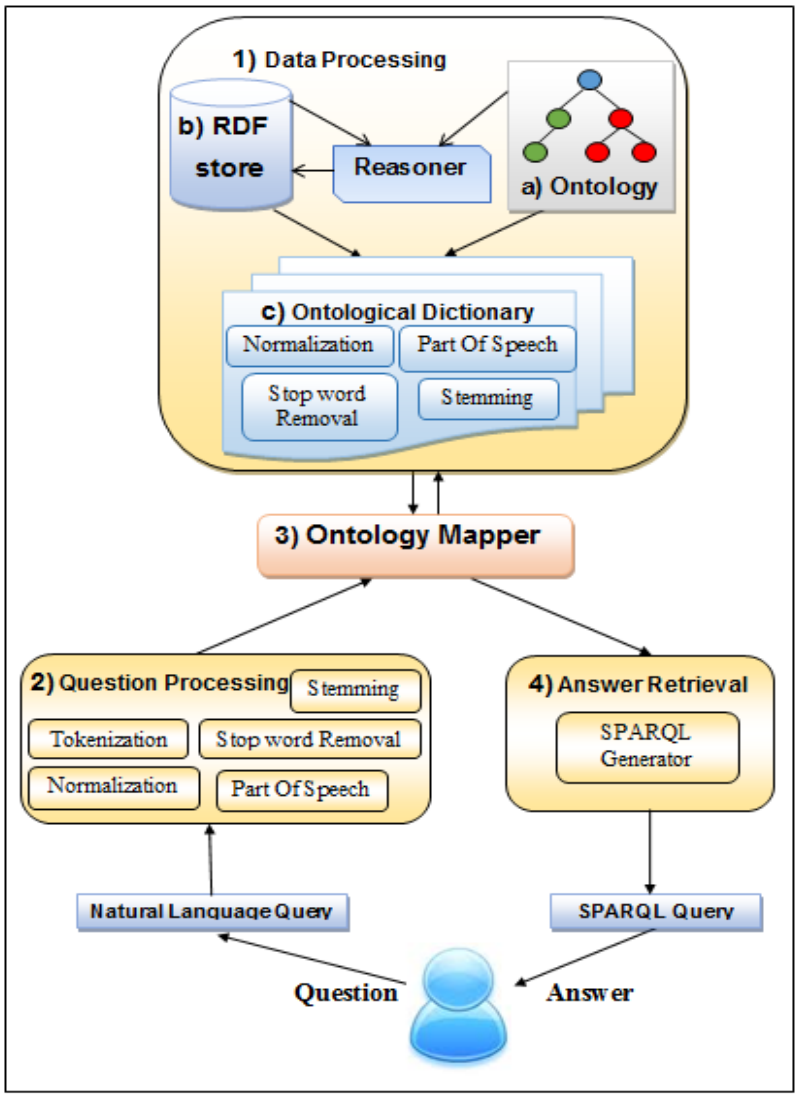


Figure 4.1: Arabic Question Answering (AQAS) Architecture

4.2 Data Processing

4.2.1 Building Arabic Ontology Domain

The proposed Arabic Question Answering system is generic in terms that it should accept any domain ontology represented in Arabic language. Due to the lack of mature ontologies that are built using Arabic language, we built a prototype Ontology that covers a restricted domain of knowledge from the science of Pathology. We used this ontology throughout this thesis to discuss our proposed approach to convert NL queries to SPARQL queries and to demonstrate sample results. However, we emphasize that the system can work with other Arabic ontologies that follow the same constrains.

There are various tools available for developing ontologies like Hozo, DOML, and AltovaSemantic Works etc. We have used Protégé which is one of the most widely used ontology development editor that defines ontology concepts (classes), properties, taxonomies, various restrictions and class instances. It also supports several ontology representation languages, including OWL [29].

Building ontology is an important task in our work. We used a top-down approach [6] to build the ontology, where abstract concepts are identified first, then specialized into more specific concepts. We construct Arabic Ontology Domain " علم الامراض" which represents the basic knowledge in our work manually.

We have developed the ontology contents for Pathology Domain, collected from a number of relevant research papers and documentations of Medical domain. The ontology is implemented with Protégé in OWL format and can be mapped into a database. Experts from the domain also helped us to validate the ontology contents.

The building of ontology consists of the following steps, as shown below:

Step 1: Determining the Domain and Scope of the Ontology

Step 2: Defining Overview of Ontology

Step 3: Defining Classes and Class Hierarchy of Ontology

Step 4: Defining the Properties of Classes (Slots)

Step 5: Defining the Facets of the Slots

Step 6: Creating Instances

Step 7: Apply Ontology Reasoner

Step 1: Determine the Domain and Scope of the Ontology

The definition of ontology domain and scope was considered the first step in ontology development, in which the ontology will be developed in order to answer some basic questions:

1. What is the domain that the ontology will cover?

The domain of the ontology will cover study of disease in general "علم الامراض" (Pathology).

2. What is the use of the ontology?

The ontology is to provide a schema-base of Pathology, which is used in our QA system to retrieve the exact answer to the question.

3. *What types of questions would be answered by the information contained in the ontology?*

The ontology would provide answers to questions relating to Pathology domain and like:

- What are the symptoms of diabetes?, "ما هي أعراض مرض السكري؟".
- What causes thalassemia disease?, "ما أسباب مرض التلاسيميا؟".
- What is the cure of cancer?, "ما هو علاج من السرطان؟".
- How diagnosed polio disease?, "كيف يشخص مرض شلل الأطفال؟".
- what organs that are infected by tuberculosis?, "ما هي الاعضاء التي تصاب بمرض السل؟".
- What are the diseases that can infect the eye?, "ما هي الأمراض التي يمكن أن تصيب العين؟".

4. *Who will use the ontology?*

The ontology will be available to the QA system to provide basic information for questions in the domain of Pathology**.**

Step 2: Overview of Ontology

We designed our ontology from scratch as a prototype model to serve in implementation of our system. We identify some diseases and data that are needed in the process of answering questions in our system. Our ontology was represented in OWL.

Our ontology contains general characteristics. It adds or deletes any information about the diseases easily and without affecting the overall structure of the ontology whether at the level of classes or properties. So it can be reused by other applications interested in the same domain. Figure 4.2 illustrates the core classes of the Pathology ontology as

well as the relationships in between. Table 4.1 depicts number of classes, object properties, and data properties in our ontology.

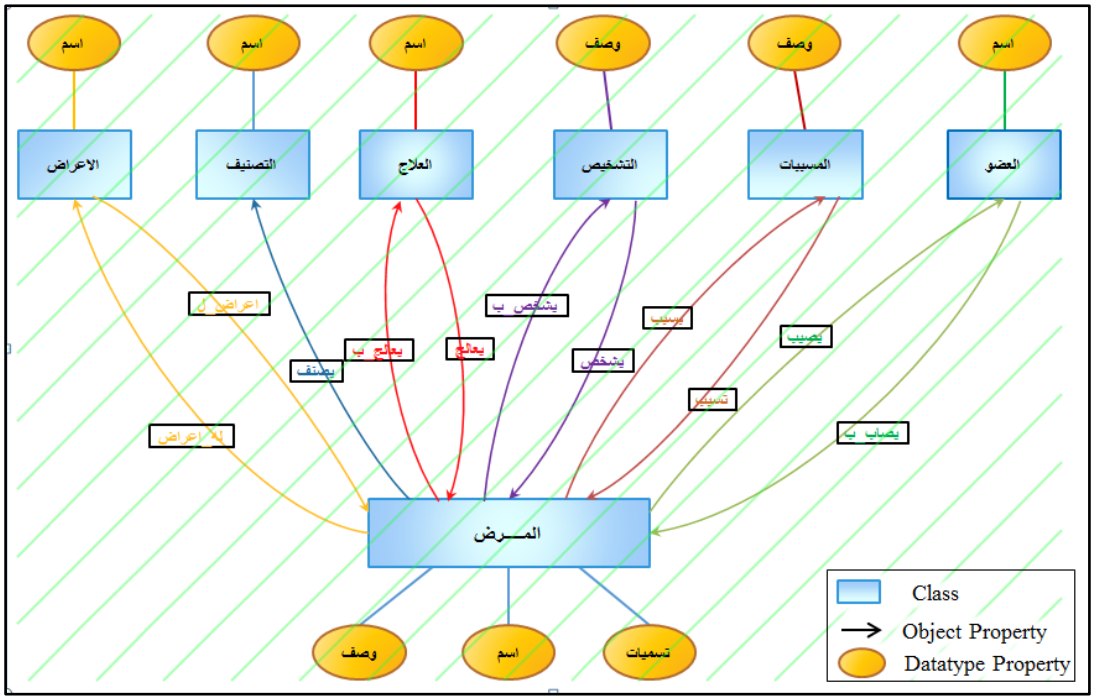


Figure 4.2: Pathology ontology design

Table 4.1: Classes, Object Properties and Data Properties in the ontology

Domain and Scope of the Ontology	Pathology "علم الامراض"
Number of classes	7
Number of object properties	11
Number of data properties	3

Step 3: Define Classes and Class Hierarchy of Ontology

This step defines classes (concepts) used in our ontology domain. These classes are not selected randomly, but they are selected depending on our domain "علم الامراض" (Pathology). Table 4.2 depicts the ontology Classes.

Table 4.2: Ontology Classes

No.	Class /Arabic	Class /English	Description
1	المرض	disease	It is a medical condition associated with specific symptoms and signs
2	المسببات	Reason	factors cause diseases from an external source or internal dysfunctions
3	الاعراض	symptom	It is a departure from normal function or feeling which is noticed by a patient
4	العضو	Organ	It is a collection of tissues joined in a structural unit to serve a common function
5	التشخيص	diagnose	It is the identification of the nature and cause of a certain phenomenon
6	العلاج	Cure	It is the end of a medical condition; the procedure that ends the medical condition
7	التصنيف	Category	It is a branch of medicine that deals with classification of diseases

Table 4.2 contains seven Ontology concepts mentioned in our domain "علم الامراض" (Pathology). Choosing these concepts has direct relation with user requirements used in the process of answering questions in Pathology domain. We mention every ontology concept in Arabic and its synonym in English including the description of the concept.

Figure 4.3 depicts the Ontology Classes in Protégé.

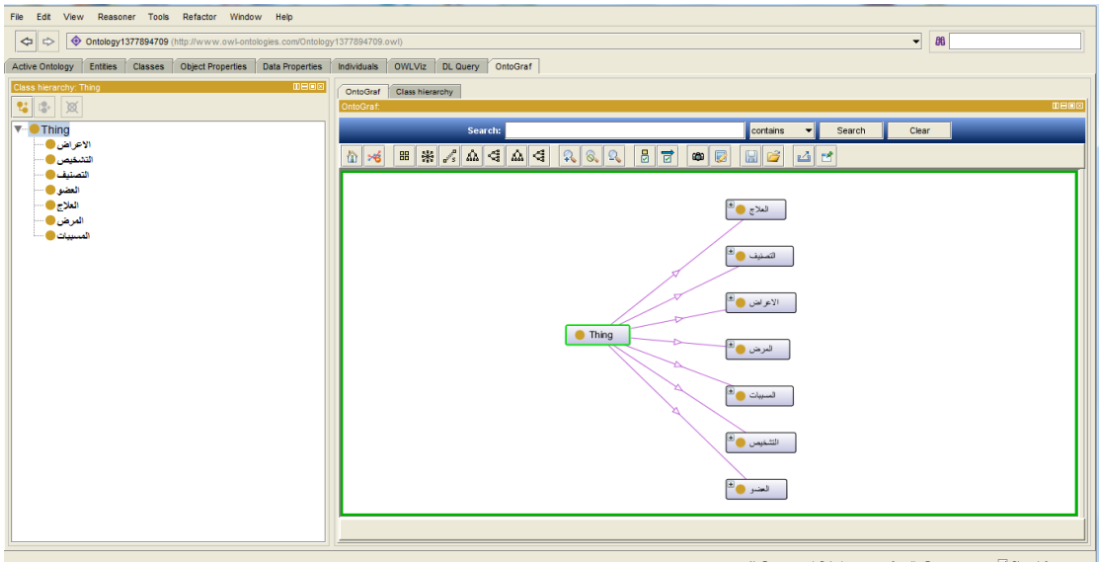
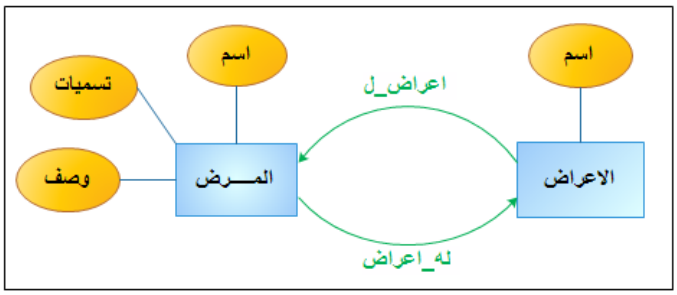


Figure 4.3: Ontology Classes in protégé

Step 4: Define the Properties of Classes (Slots)

Define object properties (relations) among classes as a requirement to come up with the ontology. Creating object properties plays a vital role in connecting classes (concepts) of the ontology in our Arabic Ontology Domain "علم الامراض" (Pathology).

We used 11 object properties that connect the important concepts which have relations with each other that are illustrated in Table 4.3. For Example:



- The class disease (المرض) is connected to another class symptom (الاعراض) using ObjectProperty (له_اعراض).
- The class symptom (الاعراض) is connected to another class disease (المرض) using ObjectProperty (اعراض_ل).
- The class disease (المرض) has three DatatypeProperty which are: hasName "اسم", hasDescription"وصف", hasSynonym"تسميات".
- The class symptom (الاعراض) has one DatatypeProperty which is: hasName "اسم".

Table 4.3: Object properties in Pathology ontology

No.	Object property/AR	Object property/EN	Domain	Range
1	اعراض_ل	symptom_of	الاعراض	المرض
2	له_اعراض	has_symptom	المرض	الاعراض
3	بسبب	caused_by	المرض	المسببات
4	تسبب	Causes	المسببات	المرض
5	يصاب_ب	infected_by	العضو	المرض
6	يصيب	Infects	المرض	العضو
7	يعالج	Cures	العلاج	المرض
8	يعالج_ب	cured_by	المرض	العلاج
9	يشخص	diagnoses	التشخيص	المرض
10	يشخص_ب	diagnosed_by	المرض	التشخيص
11	يصنف	isCategorized	المرض	التصنيف

Table 4.4: Datatype properties in Pathology ontology

No.	Datatype property/AR	Datatype property /EN	Domain	Range
1	اسم	hasName	المرض الاعراض العضو العلاج التصنيف	string
2	تسميات	hasSynonym	المرض	string
3	وصف	hasDescription	المرض التشخيص المسببات	string

Step 5: Define the Facets of the Slots

Slots have different facets that describe the value type, allowed values, the number of the values (cardinality), and other features of the values the slot can take. In our case, all the slot values are string either using ASCII or UTF-8 (Arabic). For example, the value type of data property hasSynonym (تسميات) for domain disease Class (المرض) is string and the number of the values (cardinality) has a minimum cardinality of 1: which means that each disease at least has one Synonym as shown in Figure 4.4. Also, the value type of hasName (اسم) and hasDescription (وصف) are string.

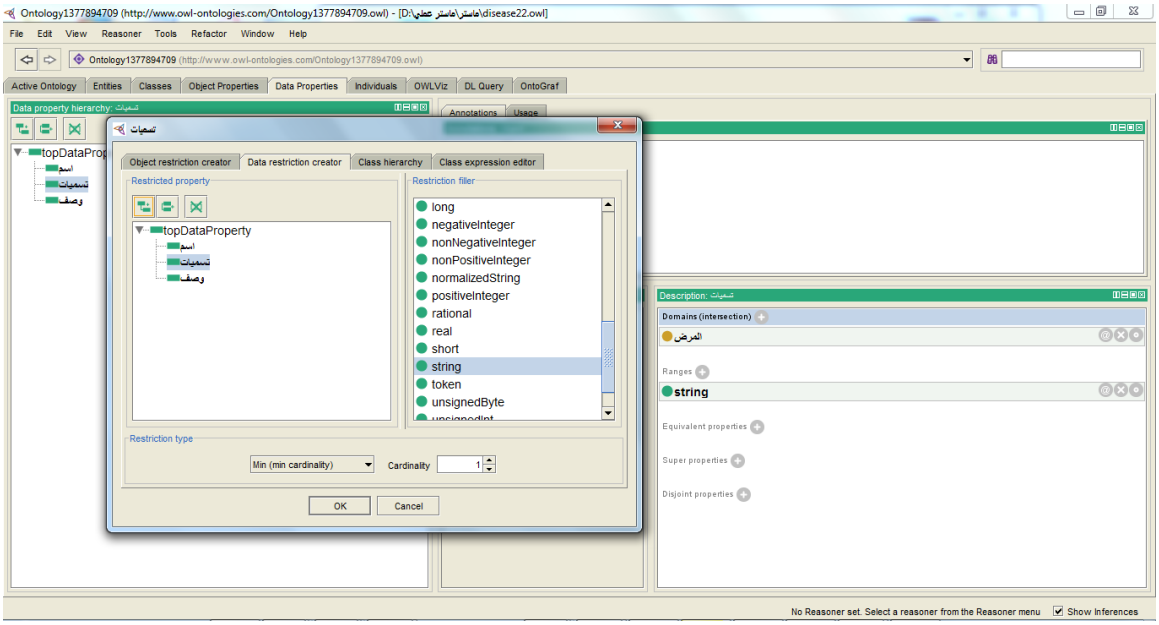


Figure 4.4: Data restriction

Step 6: Create Instances

The last step is creating instances (individuals) of classes in the hierarchy. The creation of individuals allows all the properties of the classes to record. Instances are created and stored in a separate database named RDF store in format of RDF triples. Our system is designed to keep the data separate from the ontology which should remain intact and reusable. The information of individuals is taken from a number of relevant research papers and documentations of medical domain.

