CHAPTER 1

INTRODUCTION

The number of public accessible structured resources over the web like Extensible Markup Language (XML) is growing rapidly. The flexibility of XML enables it to be used to express meaningful contents. Several popular sites like SIGMOD, DBLP publish structured resources on their sites for the purpose of information exchanged and retrieval. In addition, there are also collections of structured resources that have been produced for research evaluation from well-known real world data like Wikipedia, IEEE journal, IMDB etc. Moreover, many works (Graupmann et al., 2004) (Schenkel, Suchanek, & Kasneci, 2007) (Ley, 2009) have proven that these resources can be easily created from web contents like semi-structured hypertext documents or contents stored in database.

The primary intention of marking up and structuring these contents is for software agents to easily access them for various purposes, however, since these structured contents are openly and publicly accessible over the web, this expands the usage of structured resources to not only exchanging of information among pre-agreed machines, but enabling retrieval task such as information search. Moreover, meaningful markups used in these structured resources promote wider exploitation of such resources over the web, rather than limited to usages among agreed parties only.

Hence, these resources have become an important subset of the information published and shared on the web. And, it is obvious that much of the potentials of this subset of web remain untapped. It would be an advantage to current retrieval systems if they can utilize the structures or markups of documents for answering query needs. This leads to the active development of XML retrieval systems in recent years, which can be seen from the collaborative effort of Initiative for the Evaluation of XML retrieval (INEX) (Fuhr,

Gövert, Kazai, & Lalmas, 2002b).

Structural retrieval system exploits the structural information available in documents to implement a more focused retrieval strategy. The system returns document components or more precisely XML elements instead of complete documents in response to a user query (Pal & Mitra, 2007). The emergence of research in structured retrieval system will nevertheless benefit the field of information searching. By integrating structured retrieval (or also known as XML retrieval) methods in contemporary search systems, users will be able to directly lookup information from structured resources on the web. In such scenario, in order for users to benefit from structures or markups available in resources, query can be formulated in structured forms using methods like query languages (e.g. XML Query Language (XQuery) (Chamberlin, 2002), Narrowed Extended XPath I (NEXI) (Trotman & Sigurbjörnsson, 2004a)), forms (e.g. advance search (Barranco, Campaña, & Medina, 2005; Zwol, Baas, Oostendorp, & Wiering, 2006)) etc, whereby users can explicitly specify structures or markups in the query.

1.1 Motivation

1.1.1 On Exploiting Structural Information in Search

Structural information (i.e. markups and structures) are very useful if they are specified correctly as search constraints in a search process, whereby it can directly reflect the scope or context of a query information needs. The ability to utilize the structural information highly depends on factor like how these information can be included in the querying process (Kamps, Marx, Rijke, & Sigurbjörnsson, 2005). There are several methods of querying in structured retrieval that enable users to specify structural information. These methods can be classified into path-based, fragment-based, concept-based, form-based and keywords-based querying as follows.

1.1.1 (a) Path-based Querying

In path-based querying, user queries for desired information using expressions and paths. The most popular path-based languages for querying structured resources would be XML Path Language (XPath) (Boag et al., 2007) and XQuery (Chamberlin, 2002). XQuery is similar to Structured Query Language (SQL) for querying records in database system, whereby it allows user to specify keywords and structural constraints in a query. And, the query returns all matched answer to user without performing any ranking. Following the emergence of XML retrieval systems, needs arise in order to allow user to express precise information needs but in a simpler manner. Hence, query language like NEXI (Trotman & Sigurbjörnsson, 2004a; Trotman, 2009) is introduced to provide a more convenient querying. Unlike XQuery which is more suitable for expert user like XML application developers, NEXI uses simplified syntax. Nevertheless, these languages still require a great effort of syntax formulation and validation, which is less appropriate to be in real time search needs. The complexities of this querying method also hinder users from using the structural information efficiently.

1.1.1 (b) Fragment-based Querying

Compare to path-based querying, an effective and simpler querying method would be XML fragment query (Carmel, Maarek, Mandelbrod, Mass, & Soffer, 2003). This work avoids complex querying by allowing users to pose their query using xml fragment, e.g. <chapter><title>XML tutorials</title></chapter>. Since xml fragment is a direct adaptation of XML format, this avoids the needs to learn or remember another query language. And, different from language-based query that requires users to write a syntactically correct query path, this method gives users higher flexibility when composing the target, constraint and structure paths for a query. Responsibility of handling users' needs is passed to the system ranking mechanism.