In our ontology, we defined around 325 instances representing all ontology concepts. Figure 4.5 depicts some of these instances.

```
//****************************************** Disease 1 ************************************
List<String> disease1Synonyms = new ArrayList<String>();
disease1Synonyms.add("التدرن");
disease1Synonyms.add(" الدرن ");
disease1Synonyms.add(" الاستهلاك ");
Individual disease1 = ddentry.createDisease(dataModel, "السل", "هو مرض معدي جرثومي تسببه عصية الدرن أو عصية كوخ يؤدي لتلف في أنسجة الرئة و/أو أعضاء أخرى من الجسم", disease1Synonyms);

List<String> category1Name = new ArrayList<String>();
category1Name.add("مرض معد");
Individual category1 = ddentry.createCategory(dataModel, category1Name);


List<String> symptom1Name = new ArrayList<String>();
symptom1Name.add("الكحة التي تتصاحب مع الدم");
symptom1Name.add("الام في الصدر");
symptom1Name.add("قصر التنفس");
symptom1Name.add("فقدان الوزن");
symptom1Name.add("الحمى");
Individual symptom1 = ddentry.createSymptom(dataModel, symptom1Name);


List<String> organ1Name = new ArrayList<String>();
organ1Name.add("الفضاء المحيط بالرئتين أو ما يسمى فضاء الجنب");
organ1Name.add("الفضاء المحيط بالقلب أو ما يسمى فضاء التأمور");
organ1Name.add("العقد اللمفاوية");
organ1Name.add("(الأعضاء المحيطة بالمخ (السحايا");
organ1Name.add("الجهاز البولي التناسلي");
organ1Name.add("اقسام الجهاز الهضمي والفضاء المحيط بأحشاء البطن أو ما يسمى الفضاء البريتوني");
Individual organ1 = ddentry.createOrgan(dataModel, organ1Name);
```

Figure 4.5: Ontology Individuals

Figure 4.5 shows part of the instances used in the ontology domain "علم الامراض" (Pathology). In the figure, for example, the class "المرض" (Disease) has instances for its name "السل", description " هو مرض معدي جرثومي تسببه عصبة الدرن يؤدي لتلف في أنسجة الرئة أو أعضاء أخرى من الجسم" and their synonyms "التدرن، الدرن، الاستهلاك". Also, the class "الأعراض" (Symptom) has several instances include "الكحة التي تتصاحب مع الدم", "الحمى", "فقدان الوزن", "الام في الصدر", "قصر في التنفس".

Step 7: Apply Ontology Reasoner

After creating instance and storing them in RDF store, we apply an ontology reasoner (e.g. Jena reasoner) on the ontology and RDF data. This is necessary to identify new relations from existing ones. The reasoner is able to identify the different types of ontological relations such as transitive, symmetric, inverse and functional properties and

use them to add new facts. For example, given the statement ":Diabetes :infects :Pancreas", the reasoner will use the inverse property to add the following new fact: ":Pancreas :isInfectedBy :Diabetes". In another example, given that the property "causes" is a transitive property, and the fact that Diabetes causes high blood pressure, and high blood pressure causes heart disease. The reasoner will infer a new fact which is: Diabetes causes heart disease. The new facts inferred by the reasoner will enable the system to answer questions that cannot be answered otherwise.

Note that we are used the reasoner only to explore new relations and axioms for ontology and RDF data and not used it in the analysis of the question.

4.2.2 Building RDF store to create and store instances data

In our system instances are created and stored in a separate database named RDF store. Storing instances in the database will make the system scalable by keeping all data separate from the ontology. Regardless of the size of instance data, the ontology will remain intact and possibly reusable by other applications interested in the same domain. The other option would be to store instances in the same ontology file but this can be more time consuming and memory intensive when the ontology file becomes huge.

4.2.3 Developing Ontological Dictionary

To enable fast access and matching, all ontological concepts including classes (concepts), properties (relations) and instances (individuals) are extracted, preprocessed and stored in the Ontological Dictionary. So we retrieve all ontology contents with their names and URIs from ontology and RDF store to develop Ontological Dictionary. This dictionary represents a knowledge base for matching the user query with the ontological concepts. Given a word from the user query, the ontological dictionary should output a set of possible matching concepts from the ontology. For example, if the input is " ما علاج مرض السكر", the dictionary will produce the following output:

- علاج => :Cure (Class)
- مرض => :Disease (Class)
- السكر => :Diabetes (instance)

Another example, if the input is "كيف يشخص مرض السرطان", the dictionary will produce the following output:

- يشخص => :Diagnoses (Object property)
- مرض => :Disease (Class)
- السرطان => :Cancer (instance)

Developing the Ontological Dictionary is for the purpose of fulfilling our goal in translation NL queries to SPARQL. Ontological Dictionary contains more than 300 terms from ontology and RDF store that have been selected in pathology domain. Terms can be classes, properties, instances or literals.

All terms in Ontological Dictionary are subject to four main preprocessing operations which are:

A. Normalization: the process of transforming a text into a single canonical form that it might not have had before.
B. Removing stop words.
C. Tagging: we use tagger in order to determine the type of a word, verb or noun and obtain its root.
D. Stemming: the process of segmenting and separating affixes from a stem to produce prefix, stem and suffix parts for each word.

A. Normalization

Text normalization is a process by which a text is transformed in some ways to make it consistent in a way which it might not have been before. All terms and text in Ontological Dictionary are normalized for preprocessing them to map with words in question.

In our proposed system, normalization involves the following steps [47]:

- Remove punctuation
- Remove diacritics (primarily weak vowels)
- Remove non-letters
- Replace initial إ or أ with bare alif ا
- Replace the آ by the ا
- Replace the ىء of order by the ئ
- Replace the ى final by the ي
- Replace the ة final by the ه

Figure 4.6 shows parts of Ontological Dictionary and what happens after normalization.

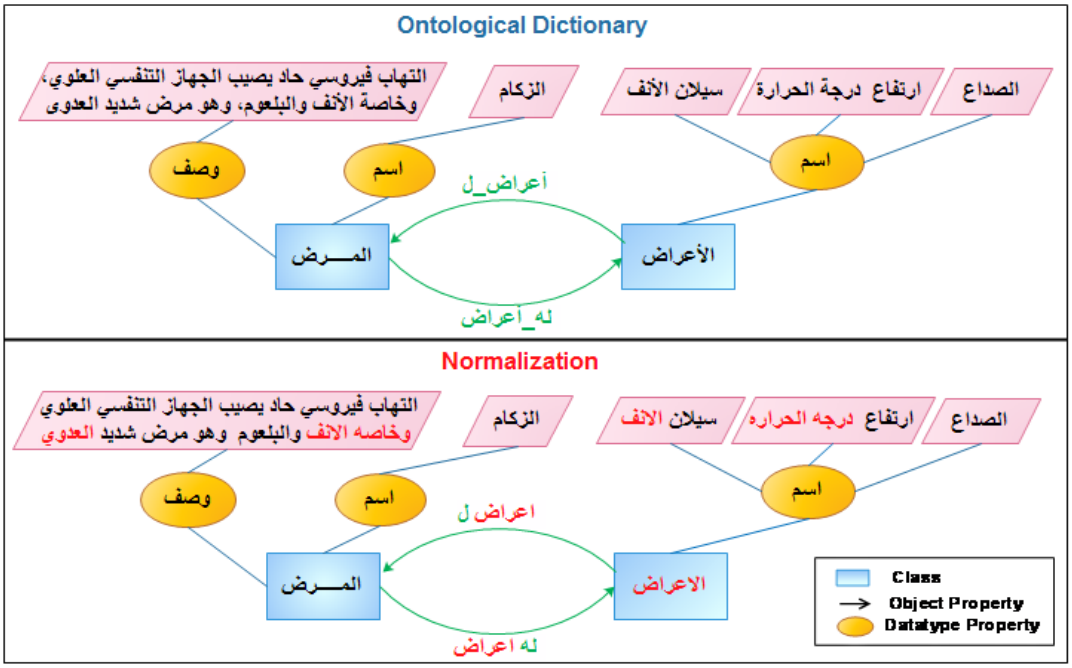


Figure 4.6: Example of normalization in Ontological Dictionary

B. Stop Word Removal

Stop words are those words which rarely contribute to useful information in terms of document relevance and appear frequently in text but provide less meaning in identifying the important content of document.

Those words include prepositions, conjunctions and other high frequency words. Table 4.5 shows some of these words:

Table 4.5: Examples of stop words

ان, بعد, ضد, يلي, الى, ى, في, من, حتى, هو, يكون, به, ليس, أحد, على, وكان, تلك, كذلك, تي, بين, فيها, لم, عليها, إن, على, لكن عن, مساء, لماذا, ليس, ذا, منذ, ذي, أما, حين, من, لا, ليسب, وكانت, أي, ما, عنه, حول, دون, مع, لكنه, لكن, له, هذا, تي, فقط, ثم, هذه, أنه, تكون, قد, بين, جدا, لن, نحو, كان, لهم, لأن, اليوم, لم, هؤلاء, فإن, فيه, ذلك, لو, عند, اللذين, كل, بد, لدى, وثي, أن, مع, فقد, بل, هو, عنها, منه, بها, في, فهو, تحت, لها, أو, إذ, علي, عليه, كما, كيف, هنا, قد, كانت, لذلك, أمام, هناك, قبل, معه, يوم, منها, إلى, إذا, هل, حيث, هي, اذا, او, ما, لا, الي, إلي, مازال, لازال, لايزال, مايزال, اصبح, أصبح, أمسى, امسى, أضحى, اضحى, ظل, مابرح, مافتئ, ماانفك, بات, صار, ليس, إن, كأن, ليت, لعل, لاسيما, لايزال, الحالي, حالي, اول, له, ذات, اي, بدلا, اليها, انه, الذين, ذين, فاء, ان, الذي, هذا, لهذا, الا, فكان, ستكون, مما, , أبو, بإن, الذي, اليه, يمكن, بهذا, لدي, أن, وهي, أبو, آل, الذي, هن, الذى, ام, ابو, عبد, برؤزير, سفير, أستاذ, مهندس, سفير, رئيس, م, ا, ب, ت, ث, ج, ح, خ, د, ر, ز, س, ش, ص, ض, ط, ظ, , ع, غ, ف, ق, ك, ل, ن, ه, و, ي, ى, ء

Figure 4.7 shows part of terms in Ontological Dictionary and what happens after normalization and stop word removal.

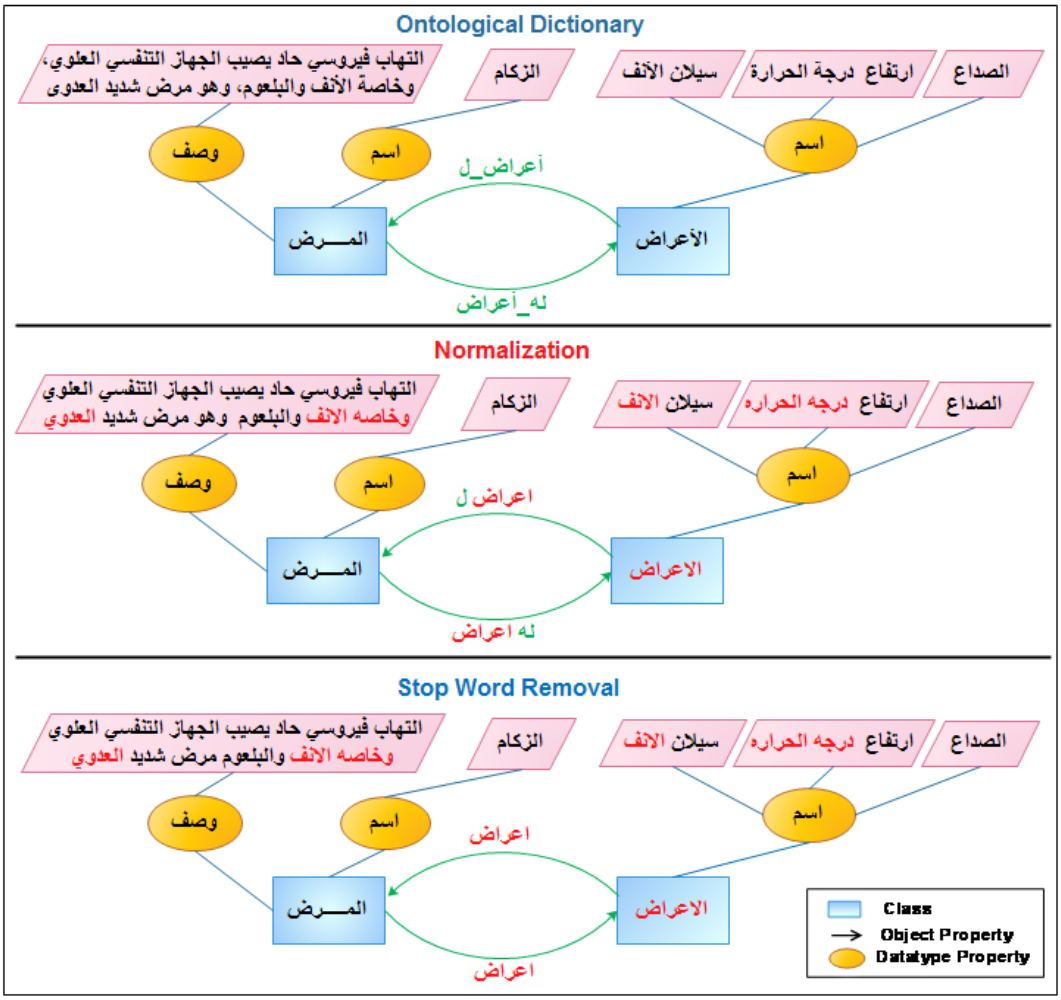


Figure 4.7: Example of stop word removal on terms in Ontological Dictionary

C. Part-of-Speech Tagging (POST)

POST is the process by which a specific tag is assigned to each word of a sentence to indicate the function of that word in the specific context. POS can be Verb, Noun, Proper Noun, Adjective, or Adverb [48].

Arabic POST (APOST) is not an easy task due to the high ambiguity that results from the absence of diacritics and also from the complexity of the Arabic morphology.

We used POS tagger to tokenize the given input sequence with their corresponding word type. This word type is decided on the basis of token itself and on the basis of context of the token in the input character sequence.

The process of assigning a part-of-speech to each term in Ontological Dictionary is important for mapping them with words in user question.

Figure 4.8 shows an example of POST to some words in Ontological Dictionary.

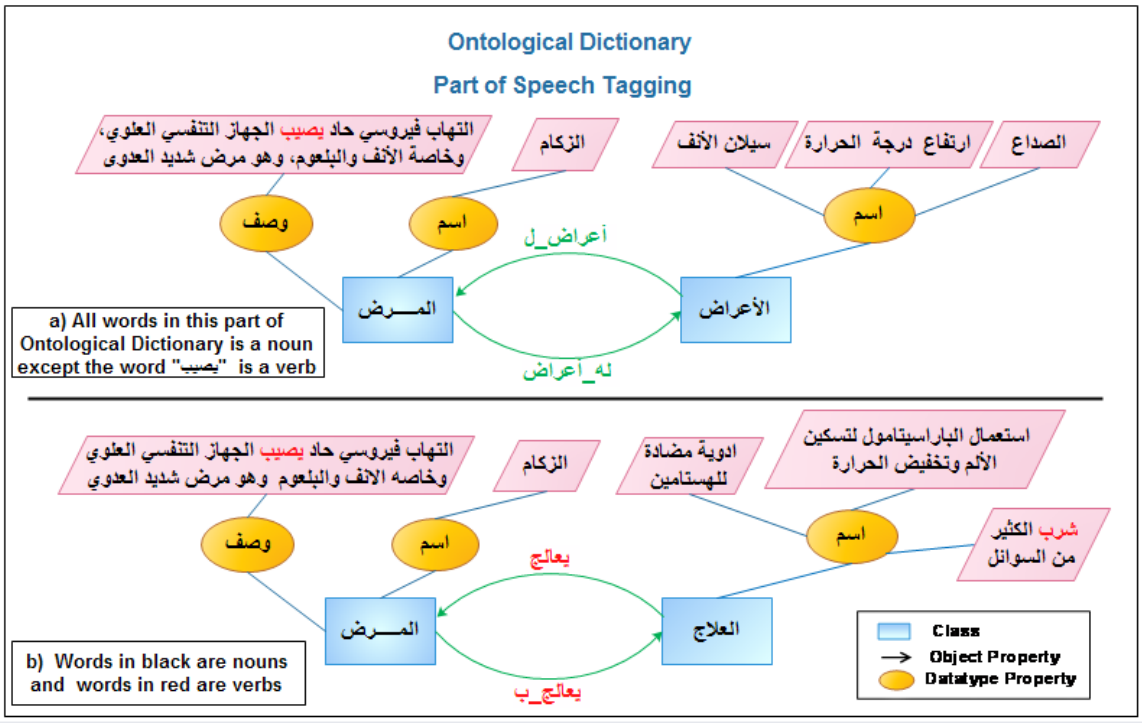


Figure 4.8: Examples of POST in Ontological Dictionary

POS is essential to resolve ambiguity that can result when matching query terms to the ontology content. Terms that have the same stems and POS tags are matched. It should be noted that a single word can match with multiple ontology terms. For example, the word "يصيب" in the query "ما الأمراض الذي تصيب الرئتين؟" can map to the ontology class ":infection" and the property ":infects" because they share the same stem. When a word matches with multiple concepts, the POS tags are compared, and only words that have the same POS tag are matched. In the above example, the word "يصيب" is mapped to the property ":infects" because they are verbs.

D. Stemming

Stemming is the process of reducing inflected (or sometimes derived) words to their word stem, base or root form, generally a written word form [49]. Stemming in Arabic language is difficult compared to English. English language has very little inflection and hence a tendency to have words that are very close to their respective roots. The distinction between the word as a unit of speech and the root as a unit of

meaning is more important in the case of languages where roots have many different forms when used in actual words as is the case in Arabic. While removing prefixes and suffixes in English may create problems, in Arabic both prefixes and suffixes are removed. The difficulty arises because Arabic has two genders, feminine and masculine; three numbers, singular, dual and plural; and three grammatical cases, nominative, genitive and accusative. A noun has the nominative case when it is a subject; accusative when it is the object of a verb; and genitive when it is the object of a preposition. The noun gender, number and grammatical case determine its form [50].

We stem all terms and text in Ontological Dictionary to their stem form to compare them with words in user question. This comparison makes the mapping much easier and increases the semantic similarity.

Stemming process has two goals. In terms of efficiency, stemming reduces the number of unique words in the index, which in turn reduces the storage space required for the index and speeds up the answering question process. In terms of effectiveness, stemming improves the recalling by reducing all forms of word to a base or stemmed form.

In this study, we perform stemming process using Khoja‘s stemmer [18] which involves the following steps to extract the root from Arabic words. For example, a stemming algorithm for Arabic should stem the words (درس ، يدرس ، مدرس، دارس، مدرسة ، تدريس) to its word stem (درس).

- Removing or adding the definite article "ال" from the beginning
- Removing the conjunction letter "و "
- Removing or adding suffixes such as the letter " ون ، ين ، ان ،ات ، يت ، ه ، ة " from the end
- Removing or adding prefixes such as the special letter "ء"
- Pattern matching. Word pattern helps in detecting the letters of the root of the word.

Before we apply stemming on the terms in Ontological Dictionary, we should apply normalizing and remove stop words. Figure 4.9 shows some terms in Ontological Dictionary as examples before stemming and after it.

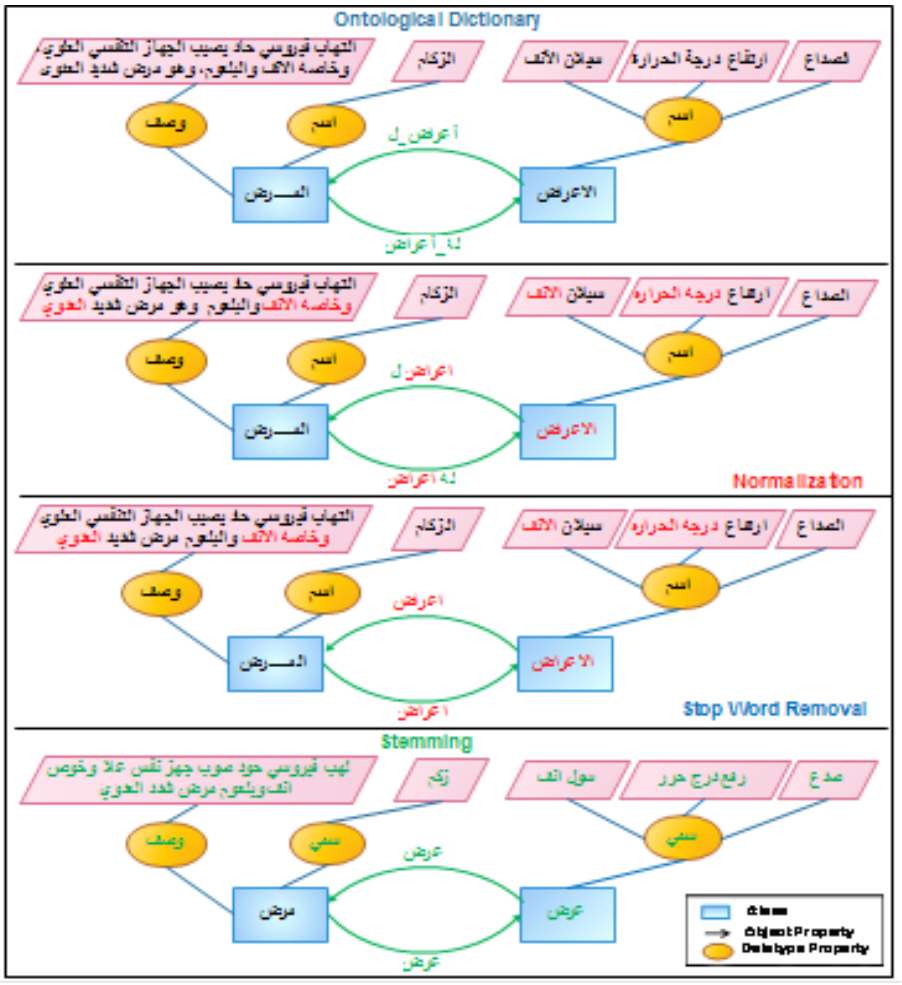


Figure 4.9: Example of stemming on terms in Ontological Dictionary

4.3 Question Processing

Question Processing component processes a natural language query using normalizing, tokenizing, stop word removal, stemming and part of speech tagging. Each pre-processing operation that has been applied to the terms in Ontological Dictionary will be applied in the Question Processing component in addition to the tokenization process. The aim of preprocessing the question is to prepare it for mapping with terms in Ontological Dictionary.

When a Question is asked, the Question processing component does the following pre-processing phases shown in Figure 4.10 and stated below:

- Normalization
- Tokenization: the tokenizer divides the user question into its separate words

- Removing the stop words
- Part of Speech Tagging
- Stemming

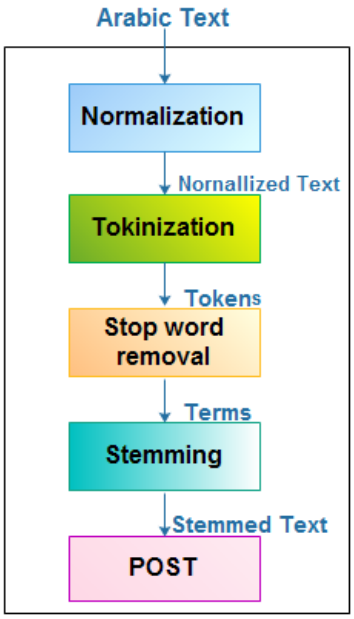


Figure 4.10: The processing phases of NL query

Note that the preprocessing operation in Ontological Dictionary is done once but in the question it is done every time a user ask a question.

4.3.1 Tokenization

Tokenization is the process of segmenting the input sequence into words, phrases, symbols, or other meaningful elements called tokens [51].

Tokenization of a raw text is a standard pre-processing step for many NLP tasks. For English, tokenization usually involves punctuation, splitting and separation of some affixes like possessives. Other languages such as Arabic require more extensive token pre-processing, which is usually called **segmentation** [52].

We use segmentation besides tokenization. Segmentation of Arabic text enables to separate pronouns and conjunctions from words (e.g. أعراضه become أعراض ه , ويصيب becomes يصيب و). This is one of the main differences between Arabic and English. In English we do not need segmentation.

In our system we applied tokenization process using Stanford Word Segmenter which supports Arabic language. As soon as a user inputs a query (e.g. ما المرض الذي من أعراضه ارتفاع درجة الحرارة؟), the Stanford Word Segmenter breaks down a text stream into

meaningful elements called tokens or items (e.g ما المرض الذي من اعراض ه ارتفاع درجه الحراره) as shown in Figure 4.11. A token is usually defined as an alphanumeric string that occurs between white space and punctuation. For each query text, the tokens will be stored in a list based on the following algorithm:

Algorithm 4.1: Tokenization
1. While (Not End of Text)
1.1. Tokenize a word
1.1.1. Word stores in list

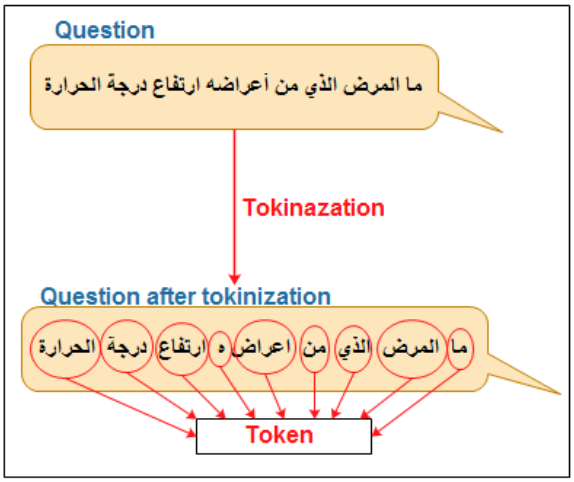


Figure 4.11: Tokenization the question

Figure 4.11 shows user question before tokenizing and after it.

4.3.2 Complete Example of pre-processing phases in Question Processing

In this section, a complete example of pre-processing phases in Question Processing will be provided. We submitted only one question, assuming that questions with similar pattern will be applied to the same phases.

Figure 4.12 shows the pre-processing phases on user question as follows:

1) Normalization phase: In this phase, we follow steps which are explained in section 4.2.3A (see section 4.2.3A). We removed punctuation which is question mark (e.g. ما المرض الذي من أعراضه ارتفاع درجة الحرارة ؟ becomes ما المرض الذي من أعراضه ارتفاع درجة الحرارة), replaced the أ initial by Alif ا (e.g. أعراضه becomes اعراضه) and replaced the ة final by the ه (e.g. درجة becomes درجه , الحرارة becomes الحراره).
2) Tokenization phase: In this phase, we break down a question into tokens and separate pronouns from words (e.g. اعراضه becomes اعراض ه).

3) Removing stop words: In this phase, we remove stop words which are (من، الذي، ما)
4) Stemming: In this phase we perform stemming process using Khoja's stemmer (see section 4.2.3C) (e.g. المرض becomes مرض, العرض becomes عرض , ارتفاع becomes رفع , درجه becomes درج , الحراره becomes حرر).

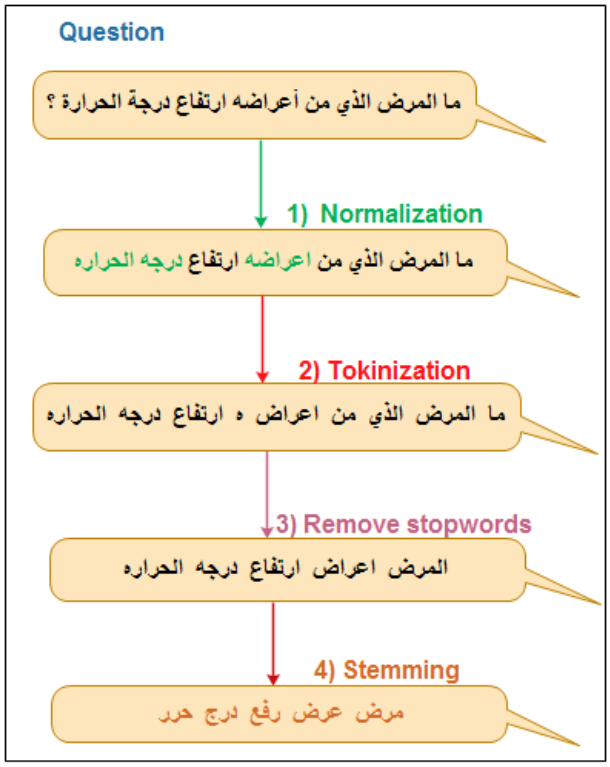


Figure 4.12: Question before and after pre-processing

4.4 Ontology Mapper

After pre-processing both of Ontological Dictionary and user question, we map each word in Natural Language (NL) queries with terms in Ontological Dictionary. When mapping succeeds, we get a list of ontology terms mapped to each word in user query. Consequently, we can determine the type of each word in the question whether it is a class, property, instance or literal.

Figure 4.13 shows Ontology Mapper process.

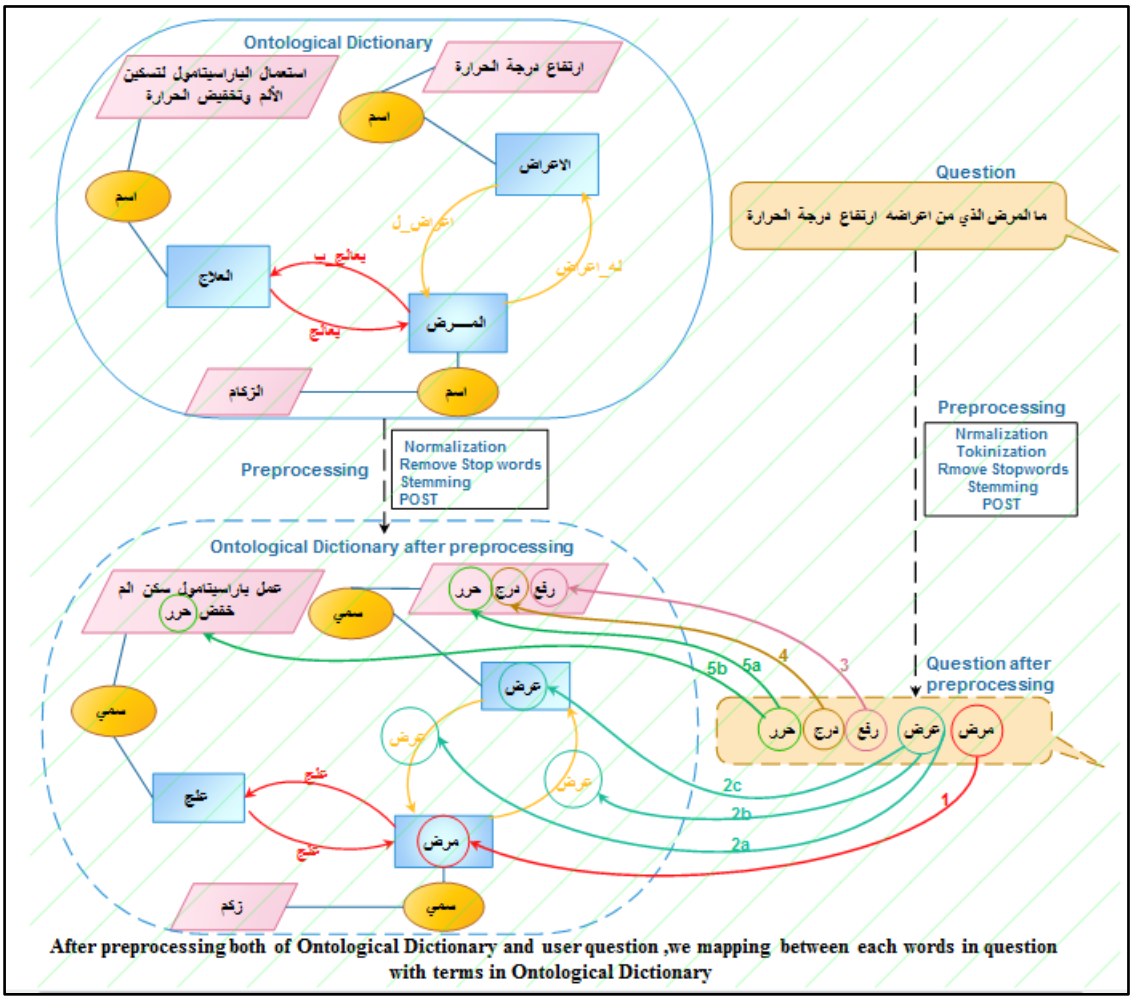


Figure 4.13: Ontology Mapper

Figure 4.13 shows the mapping processes between each word in question and terms in Ontological Dictionary after pre-processing each of them. For example:

1) The word (المرض) in user question is mapped with (class: المرض) in Ontological Dictionary.
2) The word (اعراضه) in question is mapped with:
   a) (Datatype property: اعراض_ل) in Ontological Dictionary.
   b) (Datatype property: له_اعراض) in Ontological Dictionary.
   c) (Class: الاعراض) in Ontological Dictionary.
3) The word (ارتفاع) in user question is mapped with (literal: ارتفاع) in Ontological Dictionary.
4) The word (درجة) in user question is mapped with (literal: درجة) in Ontological Dictionary.

5) The word (الحرارة) in user question is mapped with (literal: الحرارة) in Ontological Dictionary.

As we see, we have challenges in the process of mapping which are:

- Matching of itemset in user question with ontology terms in Ontological Dictionary.

Itemset: Set of items that occur together, for example:

"ارتفاع ضغط الدم", "الرمد الحبيبي", "البول السكري","ارتفاع درجة الحرارة"

The problem:

If the user question (e.g.؟ ما المرض الذي من أعراضه ارتفاع درجة الحرارة) contains an itemset (e.g. ارتفاع درجة الحرارة), and when we tokenize it as we explained in section 4.3.A, it becomes (ما المرض الذي من اعراض ه ارتفاع درجه الحراره). So each token as an item is matched to terms in Ontological Dictionary separately (e.g. الحراره in user question is mapping to any term in Ontological Dictionary which is named الحرارة or its stem is حرر , as well as both ارتفاع ، درجه), as shown in Figure 4.14.

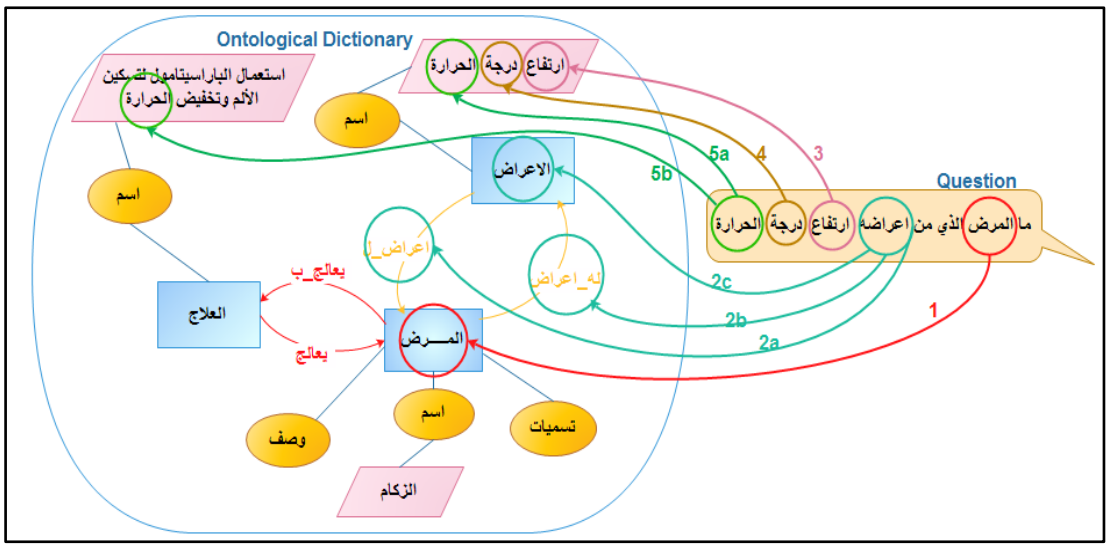


Figure 4.14: Mapping Process

It was worthwhile to take the itemset as a unit (e.g. ارتفاع درجة الحرارة) from user question (e.g.؟ ما المرض الذي من اعراضه ارتفاع درجة الحرارة) and try to find the same itemset in Ontological Dictionary for mapping with it. So we should identify which tokens in queries are item and which are itemset during the mapping process.

Figure 4.15 shows an example of item and itemset.

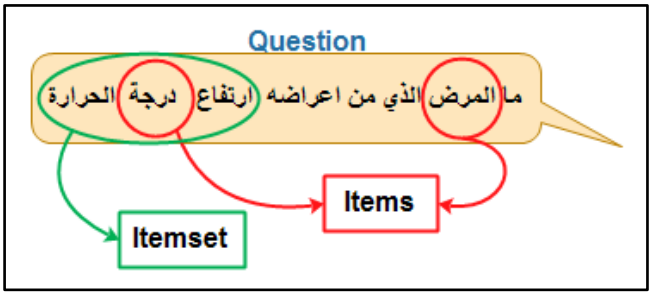


Figure 4.15: An example of item and itemset

Solution: Generation of n-gram for user query when mapping with Ontological Dictionary.

An n-gram is a contiguous sequence of n items from a given sequence of text or speech. The items can be phonemes, syllables, letters, words or base pairs according to the application. An n-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram"; size 3 is a "trigram". Larger sizes are sometimes referred to by the value of n, e.g., "four-gram", "five-gram", and so on.

In our proposed system, we solved the problem of matching itemset in user question with terms in Ontological Dictionary through tokenizing the question into n gram. N refers to size and gram refers to item. Generation of n-gram is most important to identify which tokens in queries are item and which are itemset when mapping them with terms in Ontological Dictionary.

First, we tokenize question into 1-gram (represent one item) and map it with terms in Ontological Dictionary as shown in Figure 4.16. Second, we tokenize question into 2-gram, that means taking sequence of two items in question and mapping them with terms in Ontological Dictionary as shown in Figure 4.17. Third, we tokenize into 3-gram (sequence of three items) and map them with terms in Ontological Dictionary as shown in Figure 4.18 and so on. In our system, we have used "four-gram" as a maximum limit for our case.