1.1.1 (c) Concept-based Querying

An even simpler yet expressive way of querying that utilizes structural information is concept-based querying. As featured in the work by Graupmann et al. (2004) and Graupmann (2004), structural information of keywords can be expressed as concept-value condition in the form of concept=value, e.g. title="War and Peace", author="Tolstoy". Very similar to web query, this querying method can be easily exploited by general users for specifying more precise information needs. An example of an online search service which deploys similar querying format is the DBLP categorical refinement search. Concepts such as venue and author are used to refine scope of information look up. With similar intention, Cohen, Mamou, Kanza, and Sagiv (2003) also uses this querying format, i.e. label keyword in its semantic search engine for XML.

1.1.1 (d) Form-based Querying

No matter how simple a query is to be written, requiring a general user to manually specify the structural information or concept of keywords is not as straight forward as it seems. If the underlying structure of search collection is homogeneous one, i.e. based on simple, fixed and straightforward concepts like author, venue and year in DBLP Search (see Figure 1.1), then remembering and selecting the correct structural information is not a problem. However, for heterogeneous collection with rich annotated concepts like Wikipedia (Graupmann et al., 2004), it is impose such feature in the look up process. Hence, Bricks (Zwol et al., 2006) introduces a graphical approach, using an advanced form-based query builder, to help user in selecting structural information or concepts. Although selecting structural information for richly markups collection (e.g. Wikipedia) or across different collections could be possible by using the approach proposed in this work, there are issues like too many unique concepts, confusion on usage of same naming for different concepts etc. that need to be looked into.

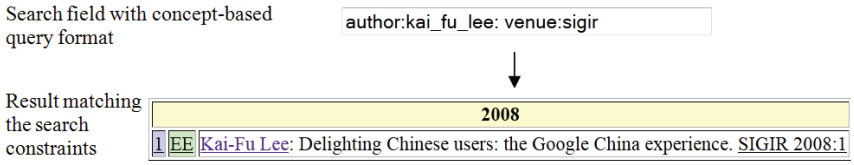

Figure 1.1: Search query with *concept and value* format in DBLP faceted search feature.

1.1.1 (e) Keywords-based Querying

The usage of keywords only queries (also known as content only or CO topic) in structured retrieval systems can be seen in INEX (Fuhr, Gövert, Kazai, & Lalmas, 2002a). Although structured resources were initially designed to be queried using structured query, in order to allow more users to harness information from openly available structured resources, keyword-based querying has become an important way of retrieving that is much more familiar and easy for users. If we looked at the user survey carried out by (Kazai & Trotman, 2007), comparing usage preference between keywords and advance search form on the Web, most still prefer the former. When only keywords are available, current works either ignore the usage of structural information in their retrieval process, or automatically add the structural information to the query.

It is obvious that without some kind of structural hints (i.e. markups or structures) in a query, it is hard to even determine the granularity of elements to retrieve, which is crucial in structured retrieval. Hence, there are recent works that automate this process (Petkova, Croft, & Diao, 2009; Kim, Xue, & Croft, 2009; Hsu, Lee, & Wu, 2004) to improve the effectiveness of keyword-based querying.

1.1.2 On Automated Construction of Structured Query

From the querying methods mentioned above, we can see that there are basically two ways of including structural information in a query, either manually specified by users or automatically included by systems in their retrieval processes.

Comparing both, a more straight forward way for exploiting structural information is to let users decide what they want manually, and directly include those information in the query by either formulating the information as syntax or select them through friendly interfaces. However, when we pose this as a search problem over the web whereby the environment is heterogeneous and information needs is defined in an adhoc manner, issues like complex syntax, naming variations and structures heterogeneity arise during query formulation process.

When users need to explicitly specify structural information, a primary obstacle is that they need to be familiar with the syntax of querying languages in order to be able to include the information in the query. Hence, we can see that many works trying to simplify methods of querying as discussed in previous section. For example, work by Zwol et al. (2006) presents that users have problem expressing the structural information needs if they need to deal with syntactical features of such languages. Similarly, another work by Carmel et al. (2003) also tries to assist users by simplifying the querying syntax.

Further, if we assume that syntax formulation issue can be addressed by using some visual aids or tools, users still need to be familiar with the underlying document structure in both explicit (e.g. naming, path, schema) and implicit manners (e.g. domain, application, context). For instance (refer Table 1.1), users need know the structural path to be able to define the correct target (e.g. inproceedings) or constraints (e.g. author). Same goes for the issue of naming. It is not practical to expect a user to remember constraint names like “booktitle” or “mtitle”. Graphical approaches may solve the issues related to utilization of explicit features of document structure but not the implicit one. For example, if there is a drop down list to let users refine the search for “SIGIR”, users must be aware that “booktitle” in DBLP collection refers conference proceedings, while “journal” refers to newsletter issue. This makes prerequisite knowledge necessary in order to utilize the structural information effectively.