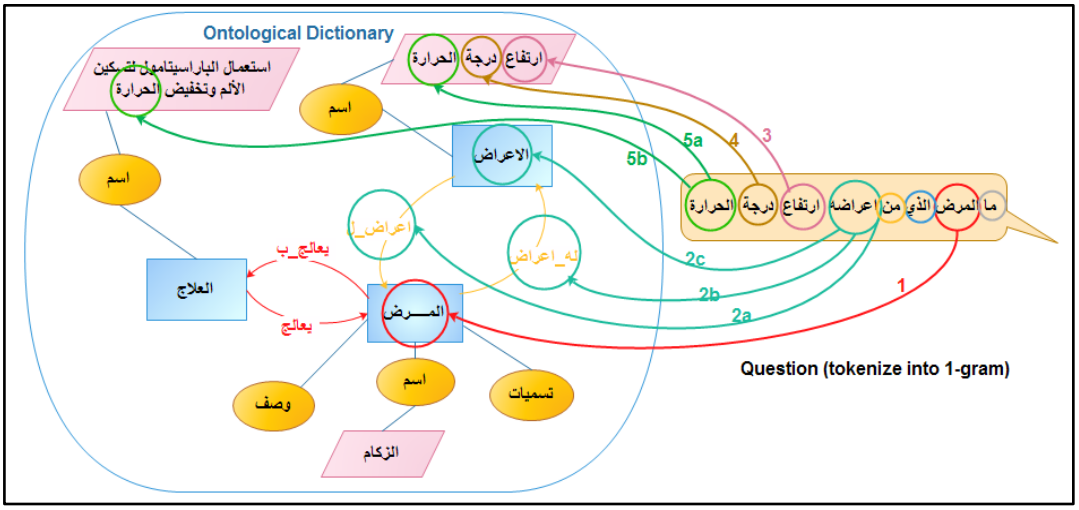


Figure 4.16: 1-gram matching

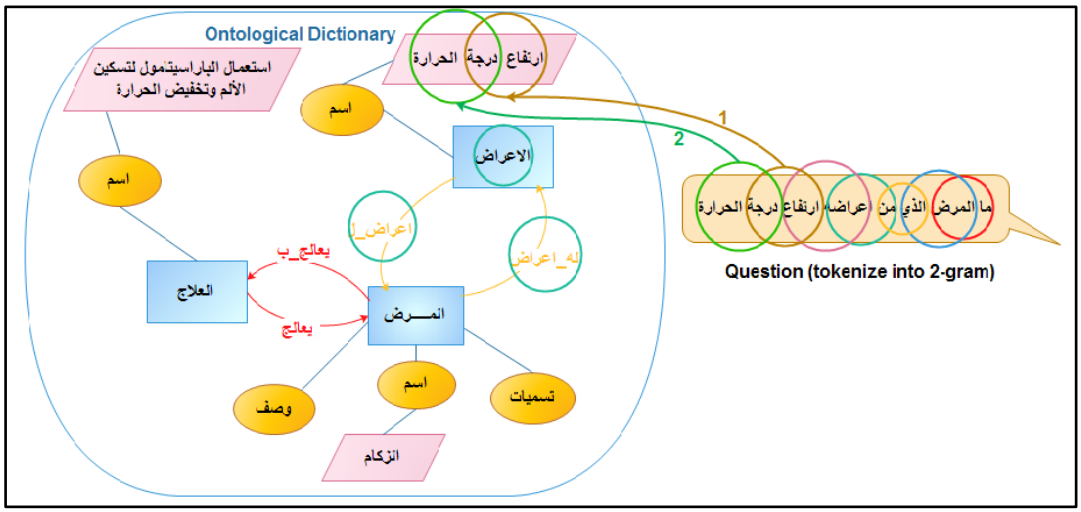


Figure 4.17: 2-gram matching

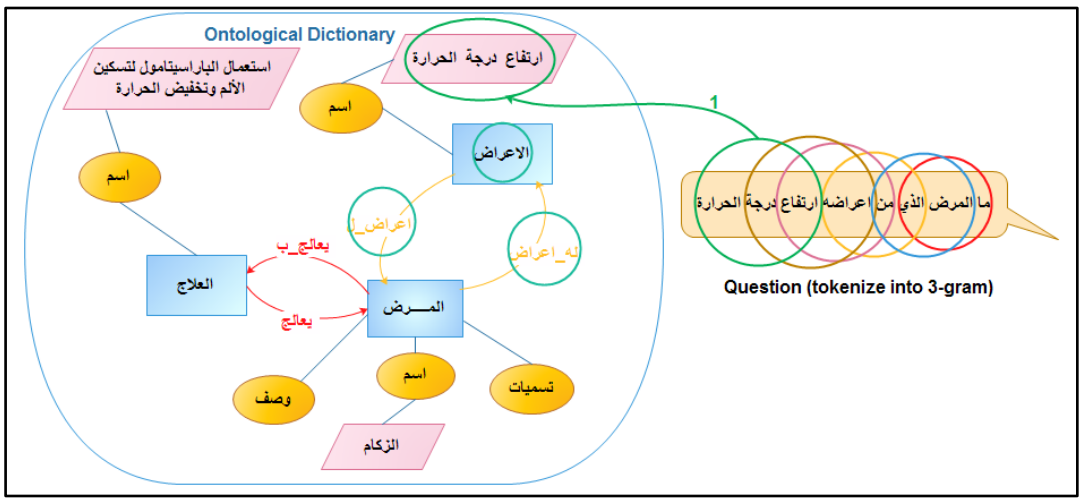


Figure 4.18: 3-gram matching

After that, we have checked if 2-gram matching occurs. We removed 1-gram matching. Also if 3-gram matching occurs, we removed 2-gram matching and so on. Based on this, it can be said that the mapping process is actually between item or itemset (n-gram) in user query with terms in Ontological Dictionary.

- Ambiguity

The problem: Sometimes the token (single item) in user query can match with multiple ontology terms that have the same stems in Ontological Dictionary at the same time; for example, the token (يصاب) in the user question (ما العضو الذي يصاب بالرمد الحبيبي) can map to the properties (يصيب ، يصاب_ب) and also to the literal (اصابة) simultaneously because they share the same stem, as shown in Figure 4.19.

So the problem lies in determining which ontology terms in Ontological dictionary are appropriate for matching with token in user question.

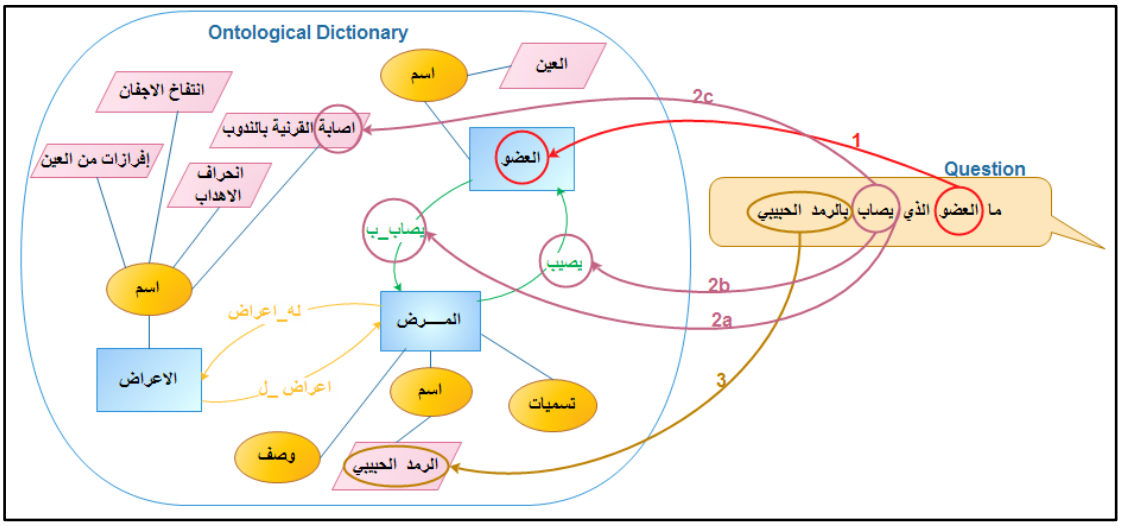


Figure 4.19: Ambiguity problem in the mapping process

In Figure 4.19:

1- The word (العضو) in user question is mapping with (class: العضو) in Ontological Dictionary.
2- The word (يصاب) in question is mapping with :
   a) (Datatype property: يصيب) in Ontological Dictionary.
   b) (Datatype property: يصاب_ب) in Ontological Dictionary.
   c) (Literal: اصابة) in Ontological Dictionary.
3- The itemset (الرمد الحبيبي) in user question is mapping with (literal:الرمد الحبيبي) in Ontological Dictionary as we explained previously.

Solution: We have used a Part Of Speech Tagging (POST) to resolve ambiguity issue. The process of assigning a part-of-speech to each token in user question is to determine its type if it is a noun. a verb or a particle for the purpose of selecting appropriate terms in Ontological Dictionary to map with it. When a word (token) matches with multiple terms in Ontological Dictionary, the POS tags are compared, and only words that have the same POS tag are matched**.**

For example, the token (يصاب) in user question (ما العضو الذي يصاب بالرمد الحبيبي؟) is a verb then the mapping will be with the properties (يصيب ، يصاب_ب) because they are verbs and eliminate the mapping with literal (اصابة) because it is a noun, as shown in Figure 4.20.

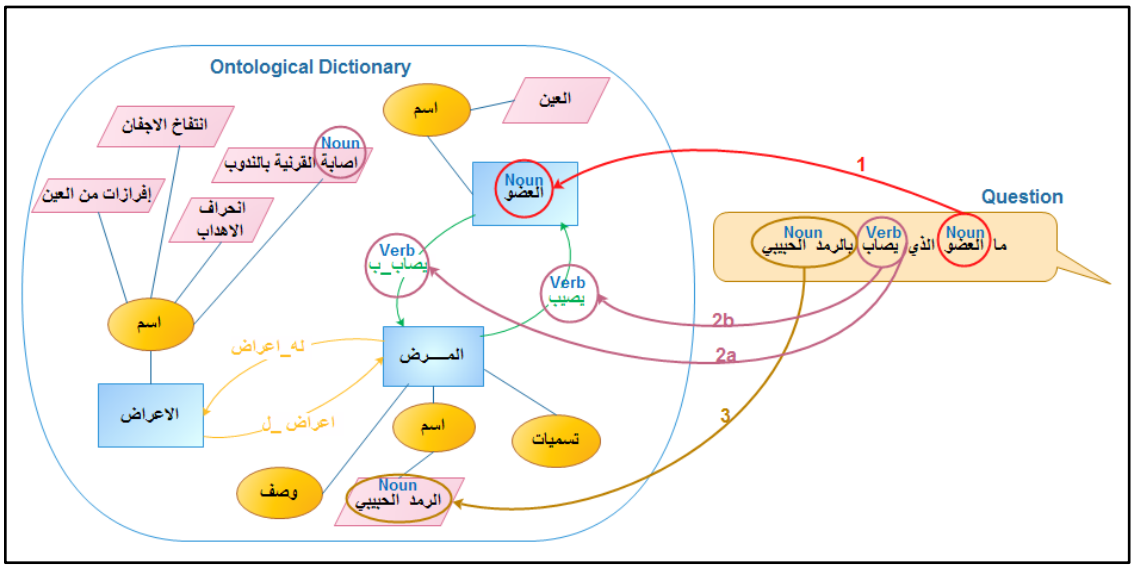


Figure 4.20: Mapping to ontology terms based on POST

Moreover, we have a special case in ambiguity issue. For example, in user question (ما المرض الذي من أعراضه ارتفاع درجة الحرارة ؟), the token (أعراضه) which is a noun mapped with class (الأعراض) which is a noun and mapped with properties (له_أعراض ، أعراض_ل) which are also nouns as shown in Figure 4.21.

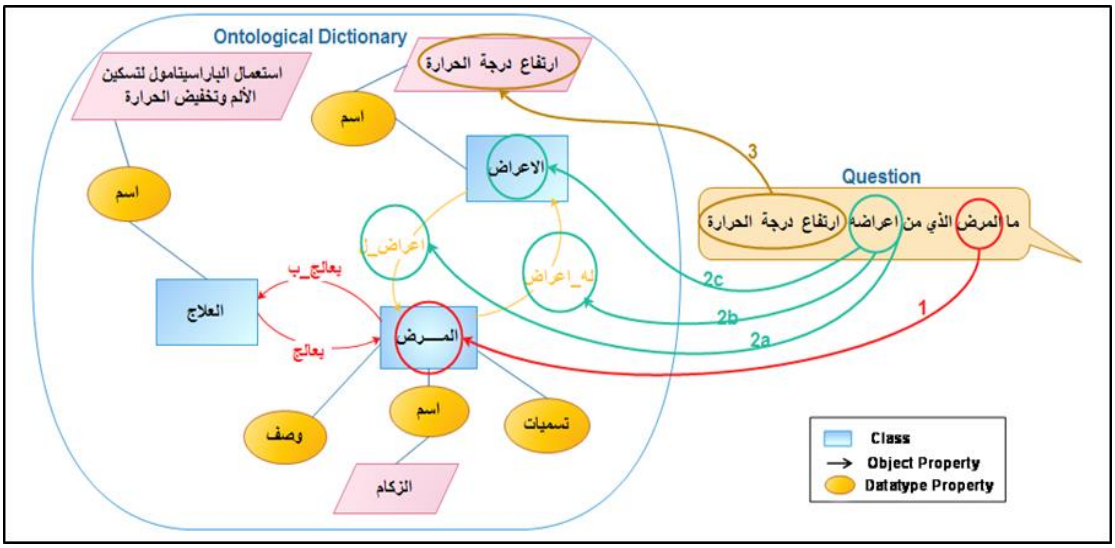


Figure 4.21: Mapping to ontology terms based on POST 2

To solve this ambiguity problem, we assume that if a word in the user query is a noun the priority for mapping will be to classes first then mapping to property. So in our example the word "أعراضه" will be mapped to the class "Symptom". The priority is for the class not the property. Then, we will look for a property to link the class "Disease" with the class "Symptom".

Chapter 5: Interpretation of NL query into SPARQL

In this Chapter, the approach used to translate Arabic queries to SPARQL is explained in detail. The approach builds on the work of AlAgha [53] who proposed an approach to translate Arabic sentences to SPARQL. We build on his work and introduce more grammar rules to cover different cases of queries.

5.1 Triple Builder

The triple Builder is the backbone component of the system and is responsible for generating RDF triples by combining the ontology terms recognized by the mapper component. It builds associations between Ontology concepts to formulate triple patterns, and then aggregates these patterns to construct a complete SPARQL query that, when executed, can retrieve the intended answers. In what follows, we begin by giving a brief overview of the most important concepts underlying SPARQL queries. Thereafter, we present our approach to translate Arabic natural query to SPARQL queries.

We emphasize that the proposed approach of interpreting the NL queries does not make intense use of linguistic techniques such as text parsing or morphological analysis. Instead, it highly depends on the quality and choice of vocabulary of the ontology as well as the rules we defined to capture dependencies between the query terms. The translation of NL queries to SPARQL requires capturing syntactic relations and dependencies between the query terms. Existing approaches that work with English script have traditionally approached this requirement by using parse trees (e.g. Stanford NLP parser) to interpret parse trees of NL queries to SARQL. However, parsing is often time-consuming, error-prone and difficult to manipulate [54]. Moreover, Text Parsing of Arabic sentences is difficult and remains much poorer than English [54].This limitation, which originates from the limited support of Arabic NLP, is subject for revision in the future versions of the system according to the progress made in the field of Arabic NLP.

The discussion and examples given throughout this section are based on a sample ontology which we developed for illustration and testing purposes and which covers the domain of diseases. Figure 5.1 depicts an excerpt of the ontology showing the ontology classes (e.g. :Cure, :Disease, :Symptom, :Organ) as well as the inter-relationships between (e.g. object properties).The prefix ont: denotes the ontology's base uri. Arabic translations of all class and property names are stored using the rdfs:label property which is not shown in the Figure 5.1 for simplicity. Arabic names are retrieved and used for matching user query, inputted in Arabic, with the ontology content. We stress that the AR2SPARQL system can be easily configured to use any ontology as long as the Arabic translation of its content is supplied.

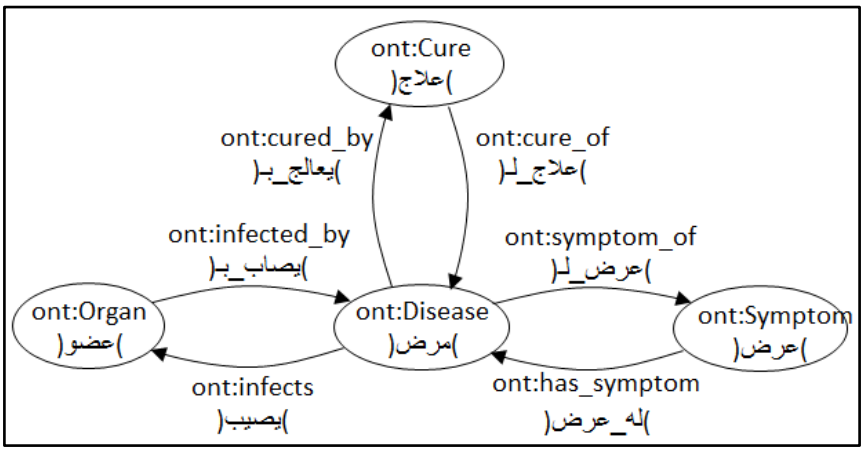


Figure 5.1: An excerpt of the disease ontology

A SPARQL query, in its basic format, consists of two parts: the SELECT clause identifies the variables to appear in the query results, and the WHERE clause provides the basic graph pattern to match against the data graph. The WHERE clause consists of one or more triple patterns <s, p, o> where s denotes the subject, p denotes the predicate and o denotes the object. In SPARQL queries, the subject, the predicate and the object can be variables, resources (written as URIs) or literal values. Variables refer to unknown components which the user often intends to know. For example, given the following SPARQL query: SELECT ?person where ?person <foaf:name> "Ahmed": the subject is the variable denoted by ?person, the predicate is the resource denoted by the URI: foaf:name, and the object is the literal value "Ahmed".

Let C be the set of all classes, P the set of all properties, I the set of all instance and L the set of all literals contained in the target knowledge base of the SPARQL queries at hand. Let $< s\ p\ o >$ be an RDF triple pattern where s, p and o denote the subject, predicate and object respectively. We define the translation function $\rho: W^* \rightarrow < s\ p\ o >^*$as a function that maps each word Wor, a sequence of words, in a user query to one or more RDF triple pattern(s). Formally, the goal of this paper is to devise the extension of ρ to Arabic natural language queries. We adopt a rule-based approach to achieve this goal as the following:

Rule 1: $\rho(x)$ *where* $x \in (P \cup I \cup L) \Rightarrow x$

Rule 2: $\rho(C) \Rightarrow ?var\ a\ C$ *where the property a is equivalent to the property rdf:type*

The above two rules define how the atomic types (e.g. classes, instance, properties and literal values) are represented in the SPARQL body. Rule 1 indicates that properties, instances and literal values are placed unchanged in the SPARQL body. Only

Class terms are represented differently as variables in the SPARQL body, as indicated in Rule 2.

The translation of NL queries to SPARQL is handled by the Triple Builder component. The process of constructing a SPARQL query from the NL query consists of three procedures that are followed based on the complexity and completeness of the NL query. These procedures are:

1- Interpretation of NL queries that map to complete triple patterns.
2- Interpretation of NL queries that do not map to complete patterns.
3- Interpretation of NL queries that consist of multiple queries linked with conjunctions.

These procedures are explained in what follows, and illustrative examples are given:

5.2 Interpretation of NL queries that map to complete triple patterns

Step 1: The query terms are first mapped to the relevant ontology terms. The output of this phase is a set of ontology terms ordered according to their occurrence in the Arabic query.

Step 2: Identified ontology terms are scanned for a complete triple pattern. A complete triple pattern <s p o> is defined as a sequence of ontology terms that map to a subject, a predicate and an object in sequence. A subject can be either a class or an instance. An object can be a class, an instance or a literal value. A predicate can by either an object property or a data type property. The subject and the object should belong to the domain and the range of the predicate respectively.

Step 3: If a complete triple pattern is captured based on the above definition, the interpretation of the NL query is straightforward. Rules 1 and 2 are applied according to the type of each term, and results are linked together to form one or more triple patterns. The translation function ρ of a complete triple pattern <s p o> can be expressed as follow:

Rule 3: $\rho(s,p,o) \Rightarrow \ \rho(s) \wedge \rho(p) \wedge \rho(o)$

$where\ s \in (C \cup I), p \in P, o \in (C \cup I \cup L), s \in Domain\ of\ P, o \in Range\ of\ P$

Step 4: The triple patterns generated from the previous step will formulate the WHERE clause of the SPARQL query. The following step is to construct the SELECT clause by choosing variables that should appear in the query results from those included in the WHERE clause. This process is done as follows: 1) find the question words such

as "ما", "من","أين" or command words such as "أذكر", 2) take the nouns that directly follow the question words as targets. 3) From the triples generated in Step 3, take variables that correspond to the target nouns. These variables will be part of the SELECT clause. Detailed rules vary for different question/command words: for example, quantity questions starting with "كم عدد" is interpreted into something like "SELECT COUNT(DISTINCT ?x)". For Yes/No questions, e.g. questions starting with the question word "هل", the ASK form is used to test whether or not a query pattern has a solution. For example, the NL query "هل يصيب السكر الكبد؟" is interpreted into something like "ASK WHERE {:Diabetes :infects :Liver}" which will return either Yes or No depending on whether or not a solution exists.

The system can also process and interpret NL questions consisting of multiple queries connected by conjunctions, e.g. " ما أسباب مرض السكر وما طرق علاجه؟" (What are the causes of Diabetes and is it treated?). The SELECT clause for such questions should include multiple variables drawn from the composite queries. The processing of questions with conjunctions is discussed later in this section.

The system is not currently able to recognize and translate comparative and similarity phrases such as "أبرز/أهم/المشابه ل/أطول/أكبر/أكثر" (main, most, longest, largest). These phrases remain a direction to extend the system in the future version.

To illustrate how the above procedure is applied, assume the schema shown in Figure 5.1 is used, and the NL query is: "ما الأمراض الذي تصيب الكبد؟" (What are the diseases that infect the liver?). The translation process is done as follows:

Step 1: The output of the mapping process is: <:Disease (class), :infects (object_property), :Liver (instance)>.

Step 2: The ontology terms are scanned for a complete triple pattern: A subject (:Disease), a predicate (:infects) and an object (:Liver) appear in sequence. The class :Disease and the object :Liver both fulfill the condition that they belong to the domain and range of the property :infects respectively. Thus, a complete triple pattern is identified.

Step 3: The target Rule 1, 2 and 3 are applied as follows:

$\rho(:Disease, :infects, :Liver) \Rightarrow \rho(:Disease) \wedge \rho(:infects) \wedge \rho(:Liver)$

=>?var rdf:type :Disease. ?var :infects :Liver

Note that the above output, which constitutes the WHERE clause of the SPARQL query, is the composite of two triple patterns: the first defines the type of the

variable ?var, while the latter indicates that the variable ?var relates to the instance :Liver through the predicate :infects.

Step 4: The SELECT clause is generated by first finding the nouns that directly follows the question word. In our example, the noun "الأمراض" (Diseases) is the target that should appear in the query results. The variable ?var from the above WHERE clause corresponds to the target noun. Thereby, the SPARQL query after generating the SELECT clause becomes:

SELECT DISTINCT ?var WHERE ?var rdf:type :Disease . ?var :infects :Liver

5.3 Interpretation of NL queries that do not map to complete triple patterns

The above procedure addresses the optimum case in which a complete triple pattern is captured. However, one or more of the triple components can be missing, for different reasons, resulting in an incomplete triple pattern. Consider the following example query: "ما أعراض سرطان البروستاتا؟" (What are the symptoms of Prostate Cancer?): the words "أعراض" و "سرطان البروستاتا"correspond to an ontology class and an instance respectively, but no word maps to a valid property. In another example: " ما الذي يصاب بالنقرس؟": the verb "يصاب" corresponds to a property, the noun "النقرس" corresponds to an instance, but no word maps to a term that can represent a valid subject.

In such cases, the query builder will not be able to construct a valid SPARQL pattern by directly following the above procedure. It is necessary first to determine valid alternatives that substitute the missing components. Only then, a complete RDF triple can be capture and, hence, the above procedure can be applied. In what follows, we address cases of incomplete triples and explain how the absence of some triple components can be compensated:

5.3.1 Omitted Predicate

There are three cases where a predicate can be omitted as the following:

Case 1: A class followed by another class: in this case, two consecutive words of the user query map to two ontology classes, but no connecting property is found. Consider the query: "ما أعراض المرض النفسي؟" (What are the symptoms of a mental disease?). Mapping the query words to the ontology will produce: (:Symptom "Class", :Mental disease "Class"). No complete triple pattern is captured because no property is found. A valid object property should be used to form a complete triple by associating

the class terms. To identify the missing property, the following SPARQL query is executed over the ontology:

SELECT DISTINCT ?property WHERE {{?property rdfs:domain ?C1 . ?property rdfs:range ?C2} UNION {?property rdfs:domain ?C2 . ?property rdfs:range ?C1}}

The above query retrieves from the ontology all properties whose domain is C1 and range is C2, and vice versa. Retrieved properties are candidates to replace the omitted property. If multiple properties exist between the two classes, the user is prompted to choose the desired property.

Given the above example and the schema shown in Figure 5.1, two properties to link the classes "Symptom" and "Disease" are retrieved: "Symptom_of" and its inverse "has_symptom". Any of these properties can be used taking into account the correct order of triple components. For example, if the inverse property "has_symptom is used", the subject and object terms are switched.

Case 2: A class followed by an instance of a class (individual): This case is similar to the above one, except that the object of the triple is an individual (e.g. instance of a class). Example of this case is the user query: "ما هي أسباب الإصابة بالإنفلونزا؟" (What are the reasons of Flue?) where the word "أسباب" maps with the class "Reason" and the word "الانفلونزا" maps with the predefined instance "Flue". The result of the mapping process consists of a class term ("Reason") followed by an instance term ("Flue"). This result cannot be directly translated to a RDF triple since the predicate component is missing. Given a class C and individual I, we execute the following SPARQL query on the knowledge base to retrieve properties that link class C to instance I:

SELECT DISTINCT ?property WHERE {{ ?x ?property I } UNION { I ?property ?x } . ?x rdf:type C }

The above query retrieves all properties that link the individual I to any individual of type C. Note that the queries will retrieve the properties and their inverse ones if there are any.

Case 3: A class term followed by a literal value: Given the user query is: ما الأدوية التي تسبب النعاس؟, and that the query is mapped to the ontology as the following: The term "الأدوية" corresponds to the class ":Disease", the term "النعاس" corresponds to the literal value of the data property "ont:hasSideEffect". It is obvious that no RDF triple can be directly formulated since no property is captured. Missing property can be retrieved by executing the following SPARQL query over the knowledge base:

SELECT ?property WHERE ?x ?property "literal_value"

5.3.2 Omitted Subject

Terms that map to RDF subjects can also be omitted from user queries because they are implied. Omitted subjects should be also replaced to have complete RDF triples. The following are cases of omitted objects and present solutions to replace them:

Case 1: An object property followed by a class (or individual): Examples of this case is the query: "كيف يعالج مرض السكر؟" (How is Diabetes treated?): The verb succeeding the question word maps to the object properties ":treated_with". The phrase "مرض السكر" maps to the instance :Diabetes.

Given an object O of type C, and a property P, the following query is used to identify the missing domain of the property:

SELECT DISTINCT ?SUBJECT_Class WHERE {{?property rdfs:domain ?SUBJECT_CLASS . ?property rdfs:range C .} UNION {?property rdfs:range ?SUBJECT_CLASS . ?property rdfs:domain C .}}

Using the above query, the system will retrieve the class ":علاج" which is part of the Domain of the property "يعالج ب".

Using the above approach, the system is capable of interpreting complex queries which are linked with relative pronouns, e.g. "الذي، التي" and which map to multiple triple patterns. The system scans the user query for consecutive triples and translates them to SPARQL. If incomplete triples are detected whilst scanning, the system applies rules in section to make them complete. While constructing triples, the system may link one triple with its preceding one. It may use some terms from one triple to substitute missing terms in another triple. To illustrate this process, Figure 5.2 shows a sample query and depicts the steps of scanning the query terms to capture RDF triple patterns. Note that the term :Disease is used as an object in the first triple and as a subject in the following triple. The question that may arise here is: why is the term :Symptom not included in the second triple instead of :Disease?. The second part of the query "يصيب القلب" lacks a subject, which is implied from the first part. To detect it correctly, we choose from the first part the term that belongs to the domain of the property ":infects". Since only the term: Disease fulfills this condition, it is used to substitute the subject of the second triple. After determining the complete triple patterns, Rules 1, 2 and 3 can be applied to get the following SPARQL query:

SELECT DISTINCT ?var1 WHERE ?var1 a :Symptom . ?var2 a :Disease . ?var1 :symptom_of ?var2 . ?var2 :infects :Heart

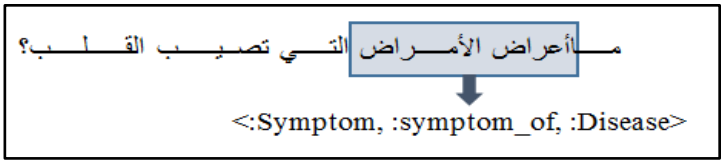


Figure 5.2: Example No. 1

5.4 Interpretation of NL queries that consist of multiple queries linked with conjunctions.

A user question may consist of multiple queries connected with coordinating conjunctions (e.g. "و", "أو"). Each query is often interpreted into one or more RDF triples. However, queries that are connected with conjunctions cannot be processed separately because they often depend on each other, e.g. one query corresponds to entities in the preceding query. Consider the following question: ما المرض الذي يصيب الكبد ويسبب عسر الهضم" (What is the disease that infects the liver and causes indigestion?).The subject in the sentence after the conjunction is omitted because it refers to the subject of the first sentence. Thus, the dependency between the sentences around the conjunction should be considered in the interpretation process.

Related efforts working on the English script often used a statistical parser to identify dependencies between phrases. Phrases whose parent is a conjunctive node in the parse tree are considered related and are processed together. However, as discussed earlier, building parse trees of Arabic text is often more complicated, and produces very poor results as compared to English counterparts. Due to the limited support for Arabic text parsing, a parser-free approach is proposed and used to determine dependencies between sentences and then to translate them to complete RDF triples. The approach consists of the following steps:

Step1: The NL query is broken into clauses based on the conjunctions.

Step 2: Find the pivot clause from the clauses resulted from Step 1. We define the pivot clause as the first clause of the user query that is interpreted into a complete RDF triple. To find the pivot sentence, we break the query into sentences using the conjunctions, and then apply the ontology matcher and the grammar rules defined in section on each sentence. The first sentence that meets the characteristics of a complete RDF triples is used as the pivot sentences. A pivot sentence holds a significant role because other sentences can refer to it. Our intention is to use the pivot sentence to help identify omitted terms in the dependent sentences and hence complete their RDF representations.

Given the NL query:"ما المرض الذي يصيب الكبد ويسبب عسر الهضم", the pivot clause is "المرض الذي يصيب الكبد" because it maps to a subject, a predicate and an object in a sequence. The subject of the clause "ويسبب عسر الهضم" is omitted because it is implied by the pivot sentences.

Note that the pivot clause is not necessarily the first segment of the user query. For instance, in the query: "ما أعراض ومسببات مرض السكر؟", the clause following the conjunction is realized as the pivot clause because it maps to a complete RDF triple.

Step3: Determining dependencies in the NL query and replacing missing components: this is done by mapping the ontology terms identified in each dependent clause to the ontology terms of the pivot clause. Terms omitted from the dependent clauses are replaced by the appropriate terms from the pivot clause according to the following rules:

Let $Subject_v, Predicate_v \; and \; Object_v$ be the components of the RDF triple corresponding to the pivot clause. Let $Subject_d, Predicate_d \; and \; Object_d$ be the components of the RDF triple corresponding to a dependent clause. Note that $Subject_v, Predicate_v \; and \; Object_v$ are all explicitly present while one, or more, of $S_d, P_d \; and \; O_d$ are missing. The aim is to replace missing components by analyzing relations between the pivot and dependent clauses. We define the following rules:

Rule 1:

$$subject_d \; is \; missing \; AND \; subject_v \in Domain \; of \; predicate_d$$
$$\Rightarrow \; Replace \; subject_v \; with \; subject_d$$

This rule means that if the subject of the dependent clause is missing, the subject of the pivot clause will be used instead if and only if it belongs to the domain of the predicate of the dependent clause. In the previous example, the dependent clause " يسبب عسر الهضم" lacks the subject component. According to the schema in Figure 5.3, the term :Disease belongs to the domain of the predicate :causes. Thus, it will be used to compensate the missing subject of the dependent clause.

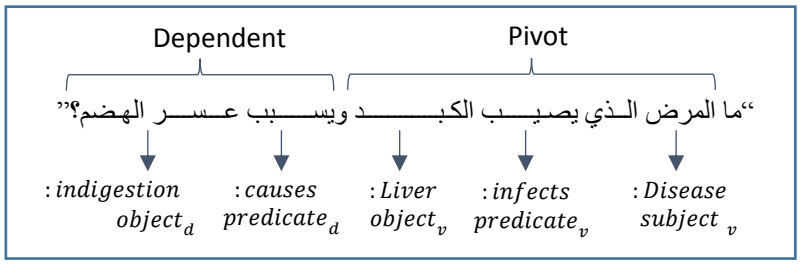


Figure 5.3: Example No. 2

Step 4: After replacing the missing subject, the dependent clause will form a complete triple, and thus can be interpreted into SPARQL by applying the procedure discussed in section. Since both pivot and dependent clauses share the same subject, they will also use the same variable that corresponds to that subject in the resultant query. The resultant SPARQL query will be as the following:

SELECT ?var WHERE {?var a :Disease . ?var :infects :Liver. ?var :causes :indigestion}

Rule 2:

$$subject_d \; AND \; predicate_d \; are \; both \; missing \; AND \; object_d \in Range \; of \; predicate_v$$
$$\Rightarrow \; Use \; subject_v \; AND \; predicate_v \; for \; dependent \; clause$$

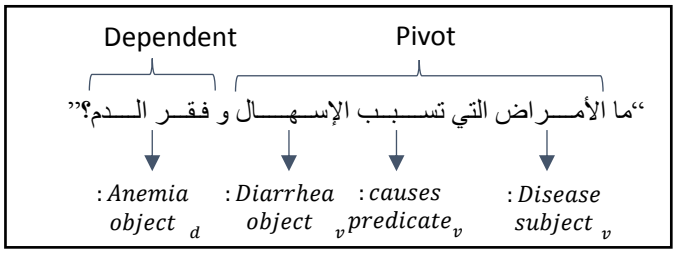


Figure 5.4: Example No. 3

In the example shown in Figure 5.4, the pivot clause is " ما الأمراض التي تسبب الإسهال؟" because it encompasses a complete RDF triple pattern. In the clause that follows the conjunction, the phrase "فقر الدم" maps to an ontology instance. According to Rule 2, if that instance belongs to the domain of the predicate of the pivot clause, the subject and predicate of the pivot clause are used to complete the dependent clause. This process is equivalent to rephrasing the above query to be: " ما الأمراض التي تسبب الإسهال والأمراض التي تسبب فقر الدم؟". The resultant SPARQL query will be:

SELECT DISTINCT ?var WHERE { ?var a :Disease . ?var :causes :Diarrha . ?var :causes :Anemia}

Rule 3:

Dependent clause is a complete triple pattern => add a new triple pattern to link $Subject_v$ *with* $Subject_d$

Rule 3 indicates that if the query consists of two triple patterns that are both complete and are connected with a conjunction, then create a new triple pattern to link the subjects of the two triple patterns. The new triple pattern works as a connector between the two other patterns. In the example query shown in Figure 5.5, the pivot clause "ما المرض الذي يصيب الكبد" corresponds to a complete triple pattern <:Disease,

:infects:, :Liver>. The dependent clause "من أعراضه اليرقان" maps to the ontology terms <:Symptom, :hasName, :Jaundice>. It is obvious that the pivot clause initially has no word that maps to a valid property. However, the missing property can be determined by using the approach explained in this section. Having two complete triple patterns connected with a conjunction, the next step is to interpret this linkage into SPARQL by creating a new triple to act as a connector. The connecting triple has the template $< subject_v, ?, subject_d >$. The missing predicate of the template has the condition that $subject_v$ *and* $subject_d$ should belong to its domain and range respectively. This predicate can be easily determined by querying the ontology as explained in this section. Referring to the running example and to the schema in Figure 5.1, the property ":hasSymptom" is the right predicate. The resultant SPARQL query will be:

SELECT DISTINCT ?var1 WHERE {?var1 a :Disease . ?var1 :infects :Liver . ?var2 a :Symptom . ?var2 :hasName "Jaundice" . ?var1 :hasSymptom ?var2}

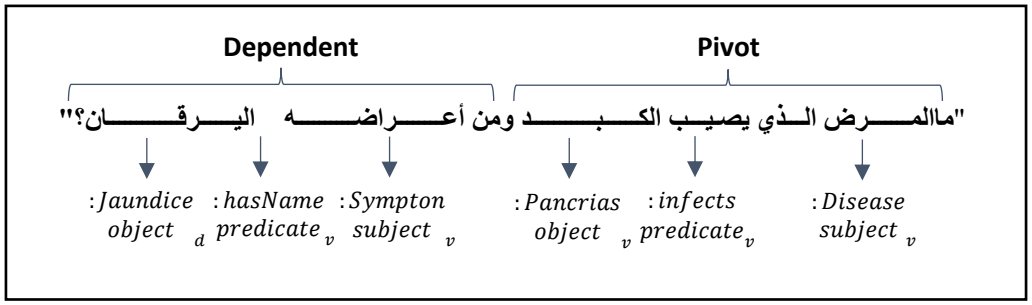


Figure 5.5: Example No. 4

5.5 Summary

In the previous sections, the process on interpreting the NL query into SPARQL was discussed. Note that one or more of the discussed procedures may be used according to the complexity and completeness of the NL query. If the NL query can be directly mapped to complete RDF triple patterns, then procedure 1 is followed. If the query has missing RDF components so that it cannot directly formulate a complete RDF triple, then procedure 2 is used to complete the triple patterns before applying procedure 1. Procedure 3 is used to determine dependencies between multiple queries linked with conjunctions. As soon as the dependencies are resolved and omitted RDF components are determined, procedure 1 is applied to generate the SPARQL query.

Chapter 6: Experimental Results and Evaluation

This chapter presents the implementation and the evaluation of our work. Firstly, we state the tools and programs used to develop the proposed model, test the system and evaluation explained next. At the end of the chapter we shall discuss our results.

6.1 Implementation of the Arabic QA System

For developing the Arabic Question Answering System, we need different components at different phases for retrieving the answer. These components will be used to understand the semantics behind the question, identify the relationships between question components, and transform to SPARQL. For this reason, we used various kinds of open source software tools. To implement the Arabic QA system, we used java programming language because most of pre-processing tools are found in java language beside special tools.

Figure 6.1 depicts the components of Arabic QA System and shows the role of each component as follows:

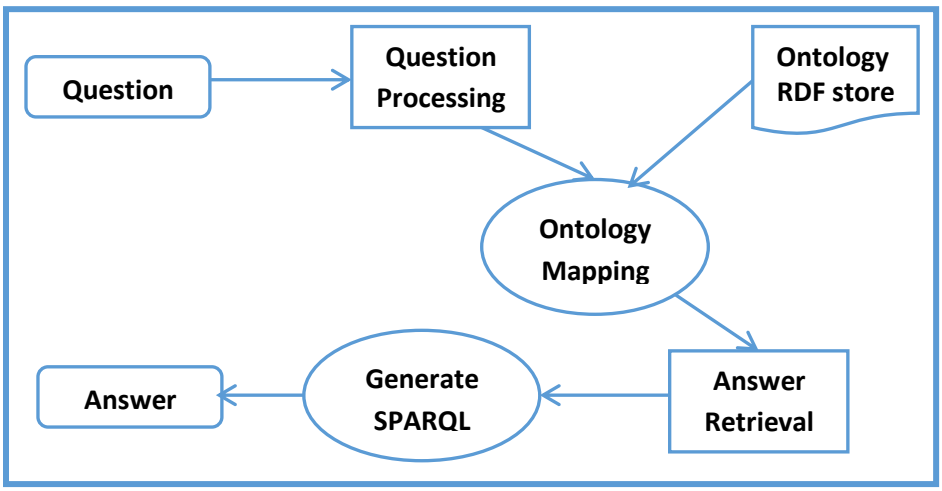


Figure 6.1: Arabic QA Components

- The Ontology, which represents the schema, is stored in a file. RDF data representing the individuals and other annotations are stored as RDF triples in a MySQL database table. This resembles the generic architecture of Semantic Web applications where RDF data is often distributed and stored in different locations while the ontology remains intact.
- We used Jena API for ontology manipulation and reasoning. Note that we applied an ontology reasoner on the ontology and the associated annotations before running the system. The reasoner enables the system to recognize new

facts that do not initially exist. For example, the reasoner can identify new relations from transitive and inverse relations.

- For adding individuals and annotation, we used a component called Diseases data entry.
- To extract everything in the ontology and rdf data and return a list of ontology entities (classes, properties, individuals), we used a component called OntDataRetrieval.
- We used OntEntry to store data retrieved from the ontology.
- Question Processing component processes a natural language query using normalizing, tokenizing, stop word removal, stemming and part of speech tagging.
- The ontology mapping components are responsible for creating n-grams and matching the question words to the ontology terms.
- When question words are mapped to the ontology terms, data structure named WordTagPair is used to store each question word with its corresponding ontology term.
- A component called SPARQL generator takes the output of the ontology mapping components and converts the user query to SPARQL query using the rules define in Chapter 5.

6.2 Tools and Programs

To implement Arabic QA system and documentation of the thesis, the following tools had been used:

- For ontology building, we used Protégé 3.4[1] which is a free, open source tool for editing and managing Ontologies [55].
- To implement the queries, Jena[2] framework has been used. Jena is a Java toolkit which provides an API for creating and manipulating RDF models.
- Java Development Kit (JDK) 1.6: A software development package from Sun Microsystems that implements the basic set of tools needed to write, test and debug Java applications.
- Eclipse Standard/SDK: This is the program which helps us to build and finish the system implementation using java language.
- MySQL 5.6: We stored all the data we need to retrieve answers.
- We use Stanford NLP[3] for normalization, tokenization and POS tagging.

[1] http://protege.stanford.edu/
[2] http://jena.sourceforge.net/
[3] http://nlp.stanford.edu/projects/arabic.shtml

- Shreen Khoja Stemmer [18]: This is a free Arabic stemmer. We use it to stem each Arabic word in the document. Also, it removes all strange words and non-letters from the text.
- Microsoft Word 2010: This is the main program used to write the documentation of the system.

6.3 Testing

It is worth mentioning that we could not find any Arabic test data that is tailored for ontology-based question answering in Arabic (there are plenty of test data in English). Therefore, we had to build our own test data.

We have developed the ontology contents for Pathology Domain. The ontology is implemented with Protégé, in OWL format, and can be mapped into a database. Experts from the domain also helped us to validate the ontology contents. Table 6.1 depicts the size of the ontology which shows the number of classes, object properties, and data properties in our ontology.

Table 6.1: The size of the ontology

Domain and Scope of the Ontology	Pathology "علم الامراض"
Number of Classes	7
Number of Object Properties	11
Number of Data Properties	3
Number of Instances	325

In our ontology we created and stored instances in a separate database in format of RDF triples. We defined around 325 instances representing all ontology concepts. Table 6.2 depicts the number of instance.

Table 6.2: The number of individuals

Name of Classes	Number of individuals
Disease Name	15
Disease Description	15
Disease Synonyms	29
Categories	30

Symptoms	67
Organs	24
Diagnoses	40
Cures	53
Reasons	52

We performed a series of experiments to demonstrate sample results. All our experiments depend on the data in ontology domain and individuals that created system development. We have collected 30 questions from different people ranging in age from 18 to 40 years old after giving them information about diseases in the knowledge base. Then we incorporated these questions into our system and tested them for answers. Table 6.3 shows the questions that have been tested.

Table 6.3: Questions that have been tested

No.	Question/English	Question/Arabic
1	What are the symptoms of a tuberculosis?	ما أعراض مرض السل ؟
2	What organs that are infected by tuberculosis?	ما هي الاعضاء التي يمكن ان تصاب بمرض التدرن ؟
3	What are the varieties of diabetes ?	ما هي اصناف مرض السكري ؟
4	What is the cause of trachoma?	ما سبب الرمد الحبيبي ؟
5	How to diagnose thalassemia?	كيف يشخص مرض الثلاسيميا ؟
6	What is gout?	ما هو مرض النقرس؟
7	Mention the names of infectious diseases?	اذكر أسماء الأمراض المعدية ؟
8	What are the diseases that infect the blood?	ما هي الامراض التي تصيب الدم ؟
9	What is a cure of meningitis?	ما علاج مرض التهاب السحايا؟
10	What are diseases which their symptoms is high temperature?	ما الامراض التي من اعراضها ارتفاع درجة الحرارة ؟
11	What are diseases that infect the blood and their symptom is high temperature?	ما الأمراض التي تصيب الدم ومن أعراضها ارتفاع درجة الحرارة ؟
12	What are the infectious diseases?	ما هي الامراض المعدية ؟
13	What is the flu disease and what are their symptoms?	ما هو مرض الزكام وما هي اعراضه؟
14	How is cured cholera ?	كيف يعالج مرض الكوليرا ؟
15	What is the disease that infects the intestines and their symptoms are vomiting and diarrhea?	ما هو المرض الذي يصيب الامعاء ومن اعراضه القيء والاسهال؟
16	Describe the malaria?	اوصف مرض الملاريا؟
17	What is the classification of asthma?	ما هو تصنيف مرض الربو؟
18	How is the diagnosis of asthma?	كيف يتم تشخيص مرض الربو؟

19	Mention other names to AIDS?	اذكر تسميات اخرى لمرض الايدز؟
20	What are the causes of measles?	ما هي اسباب مرض الحصبة
21	What are the kinds of cancer disease?	ما اصناف مرض السرطان؟
22	What is cancer disease Labels?	ما هي تسميات مرض السرطان؟
23	How to diagnose cancer disease and what is the treatment methods?	كيف يشخص مرض السرطان وما هي طرق علاجه؟
24	What are the causes of AIDS and how is it diagnosed?	ما اسباب مرض الايدز وكيف يتم تشخيصه؟
25	What are diseases that treated with antibiotics?	ما هي الامراض التي تعالج بالمضادات الحيوية؟
26	What is the disease that infects the brain and causes the infection?	ما هو المرض الذي يصيب الدماغ ومن اسبابها العدوى؟
27	How to diagnose plague disease?	كيف يشخص مرض الطاعون؟
28	Mention the name of a disease which symptoms are normal and malignant tumors?	اذكر اسم مرض من اعراضه الأورام العادية والخبيثة؟
29	What is the name of organ which is infected by meningitis ?	ما اسم العضو الذي يصاب بمرض التهاب السحايا؟
30	What is the disease that treated by insulin and caused by genetics?	ما المرض الذي يعالج بالأنسولين وسببه الوراثة؟

In general, we tested the system with 30 different NL questions. These 30 questions were carefully chosen to test the different routes of the algorithm used to translate NL queries to SPARQL. The questions vary in complexity and structure: For example, some questions can directly map to RDF triples (e.g. What are the diseases that infect blood? "ما الأمراض التي تصيب الدم؟"), other questions do not directly map to RDF triples, and thus the system should infer missing or implied components before being able to transform the queries (e.g. What is the cause of trachoma? " ما سبب الرمد الحبيبي؟ "). We also set some complex questions consisting of multiple queries linked with conjunctions (e.g. What diseases infect the blood and their symptom is high temperature? "ما الأمراض التي تصيب الدم ومن أعراضها ارتفاع درجة الحرارة؟").

The variety of the testing question set ensures to assess the different aspects of the system. Table 6.4 shows the questions that have been tested and the generated SPARQL queries.

Table 6.4: Questions that have been tested and the results

Q1: What are the symptoms of a tuberculosis?	**السؤال 1 : ما أعراض مرض السل؟**
SELECT ?dx362 WHERE ?dx362 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#الاعراض> . ?dx362 <http://www.owl-ontologies.com/Ontology1377894709.owl#اعراض_ل> ?xer72 . ?xer72 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?xer72 <http://www.owl-	

ontologies.com/Ontology1377894709.owl#اسم> "السل"	
Q2: What organs that are infected by tuberculosis?	السؤال 2 : ما هي الاعضاء التي يمكن ان تصاب بالتدرن؟
SELECT ?gvb49 WHERE ?gvb49 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#العضو> . ?gvb49 <http://www.owl-ontologies.com/Ontology1377894709.owl#يصاب_ب> ?hjr73 . ?hjr73 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ? hjr73 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "التدرن"	
Q3: What are the varieties of diabetes ?	السؤال 3 : ما هي اصناف مرض السكري؟
SELECT ?xdft3 WHERE ?xdft3 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التصنيف> . ?n2w46 <ontologies.com/Ontology1377894709.owl#يصنف> ?xdft3. ?n2w46 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?n2w46 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "السكري"	
Q4: What is the cause of trachoma?	السؤال 4 : ما سبب الرمد الحبيبي ؟
SELECT ?rdf3r WHERE ?rdf3r rdf:type < http://www.owl-ontologies.com/Ontology1377894709.owl#المسببات> . ?rdf3r <http://www.owl-ontologies.com/Ontology1377894709.owl#تسبب> ?trf18 . ?trf18 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?trf18 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الرمد الحبيبي"	
Q5: How to diagnose thalassemia?	السؤال 5 : كيف يشخص مرض الثلاسيميا ؟
SELECT ?mz32s WHERE ?mz32s rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التشخيص> . ?mz32s <http://www.owl-ontologies.com/Ontology1377894709.owl#يشخص> ?k2dui . ?k2dui rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?k2dui <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الثلاسيميا"	
Q6: What is gout?	السؤال 6 : ما هو مرض النقرس؟
SELECT ?f3ee6 WHERE ?f3ee6 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?f3ee6 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "النقرس"	
Q7: Mention the names of infectious diseases?	السؤال 7 : اذكر أسماء الأمراض المعدية ؟
SELECT ?xh5re WHERE ?k37ud <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> ?xh5re . ?k37ud rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التصنيف> . ?k37ud <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "مرض معد"	
Q8: What are the diseases that infect blood?	السؤال 8 : ما هي الامراض التي تصيب الدم ؟
SELECT ?oe53y WHERE ?oe53y rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?oe53y <http://www.owl-ontologies.com/Ontology1377894709.owl#يصيب> ?bs26i . ?bs26i rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#العضو> . ? bs26i <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الدم"	
Q9: What is a cure of meningitis?	السؤال 9 : ما علاج مرض التهاب السحايا ؟
SELECT ?apdd6 WHERE ?apdd6 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#العلاج> . ?apdd6 <http://www.owl-ontologies.com/Ontology1377894709.owl#يعالج> ?ci7yh . ?ci7yh rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ? ci7yh <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "السحايا التهاب"	
Q10: What are diseases which their symptoms is high temperature?	السؤال 10 : ما الامراض التي من اعراضها ارتفاع درجة الحرارة؟
SELECT ?kf83e WHERE ?kf83e rdf:type <http://www.owl-	

ontologies.com/Ontology1377894709.owl#المرض> . ?kf83e <http://www.owl-ontologies.com/Ontology1377894709.owl#له_اعراض> ?e5i4 . ?e5i4 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#الاعراض> . ?e5i4 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "ارتفاع درجة الحرارة"	
Q11: What are diseases that infect the blood and their symptom is high temperature?	**السؤال 11 : ما الأمراض التي تصيب الدم ومن أعراضها ارتفاع درجة الحرارة؟**
SELECT ?xie3l WHERE ?xie3l rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?xie3l <http://www.owl-ontologies.com/Ontology1377894709.owl#يصيب> ?f9u50 . ?f9u50 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#العضو> . ?f9u50 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الدم" . ?xie3l <http://www.owl-ontologies.com/Ontology1377894709.owl#له_اعراض> ?ymws7 . ?ymws7 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#الاعراض> . ?ymws7 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "ارتفاع درجة الحرارة"	
Q12: What are the infectious diseases?	**السؤال 12 : ما هي الامراض المعدية؟**
SELECT ?oe4r3 WHERE ?oe4r3 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التصنيف> . ?oe4r3 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "مرض معد"	
Q13: What is the flu disease and what are their symptoms?	**السؤال 13: ما هو مرض الزكام وماهي اعراضه؟**
SELECT ?xner9 ?gn4op WHERE ?xner9 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?xner9 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الزكام" . ?gn4op rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#الاعراض> . ?gn4op <http://www.owl-ontologies.com/Ontology1377894709.owl#اعراض_ل> ?xner9	
Q14: How is cured cholera ?	**السؤال 14 : كيف يعالج مرض الكوليرا؟**
SELECT ?r71pv WHERE ?r71pv rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#العلاج> . ?r71pv <http://www.owl-ontologies.com/Ontology1377894709.owl#يعالج> ?bne3p . ?bne3p rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ? bne3p <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الكوليرا" .	
Q15: What is the disease that infects the intestines and their symptoms are vomiting and diarrhea?	**السؤال 15 : ما هو المرض الذي يصيب الامعاء ومن اعراضه القيء والاسهال؟**
SELECT ?x91nc WHERE ?x91nc rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?x91nc <http://www.owl-ontologies.com/Ontology1377894709.owl#يصيب> ?qpst3 . ?qpst3 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#العضو> . ?qpst3 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الامعاء" . ?x91nc <http://www.owl-ontologies.com/Ontology1377894709.owl#له_اعراض> ?e4ht1 . ?e4ht1 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#الاعراض> . ?e4ht1 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "القيء" . ?x91nc <http://www.owl-ontologies.com/Ontology1377894709.owl#له_اعراض> ?ms7ur . ?ms7ur rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#الاعراض> . ?ms7ur <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الاسهال"	
Q16: Describe the malaria?	**السؤال 16 : اوصف مرض الملاريا؟**
SELECT ?f8izv WHERE ?e3jqd <http://www.owl-ontologies.com/Ontology1377894709.owl#وصف> ?f8izv . ?e3jqd rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?e3jqd <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "ملاريا"	

Q17: What is the classification of asthma?	السؤال 17 : ما هو تصنيف مرض الربو؟
SELECT ?k1w9s WHERE ?k1w9s rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التصنيف> . ?gwl5a <http://www.owl-ontologies.com/Ontology1377894709.owl#يصنف> ?k1w9s . ?gwl5a rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?gwl5a <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الربو"	
Q18: How is the diagnosis of asthma?	السؤال 18 : كيف يتم تشخيص مرض الربو؟
SELECT ?ipe4u WHERE ?ipe4u rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التشخيص> . ?ipe4u <http://www.owl-ontologies.com/Ontology1377894709.owl#يشخص> ?fb6ys . ?fb6ys rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?fb6ys <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الربو"	
Q19: Mention other names to AIDS?	السؤال 19 : اذكر تسميات اخرى لمرض الايدز؟
SELECT ?hdd4c WHERE ?nc8ee <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> ? hdd4c . ? nc8ee rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?k37ud <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الإيدز"	
Q20: What are the causes of measles?	السؤال 20 : ما هي اسباب مرض الحصبة ؟
SELECT ?ps5r2 WHERE ?ps5r2 rdf:type < http://www.owl-ontologies.com/Ontology1377894709.owl#المسببات> . ?ps5r2 <http://www.owl-ontologies.com/Ontology1377894709.owl#تسبب> ?a3nnd . ?a3nnd rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?a3nnd <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الحصبة"	
Q21: What are the kinds of cancer disease?	السؤال 21 : ما اصناف مرض السرطان؟
SELECT ?tlf59 WHERE ?tlf59 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التصنيف> . ?x1ek6 <http://www.owl-ontologies.com/Ontology1377894709.owl#يصنف> ?tlf59 . ?x1ek6 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?x1ek6 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "السرطان"	
Q22: What is cancer disease Labels?	السؤال 22 : ما هي تسميات مرض السرطان؟
SELECT ?hdd4c WHERE ?nc8ee <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> ? hdd4c . ? nc8ee rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?k37ud <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "السرطان"	
Q23: How to diagnose cancer disease and what is the treatment methods?	السؤال 23 : كيف يشخص مرض السرطان وما هي طرق علاجه؟
SELECT ?uf5np ?x2fto WHERE ?uf5np rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التشخيص> . ?uf5np <http://www.owl-ontologies.com/Ontology1377894709.owl#يشخص> ?xjnc3 . ?xjnc3 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?xjnc3 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "السرطان" . ?x2fto rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#العلاج> . ?x2fto <http://www.owl-ontologies.com/Ontology1377894709.owl#يعالج> ?uf5np	
Q24: What are the causes of AIDS and how is it diagnosed?	السؤال 24 : ما اسباب مرض الايدز وكيف يتم تشخيصه؟
SELECT ?flz4t ?stu8v WHERE ?flz4t rdf:type < http://www.owl-ontologies.com/Ontology1377894709.owl#المسببات> . ?flz4t <http://www.owl-ontologies.com/Ontology1377894709.owl#تسبب> ?bqms2 . ?bqms2 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?bqms2 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الايدز" . ?stu8v rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التشخيص> . ?stu8v <http://www.owl-	

ontologies.com/Ontology1377894709.owl#يشخص> ?flz4t	
Q25: What are diseases that treated with antibiotics?	السؤال 25 : ما هي الامراض التي تعالج بالمضادات الحيوية؟
SELECT ?s7uwm WHERE ?s7uwm rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?s7uwm < http://www.owl-ontologies.com/Ontology1377894709.owl#يعالج_ب> ?kd4rm . ?kd4rm rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#العلاج> . ?kd4rm <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "المضادات الحيوية"	
Q26: What is the disease that infects the brain and causes the infection?	السؤال 26 : ما هو المرض الذي يصيب الدماغ ومن اسبابها العدوى؟
SELECT ?xcu2w WHERE ?xcu2w rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?xcu2w <http://www.owl-ontologies.com/Ontology1377894709.owl#يصيب> ?vpr5t . ?vpr5t rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#العضو> . ?vpr5t <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الدماغ" . xcu2w <http://www.owl-ontologies.com/Ontology1377894709.owl#بسبب> ?fnb6s . ?fnb6s rdf:type < http://www.owl-ontologies.com/Ontology1377894709.owl#المسببات> .?fnb6s <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "العدوى"	
Q27: How to diagnose plague disease?	السؤال 27 : كيف يشخص مرض الطاعون؟
SELECT ?x3lqv WHERE ?x3lqv rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التشخيص> . ?js8ml rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?js8ml <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الطاعون" . ?x3lqv <http://www.owl-ontologies.com/Ontology1377894709.owl#يشخص> ?js8ml	
Q28: Mention the name of a disease which symptoms are normal and malignant tumors?	السؤال 28 : اذكر اسم مرض من اعراضه الأورام العادية والخبيثة؟
SELECT ?xp9ns WHERE ?eiu19 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> ?xp9ns . ?eiu19 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?eiu19 <http://www.owl-ontologies.com/Ontology1377894709.owl#له_اعراض> ?bx1qa . ?bx1qa rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#الاعراض> . ?bx1qa <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الأورام العادية والخبيثة"	
Q29: What is the name of organ which is infected by meningitis ?	السؤال 29 : ما اسم العضو الذي يصاب بمرض التهاب السحايا؟
SELECT ?yev6f WHERE ?xdb8w <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> ?yev6f . ?xdb8w rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#العضو> . ?ka5i1 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?xdb8w <http://www.owl-ontologies.com/Ontology1377894709.owl#يصاب_ب> ?ka5i1 . ?ka5i1 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "التهاب السحايا"	
Q30: What is the disease that treated by insulin and caused by genetics ?	السؤال 30 : ما المرض الذي يعالج بالأنسولين وسببه الوراثة؟
SELECT ?mde4i WHERE ?mde4i rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?mde4i <http://www.owl-ontologies.com/Ontology1377894709.owl#يعالج_ب> ?fxe0h . ?fxe0h rdf:type < http://www.owl-ontologies.com/Ontology1377894709.owl#العلاج>. ?fxe0h <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الانسولين" . ?mde4i <http://www.owl-ontologies.com/Ontology1377894709.owl#بسبب> ?ej2f9 . ?ej2f9 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المسببات> . ?ej2f9 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "الوراثة"	

Table 6.4 displays the testing results. Results in red are incorrect.

6.4 Preliminary Evaluation

In this section, we present the preliminary evaluation of our system to determine whether it behaves exactly as we expect. The main goal of this evaluation is to assess the system's ability to translate NL queries to valid SPARQL queries and retrieve answers from the RDF knowledge base.

System evaluation depends on rules and procedures in our approach of interpreting the Arabic NL queries to SPARQL queries and finding correct answers for questions related to the ontology domain. We have tested 30 questions in our related Arabic Ontology Domain "علم الامراض" (Pathology). We verify that the goal has been performed according to the rules and procedures that we have set in our thesis. Table 6.5 depicts the number of questions and their results.

Table 6.5: The number of questions and their results

Total questions	Correct Results	Incorrect Results
30	**28**	**2**

As for comparison and contrast with other methods, we are not aware of any previous work that uses ontologies for Arabic Question Answering System. Therefore, we cannot compare or contrast our methodology with other researches.

6.5 Discussion

The proposed system has been tested with a sample ontology and a testing set consisting of 30 different questions. Results has shown that the system can correctly answer 28 out of the 30 questions. Table 6.6. shows the Questions that have been tested and the rules that have been applied.

Table 6.6. Testing questions and rules that are applied

Questions that have been tested and the rules that have been applied	
Q2: What organs that are infected by tuberculosis?	السؤال 2 : ما هي الاعضاء التي يمكن ان تصاب بالتدرن؟
Q29: What is the name of organ which is infected by meningitis ?	السؤال 29 : ما اسم العضو الذي يصاب بمرض التهاب السحايا؟
Q8: What are the diseases that infect blood?	السؤال 8 : ما هي الامراض التي تصيب الدم ؟
Q25: What are diseases that treated with antibiotics?	السؤال 25 : ما هي الامراض التي تعالج بالمضادات الحيوية؟
Procedure: Interpretation of NL queries that map to complete triple patterns **Rules:**	

Rule 1: $\rho(x)$ $where$ $x \in (P \cup I \cup L) \Rightarrow x$ ***Rule 2:*** $\rho(C) \Rightarrow ?var\ a\ C$ *where the property a is equivalent to the property rdf:type* ***Rule 3:*** $\rho(s,p,o) \Rightarrow \rho(s) \wedge \rho(p) \wedge \rho(o)$ $where\ s \in (C \cup I), p \in P, o \in (C \cup I \cup L), s \in Domain\ of\ P, o \in Range\ of\ P$	
Q20: What are the causes of measles?	السؤال 20 : ما هي اسباب مرض الحصبة ؟
Q21: What are the kinds of cancer disease?	السؤال 21 : ما اصناف مرض السرطان؟
Q10: What are diseases which their symptoms is high temperature?	السؤال 10 : ما الامراض التي من اعراضها ارتفاع درجة الحرارة؟
Q28: Mention the name of a disease which symptoms are normal and malignant tumors?	السؤال 28 : اذكر اسم مرض من اعراضه الأورام العادية والخبيثة؟
Q1: What are the symptoms of tuberculosis?	السؤال 1 : ما أعراض مرض السل؟
Q3: What are the varieties of diabetes?	السؤال 3 : ما هي اصناف مرض السكري؟
Q9: What is a cure of meningitis?	السؤال 9 : ما علاج مرض التهاب السحايا ؟
Q17: What is the classification of asthma?	السؤال 17 : ما هو تصنيف مرض الربو؟
Procedure: Interpretation of NL queries that do not map to complete triple patterns **Omitted Predicate** **Case 1**: A class followed by another class To identify the missing property, the following SPARQL query is executed over the ontology: SELECT DISTINCT ?property WHERE {{?property rdfs:domain ?C1 . ?property rdfs:range ?C2} UNION {?property rdfs:domain ?C2 . ?property rdfs:range ?C1}}	
Q19: Mention other names to AIDS?	السؤال 19 : اذكر تسميات اخرى لمرض الايدز؟
Q22: What is cancer disease Labels?	السؤال 22 : ما هي تسميات مرض السرطان؟
Q4: What is the cause of trachoma?	السؤال 4 : ما سبب الرمد الحبيبي ؟
Q16: Describe the malaria?	السؤال 16 : اوصف مرض الملاريا؟
Q6: What is gout?	السؤال 6 : ما هو مرض النقرس؟
Procedure: Interpretation of NL queries that do not map to complete triple patterns **Omitted Predicate** **Case 2**: A class followed by an instance of a class (individual) Given a class C and individual I, we execute the following SPARQL query on the knowledge base to retrieve properties that link class C to instance I: SELECT DISTINCT ?property WHERE {{ ?x ?property I } UNION { I ?property ?x } . ?x rdf:type C }	
Q18: How is the diagnosis of asthma?	السؤال 18 : كيف يتم تشخيص مرض الربو؟
Q27: How to diagnose plague disease?	السؤال 27 : كيف يشخص مرض الطاعون؟
Q5: How to diagnose thalassemia?	السؤال 5 : كيف يشخص مرض الثلاسيميا ؟
Q14: How is cured cholera ?	السؤال 14 : كيف يعالج مرض الكوليرا؟
Procedure : Interpretation of NL queries that do not map to complete triple patterns **Omitted Subject** **Case 1**: An object property followed by a class (or individual): Given an object O of type C, and a property P, the following query is used to identify the missing domain of the property: SELECT DISTINCT ?SUBJECT_Class WHERE {{?property rdfs:domain ?SUBJECT_CLASS . ?property rdfs:range C .} UNION {?property rdfs:range ?SUBJECT_CLASS . ?property rdfs:domain C .}}	
Q26: What is a disease that infects the brain and causes the infection?	السؤال 26 : ما هو المرض الذي يصيب الدماغ ومن اسبابها العدوى؟
Q30: What is a disease that treated by insulin and caused by genetics ?	السؤال 30 : ما المرض الذي يعالج بالأنسولين وسببه الوراثة؟

Q11: What are diseases that infect the blood and their symptom is high temperature? Q15: What is a disease that infects the intestines and their symptoms are vomiting and diarrhea? Q13: What is the flu disease and what are their symptoms? Q24: What are the causes of AIDS and how is it diagnosed?	السؤال 11 : ما الأمراض التي تصيب الدم ومن أعراضها ارتفاع درجة الحرارة؟ السؤال 15 : ما هو المرض الذي يصيب الامعاء ومن اعراضه القيء والاسهال؟ السؤال 13: ما هو مرض الزكام وماهي اعراضه؟ السؤال 24 : ما اسباب مرض الايدز وكيف يتم تشخيصه؟
Procedure: Interpretation of NL queries that consist of multiple queries linked with conjunctions. **Rule 3:** *Dependent clause is a complete triple pattern => add a new triple pattern to link* $\boldsymbol{Subject_v}$ *with* $\boldsymbol{Subject_d}$	

In the following discussion, we pick two examples NL queries that were correctly translated to SPARQL and show how the system translated them. The queries that the system could not correctly translate are also explained in detail.

Example 1: The question no. 25

What are the diseases treated with antibiotics? "ما هي الامراض التي تعالج بالمضادات الحيوية؟" shown in Table 6.4, applied rules in Section 5.2, as follows:

Step 1: The output of the mapping process is: <:Disease (class), :cured_by (object_property), : Antibiotics (instance) >.

تعالج_ب is matched with "cures" or "cured_by" (ambiguity). To resolve this ambiguity, the system refers to the subject (the ontology term that precedes the property term) and the object (the term that follows the property term) of the triple and then chooses the property whose domain and range fulfill the given subject and object. For example, the domain and range of the property ":cures" include the term ":Antibiotics" and "Disease" respectively. Therefore, the property ":cures" is excluded because it does not fulfill the given subject and object. The property ":curedBy" is the valid choice because its domain includes the class ":Disease", and its range includes the class "Antibiotics"

Step 2: The ontology terms are scanned for a complete triple pattern: A subject (:Disease), a predicate (:cured_by) and an object (:Antibiotics) appear in sequence. The class :Disease and the object : Antibiotics both fulfill the condition that they belong to the domain and range of the property : cured_by respectively. Thus, a complete triple pattern is identified.

Step 3: The target Rule 1, 2 and 3 are applied as follows:

$$\rho(:\text{Disease}, :\text{cured\_by}, :\text{Antibiotics}) \Rightarrow \rho(:\text{Disease}) \wedge \rho(:\text{cured\_by}) \wedge \rho(:\text{Antibiotics})$$

=>?var rdf:type :Disease . ?var : cured_by : Antibiotics

Step 4: The SPARQL query after generating the SELECT clause becomes:

SELECT DISTINCT ?var WHERE ?var rdf:type :Disease . ?var : cured_by: Antibiotics

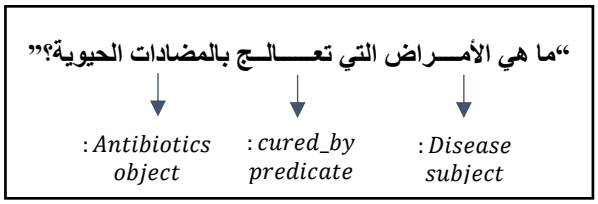


Figure 6.2: Triple pattern of the question no. 25

Example 2: The question no. 30

What disease that treated by insulin and caused by genetics? "ما المرض الذي يعالج بالأنسولين وسببه الوراثة؟" shown in Table 6.4 consists of multiple queries connected with conjunctions (و). So it applied rules in Section 5.4 Case1 as follows:

Step 1: The NL query is broken into clauses based on the conjunctions.

Step 2: Find the pivot clause from the clauses resulted from Step 1.

The pivot clause is "المرض الذي يعالج بالأنسولين" because it maps to a complete triple pattern <:Disease, :cured_by, : Insulin>.

Step 3: Determining dependencies in the NL query and replacing missing components. In this question we have two complete triple patterns connected with a conjunction, so we applied the following rule:

Dependent clause is a complete triple pattern => add a new triple pattern to link $Subject_v$ *with* $Subject_d$

The pivot clause "المرض الذي يعالج بالأنسولين" corresponds to complete triple pattern <:Disease, : cured_by , : Insulin >.

The dependent clause "سببه الوراثة" maps to the ontology terms, where the word "سبب" maps the class "Reason", the word "الوراثة" maps to the predefined instance of the class "Reason" and the property is missing. Given a class C and individual I, We execute the following SPARQL query on the knowledge base to retrieve a valid property that links class C to instance I:

SELECT DISTINCT ?property WHERE {{ ?x ?property I } UNION { I ?property ?x } . ?x rdf:type C }

The above query retrieves the property ":hasName", so the triple patterns of the dependent clause "سببه الوراثة" become <:Reason, :hasName, : heredity >.

Step 4: The next step to interpret into SPARQL by creating a new triple is to act as a connector. The connecting triple pattern is <:Disease, ??, :Reason>. Mapping these words to the ontology will produce: (:Disease "Class", :Reason "Class"). The missing predicate should belong to its domain and range respectively. To identify the missing property, the following SPARQL query is executed over the ontology:

SELECT DISTINCT ?property WHERE {{?property rdfs:domain ?C1 . ?property rdfs:range ?C2} UNION {?property rdfs:domain ?C2 . ?property rdfs:range ?C1}}

The above query retrieves the property ":CausedBy" which is the right predicate.

The final resultant SPARQL query will be:

SELECT DISTINCT ?var1 WHERE {?var1 a :Disease . ?var1 :cured_by :Insulin. ?var2 a : Reason. ?var2 :hasName "heredity" . ?var1 :CausedBy ?var2}

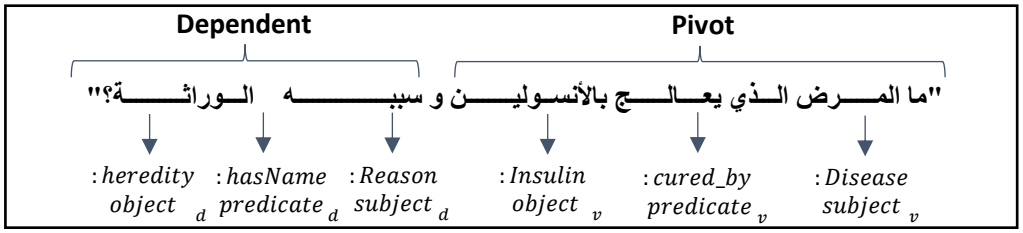


Figure 6.3: Triple pattern of the question no.30

Two of the given 30 questions were not correctly answered. In the following part, we explain why the system cannot translate these questions to SPARQL:

In question no. 7

Mention the names of infectious diseases? "اذكر أسماء الأمراض المعدية؟"

This result of the mapping process is as follows:

<:اسم (Data-type Property), :مرض معدي (:instance of the class "تصنيف")>

If the query starts with a data-type property followed by a class (or an instance), equivalent the SPARQL query should be:

<: مرض معدي (Class), :اسم (Data-type Property) , ?>

Although the generated SPARQL query was correct, the system could not provide an answer. This is because the instance "مرض معدي:" was not associated with the data-type property "اسم" in the RDF store. This result indicates that the coverage of the ontology and the associated annotations is very important to answer user queries.

In question no. 23

How to diagnose cancer and what treatment methods? " كيف يشخص مرض السرطان وما هي طرق علاجه؟"

SELECT ?uf5np ?x2fto WHERE ?uf5np rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التشخيص> . ?uf5np <http://www.owl-ontologies.com/Ontology1377894709.owl#يشخص> ?xjnc3 . ?xjnc3 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?xjnc3 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "السرطان" . ?x2fto rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#العلاج> . ?x2fto <http://www.owl-ontologies.com/Ontology1377894709.owl#يعالج> ?uf5np

This question is translated as follows:

- The first clause: كيف يشخص مرض السرطان:

<? ، يشخص: , : مرض> ➔ missing subject ➔ Use Case 1 in Section 5.3.2 to identify the missing subject

<مرض:, ? , السرطان> ➔ missing property ➔ Use Case 2 in Section 5.3.1 to identify the missing property. After identifying missing parts, this clause is translated as follows:

?uf5np rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#التشخيص> . ?uf5np <http://www.owl-ontologies.com/Ontology1377894709.owl#يشخص> ?xjnc3 . ?xjnc3 rdf:type <http://www.owl-ontologies.com/Ontology1377894709.owl#المرض> . ?xjnc3 <http://www.owl-ontologies.com/Ontology1377894709.owl#اسم> "السرطان"

- The second clause: وما طرق علاجه ؟

The only word that maps to the ontology terms is the word "علاجه". We still need two other terms to have a complete triple which cannot be identified. The only way to identify the object of the triple is to determine the correct entity which the pronoun "علاجه" refers to. However, in our approach we did not use any means for co-reference resolution. Therefore, the second clause was incorrectly translated.

The above results highlight some of the system strengths and limitations which we can summarize as follows:

- The system has the ability to translate simple and complex NL queries to SPARQL and to identify the missing parts of RDF triples using the grammar rules we defined.
- The system relies heavily on the ontology knowledge to translate the query, and it makes less use of NLP techniques. For example, the system uses the domain and range of properties to resolve ambiguity, retrieve correct relations and identify dependencies between clauses linked by conjunctive words. However, the lack of NLP processing may result in incorrect translations such as the lack of co-reference resolution which is important to identify expressions and pronouns that map to a single entity.
- While the lack of NLP-based processing can be seen as a limitation, we think that it is also an important strength point considering the limited support for Arabic NLP in comparison with the NLP of Latin- and English-based scripts. Existing tools for Arabic NLP are often error-prone and less accurate than their counterparts in English. Therefore, we made the decision to rely heavily on the ontology semantics to answer NL queries. However, this decision can be revised in the future according to the advancement in the Arabic NLP tools.

Note also that the ability of the system to provide answers depends on the coverage of the ontology and the association knowledge base. The more complete the ontology in representing the domain of knowledge and expressing the different relationships between the domain terms, the more the system becomes capable of proving correct answers. Therefore, we think that the inability to answer NL queries due to limited coverage cannot be considered as a limitation in the system.

In our future work, we aim to test the system with different domains of knowledge to ensure its portability feature (i.e. the ability to interface to different ontologies). We will also test it with users and human experts. We will also try to identify more rules for more complex NL queries.

We also think the combination of linguistic and semantic processing is the best approach to answer NL query. Therefore, we will assess the latest advances in Arabic NLP to explore to what extent NLP techniques can contribute to the system.

6.6 Summary

This chapter, talked about realization, experimental results and evaluation of the proposed system. The first section explained the components that are used in our Arabic

QA System and showed the role of each component. In the second section, we presented the tools and programs used in our work. The third section presented the questions that have been tested and the results that generated from the system. The fourth section presented the evaluation for our model showing that the system can correctly answer 28 out of the 30 questions. The fifth section discussed the results.

Chapter 7: Conclusion and Future Work

In this research, we have developed a model for Arabic question answering system based on Ontology that transforms NL queries in Arabic to SPARQL queries. The system is designed to be ontology-portable. This ontology-based model uses ontology components for matching with user queries. Our model consists of several components which are Data Processing, Question Processing, Ontology Mapper and Answer Retrieval.

The process of constructing a SPARQL query from the NL query contains procedures based on the complexity and completeness of the NL query. Also, rules were defined to capture dependencies between the query terms. We support that with examples that cover all different types of procedures.

Experiments were performed to test the system for interpreting NL queries in Arabic to SPARQL queries. The questions that have been tested depend on the data in ontology domain. In the evaluation process, the results that generated from the system show that the system can correctly answer 28 out of the 30 questions.

The main contribution of this research is that using the ontology can support the process of answering the question in resolving word disambiguation and identifying intelligent answers.

Since only a prototype of the proposed system is implemented, in a future work we look forward to implementing a complete system. Success of our proposed prototype encourages us to look for ways to increase the scope of this research to include more types of questions such as comparative and similarity phrases. In addition, we look to extend the ontology by adding more data and semantic information. Also, we look forward to increasing our rules and procedures of the NL query to cover the largest amount of questions and obtain more accurate results.

References:

[1] P. Rosso, Y. Benajiba, and A. Lyhyaoui, "Towards an Arabic question answering system," in *Proc. 4th Conf. on Scientific Research Outlook & Technology Development in the Arab world, SROIV, Damascus, Syria*, 2006, pp. 11-14.

[2] B. Gorenjak, M. Ferme, and M. Ojsteršek, "A Question Answering System on Domain Specific Knowledge with Semantic Web Support."

[3] M. R. Kangavari, S. Ghandchi, and M. Golpour, "Information retrieval: improving question answering systems by query reformulation and answer validation," *World Academy of Science, Engineering and Technology*, vol. 48, pp. 303-310, 2008.

[4] M. Shaheen and A. M. Ezzeldin, "Arabic Question Answering: Systems, Resources, Tools, and Future Trends," *Arabian Journal for Science and Engineering*, pp. 1-24, 2014.

[5] W. Brini, M. Ellouze, O. Trigui, S. Mesfar, H. Belguith, and P. Rosso, "Factoid and definitional Arabic question answering system," *Post-Proc. NOOJ-2009, Tozeur, Tunisia, June*, pp. 8-10, 2009.

[6] N. F. Noy and D. L. McGuinness, "Ontology development 101: A guide to creating your first ontology," ed: Stanford knowledge systems laboratory technical report KSL-01-05 and Stanford medical informatics technical report SMI-2001-0880, 2001.

[7] Q. Guo and M. Zhang, "Question answering system based on ontology and semantic web," in *Rough Sets and Knowledge Technology*, ed: Springer, 2008, pp. 652-659.

[8] W3C. *Resource Description Framework* Available: http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/

[9] B. Gorenjak, M. Ferme, and M. Ojstersek, "A question answering system on domain specific knowledge with semantic web support," 2011.

[10] O. Trigui, H. Belguith, and P. Rosso, "DefArabicQA: Arabic definition question answering system," in *Workshop on Language Resources and Human Language Technologies for Semitic Languages, 7th LREC, Valletta, Malta*, 2010, pp. 40-45.

[11] R. Mihalcea, H. Liu, and H. Lieberman, "NLP (natural language processing) for NLP (natural language programming)," in *Computational Linguistics and Intelligent Text Processing*, ed: Springer, 2006, pp. 319-330.

[12] G. Gangwal, "Question Answering System using Open Source Software," 2012.

[13] I. A. Al-Sughaiyer and I. A. Al-Kharashi, "Arabic morphological analysis techniques: A comprehensive survey," *Journal of the American Society for Information Science and Technology*, vol. 55, pp. 189-213, 2004.

[14] R. Al-Shalabi, G. Kanaan, B. Al-Sarayreh, K. Khanfer, A. Al-Ghonmein, H. Talhouni, *et al.*, "Proper noun extracting algorithm for arabic language," in *International conference on IT, Thailand*, 2009.

[15] B. Hammo, H. Abu-Salem, and S. Lytinen, "QARAB: A question answering system to support the Arabic language," in *Proceedings of the ACL-02 workshop on Computational approaches to semitic languages*, 2002, pp. 1-11.

[16] R. Sonbol, N. Ghneim, and M. S. Desouki, "An Application Oriented Arabic Morphological Analyzer," in *Proceedings of the workshop of morphological analyzer experts for Arabic language, organized by Arab League Educational, Cultural and Scientific Organization (ALECSO), King Abdul-Aziz City of Science and Technology (KACST) and Arabic Language Academy*, 2009.

[17] H. Al Ameed, S. Al Ketbi, A. Al Kaabi, K. Al Shebli, N. Al Shamsi, N. H. Al Nuaimi, *et al.*, "Arabic light stemmer: A new enhanced approach," in *The Second International Conference on Innovations in Information Technology (IIT'05)*, 2005.
[18] S. Khoja, "Stemming Arabic Text,"" http://zeus.cs.pacificu.edu/shereen/research.htm, checked Jan 6th, 2015.
[19] S. Khoja, "APT: Arabic part-of-speech tagger," in *Proceedings of the Student Workshop at NAACL*, 2001, pp. 20-25.
[20] A. Aliwy, "Tokenization as Preprocessing for Arabic Tagging System," *International Journal of Information and Education Technology: IJIET*, vol. 348, 2012.
[21] A. Ezzeldin and M. Shaheen, "A Survey of Arabic question answering: challenges, tasks, approaches, tools, and future trends," in *Proceedings of The 13th International Arab Conference on Information Technology (ACIT 2012)*, 2012, pp. 1-8.
[22] P. Gupta and V. Gupta, "A survey of text question answering techniques," *International Journal of Computer Applications*, vol. 53, pp. 1-8, 2012.
[23] S. Kalaivani and K. Duraiswamy, "Comparison of Question Answering Systems Based on Ontology and Semantic Web in Different Environment," *Journal of Computer Science*, vol. 8, 2012.
[24] Y. Benajiba and P. Rosso, "Arabic question answering," *Diploma of advanced studies. Technical University of Valencia, Spain*, 2007.
[25] M. E. Sucunuta and G. E. Riofrio, "Architecture of a question-answering system for a specific repository of documents," in *Software Technology and Engineering (ICSTE), 2010 2nd International Conference on*, 2010, pp. V2-12-V2-16.
[26] L. Hirschman and R. Gaizauskas, "Natural language question answering: the view from here," *Natural Language Engineering*, vol. 7, pp. 275-300, 2001.
[27] F. Mohammed, K. Nasser, and H. Harb, "A knowledge based Arabic question answering system (AQAS)," *ACM SIGART Bulletin*, vol. 4, pp. 21-30, 1993.
[28] J. Z. Pan and I. Horrocks, "Rdfs (fa): connecting rdf (s) and owl dl," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 19, pp. 192-206, 2007.
[29] V. Jain and M. Singh, "Ontology Development and Query Retrieval using Protégé Tool," *International Journal of Intelligent Systems and Applications (IJISA)*, vol. 5, p. 67, 2013.
[30] S. Ou, C. Orasan, D. Mekhaldi, and L. Hasler, "Automatic Question Pattern Generation for Ontology-based Question Answering," in *FLAIRS Conference*, 2008, pp. 183-188.
[31] D. Damljanovic, V. Tablan, and K. Bontcheva, "A Text-based Query Interface to OWL Ontologies," in *LREC*, 2008.
[32] S. Boyce and C. Pahl, "Developing domain ontologies for course content," *Educational Technology & Society-ETS*, vol. 10, pp. 275-288, 2007.
[33] Wikipedia. Available: http://en.wikipedia.org/wiki/SPARQL
[34] L. McCarthy, B. Vandervalk, and M. Wilkinson, "SPARQL Assist language-neutral query composer," *BMC bioinformatics*, vol. 13, p. S2, 2012.
[35] E. Kaufmann, A. Bernstein, and R. Zumstein, "Querix: A natural language interface to query ontologies based on clarification dialogs," in *5th International Semantic Web Conference (ISWC 2006)*, 2006, pp. 980-981.
[36] C. Wang, M. Xiong, Q. Zhou, and Y. Yu, "Panto: A portable natural language interface to ontologies," in *The Semantic Web: Research and Applications*, ed: Springer, 2007, pp. 473-487.
[37] V. Lopez, V. Uren, E. Motta, and M. Pasin, "AquaLog: An ontology-driven question answering system for organizational semantic intranets," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, pp. 72-105, 2007.

[38] P. Cimiano, P. Haase, and J. Heizmann, "Porting natural language interfaces between domains: an experimental user study with the orakel system," in *Proceedings of the 12th international conference on Intelligent user interfaces*, 2007, pp. 180-189.
[39] O. Ferrández, R. Izquierdo, S. Ferrández, and J. L. Vicedo, "Addressing ontology-based question answering with collections of user queries," *Information Processing & Management*, vol. 45, pp. 175-188, 2009.
[40] V. Tablan, D. Damljanovic, and K. Bontcheva, *A natural language query interface to structured information*: Springer, 2008.
[41] D. Damljanovic, M. Agatonovic, and H. Cunningham, "Natural language interfaces to ontologies: Combining syntactic analysis and ontology-based lookup through the user interaction," in *The semantic web: Research and applications*, ed: Springer, 2010, pp. 106-120.
[42] W. Brini, M. Ellouze, S. Mesfar, and L. H. Belguith, "An Arabic Question-Answering system for factoid questions," in *Natural Language Processing and Knowledge Engineering, 2009. NLP-KE 2009. International Conference on*, 2009, pp. 1-7.
[43] G. Kanaan, A. Hammouri, R. Al-Shalabi, and M. Swalha, "A new question answering system for the Arabic language," *American Journal of Applied Sciences*, vol. 6, p. 797, 2009.
[44] S. Bekhti, A. Rehman, M. Al-Harbi, and T. Saba, "AQUASYS: AN ARABIC QUESTION-ANSWERING SYSTEM BASED ON EXTENSIVE QUESTION ANALYSIS AND ANSWER RELEVANCE SCORING," *International Journal of Academic Research*, vol. 3, 2011.
[45] H. Abdelnasser, R. Mohamed, M. Ragab, A. Mohamed, B. Farouk, N. El-Makky, *et al.*, "Al-Bayan: An Arabic Question Answering System for the Holy Quran," *ANLP 2014*, p. 57, 2014.
[46] H. Kurdi, S. Alkhaider, and N. Alfaifi, "DEVELOPEMNT AND EVALUATION OF A WEB BASED QUESTION ANSWERING SYSTEM FOR ARABIC LANGUAG."
[47] T. Dilekh and A. Behloul, "Implementation of a New Hybrid Method for Stemming of Arabic Text," *analysis*, vol. 3, p. 5, 2012.
[48] Y. El Hadj, I. Al-Sughayeir, and A. Al-Ansari, "Arabic part-of-speech tagging using the sentence structure," in *Proceedings of the Second International Conference on Arabic Language Resources and Tools, Cairo, Egypt*, 2009.
[49] Wikipedia. (14/10/2014). Available: http://en.wikipedia.org/wiki/Stemming
[50] "QArabPro: A Rule Based Question Answering System for Reading Comprehension Tests in Arabic," 2011.
[51] Wikipedia. (15/9/2014). Available: http://en.wikipedia.org/wiki/Tokenization_(lexical_analysis)
[52] *Stanford Word Segmenter*. Available: http://nlp.stanford.edu/software/segmenter.shtml
[53] I. AlAgha "AR2SPARQL: An Arabic Natural Language Interface for the Semantic Web," *The Arabian Journal of Science and Engineering (Under Review)*.
[54] A. Farghaly and K. Shaalan, "Arabic natural language processing: Challenges and solutions," *ACM Transactions on Asian Language Information Processing (TALIP)*, vol. 8, p. 14, 2009.
[55] *Welcome to protégé.* Available: http://protege.stanford.edu/

Appendix A: OWL Source Code

```
<?xml version="1.0"?>
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
   xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns="http://www.owl-ontologies.com/Ontology1377894709.owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:swrl="http://www.w3.org/2003/11/swrl#"
   xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xml:base="http://www.owl-ontologies.com/Ontology1377894709.owl">
 <owl:Ontology rdf:about=""/>
 <owl:Class rdf:ID="الاعراض"/>
 <owl:Class rdf:ID="المرض"/>
 <owl:Class rdf:ID="العضو"/>
 <owl:Class rdf:ID="العلاج"/>
 <owl:Class rdf:ID="المسببات"/>
 <owl:Class rdf:ID="التشخيص"/>
 <owl:Class rdf:ID="التصنيف"/>
 <owl:ObjectProperty rdf:ID="يشخص_ب">
  <rdfs:domain rdf:resource="#المرض"/>
  <rdfs:range rdf:resource="#التشخيص"/>
  <owl:inverseOf>
   <owl:ObjectProperty rdf:ID="يشخص"/>
  </owl:inverseOf>
 </owl:ObjectProperty>
 <owl:ObjectProperty rdf:ID="يصنف">
  <rdfs:range rdf:resource="#التصنيف"/>
  <rdfs:domain rdf:resource="#المرض"/>
 </owl:ObjectProperty>
 <owl:ObjectProperty rdf:ID="له_اعراض">
  <owl:inverseOf>
   <owl:ObjectProperty rdf:ID="اعراض_ل"/>
  </owl:inverseOf>
  <rdfs:domain rdf:resource="#المرض"/>
  <rdfs:range rdf:resource="#الاعراض"/>
 </owl:ObjectProperty>
 <owl:ObjectProperty rdf:ID="يعالج">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#InverseFunctionalProperty"/>
  <owl:inverseOf>
   <owl:InverseFunctionalProperty rdf:ID="يعالج_ب"/>
  </owl:inverseOf>
  <rdfs:domain rdf:resource="#العلاج"/>
  <rdfs:range rdf:resource="#المرض"/>
 </owl:ObjectProperty>
```

```
<owl:ObjectProperty rdf:about="#اعراض_ل">
  <rdfs:domain rdf:resource="#الاعراض"/>
  <rdfs:range rdf:resource="#المرض"/>
  <owl:inverseOf rdf:resource="#له_اعراض"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#يشخص">
  <owl:inverseOf rdf:resource="#يشخص_ب"/>
  <rdfs:range rdf:resource="#المرض"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#InverseFunctionalProperty"/>
  <rdfs:domain rdf:resource="#التشخيص"/>
</owl:ObjectProperty>
<owl:DatatypeProperty rdf:ID="تسميات">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#المرض"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="اسم">
  <rdfs:domain>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#العضو"/>
        <owl:Class rdf:about="#المرض"/>
        <owl:Class rdf:about="#العلاج"/>
        <owl:Class rdf:about="#الاعراض"/>
        <owl:Class rdf:about="#التصنيف"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="وصف">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#التشخيص"/>
        <owl:Class rdf:about="#المسببات"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
</owl:DatatypeProperty>
<owl:FunctionalProperty rdf:ID="بسبب">
  <owl:inverseOf>
    <owl:InverseFunctionalProperty rdf:ID="تسبب"/>
  </owl:inverseOf>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:range rdf:resource="#المسببات"/>
  <rdfs:domain rdf:resource="#المرض"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="يصاب_ب">
```

```
    <rdfs:domain rdf:resource="#العضو"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:range rdf:resource="#المرض"/>
    <owl:inverseOf>
      <owl:InverseFunctionalProperty rdf:ID="يصيب"/>
    </owl:inverseOf>
  </owl:FunctionalProperty>
  <owl:InverseFunctionalProperty rdf:about="#يصيب">
    <rdfs:range rdf:resource="#العضو"/>
    <owl:inverseOf rdf:resource="#يصاب_ب"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:domain rdf:resource="#المرض"/>
  </owl:InverseFunctionalProperty>
  <owl:InverseFunctionalProperty rdf:about="#تسبب">
    <rdfs:domain rdf:resource="#المسببات"/>
    <owl:inverseOf rdf:resource="#بسبب"/>
    <rdfs:range rdf:resource="#المرض"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  </owl:InverseFunctionalProperty>
  <owl:InverseFunctionalProperty rdf:about="#يعالج_ب">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <owl:inverseOf rdf:resource="#يعالج"/>
    <rdfs:range rdf:resource="#العلاج"/>
    <rdfs:domain rdf:resource="#المرض"/>
  </owl:InverseFunctionalProperty>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.5, Build 663)  http://protege.stanford.edu -->
```