Table 1.1: Some examples of NEXI query

Collection Type	Source	Example NEXI Query
Conference Article	DBLP	//inproceedings[about(.//author, Kai-Fu Lee) and about(.//booktitle, SIGIR)]
Forum Newsletter	DBLP	//article[about(.//author, Bruce Croft) and about(.//journal, SIGIR Forum)]
Conference Abstract	SIGIR Abstract	//proceedings[about(.//author, Croft) and about(.//mtitle, SIGIR)]

If we look at the results of INEX 2005 adhoc search track for the Wikipedia collection, queries with added structural constraints appear to perform similarly to those that do not specify one (Trotman & Lalmas, 2006). These results contrasted the theory of structured query, where structural constraints improve the precision of structured retrieval systems. The main reason mentioned is that users are bad at specifying structural hints. A later work (Trotman, Rocio Gomez Crisostomo, & Lalmas, 2009) then reinforces this claim through an analysis of the queries in INEX 2008 collection. The work shows that the usages of structures are merely for targeting the size of results only, similar to the observation made by Lehtonen, 2006.

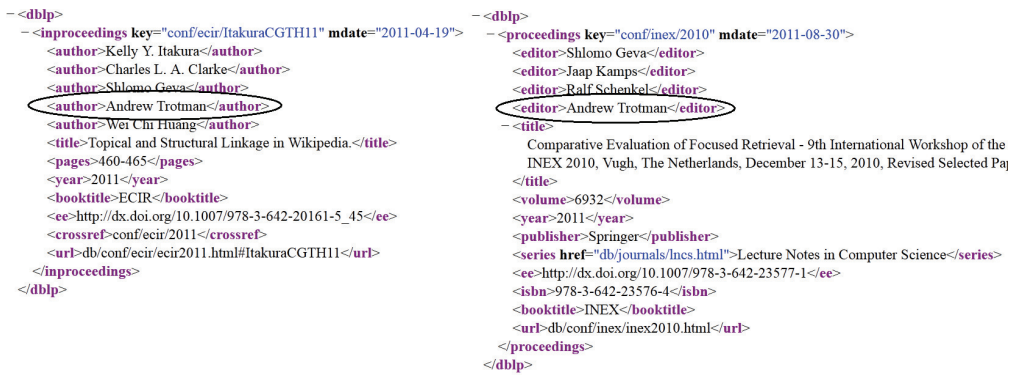

Figure 1.2: Example of semantic markups in XML documents from DBLP.

Here, we have noted that this problem is highly related to the type of structures used in resources. As mentioned in Zwol et al., 2006, three types of markups may be used in a structured document, i.e. semantical, logical and presentation markups. And, collections like IEEE, SIGMOD XML, INEX Wikipedia (up to 2008) fall into the logical

category. When resource structures are logical type, users cannot exploit much of the markups to further refine or reflect their information needs conceptually. For instance, when meaningful structural information like <author> or <editor> is used (see Figure 1.2), it will narrow the search scope to a specific concept, which will significantly improve answers relevancy. Whereas for logical markups such as <article>,<section>, <figure> etc., they are mostly used for defining the size of result, leading to little or no improvement in precision.

Therefore, most of the time users find it difficult to use the correct markups as structural constraints in their queries, not to mention structuring the queries manually. This has motivated solution that will automatically infer structural information (both markups and their structures), switching the burden of users to retrieval system.

1.2 State of the Art

Current works on query transformation from unstructured to structured form for structured collections on the web can be seen from those from information retrieval field or databases field. Although both communities may refer to a similar structured representation, i.e. XML, the former works on XML documents (Petkova et al., 2009; Tannier, 2005) while the latter works on XML database repositories (Calado, Silva, Vieira, Laender, & Ribeiro-Neto, 2002; Barranco et al., 2005). Since contents from XML documents are publicly available, while contents from XML database are remained for internal usage, as such, it is more likely for the current web search solution to acquire contents from the former collections rather than the latter. Therefore, in this section, we present the state of the art concerning approaches used by the information retrieval rather than databases. The state of the art is discussed from two aspects of the query transformation process, i.e. the inferring of structural information and the construction of structured queries.

1.2.1 Inferring Structural Information

The key source of structural information that can be used for query structuring is in fact the markups and structures of the collection itself. Unless the structures are logical one, where corpus knowledge would not be relevant in inferring query's intention, otherwise collections schema or annotations are useful sources for query context analysis. The simplest way to obtain the relationship between a term and a particular structure is by capturing all the relationships between term and its markup/structure/structure path in the collection, and then use them for marking up query during retrieval time. Probabilistic methods are used to estimate the association between a term and its structure (Petkova et al., 2009; Kim et al., 2009; Hsu et al., 2004; Bao, Lu, Ling, & Chen, 2010). As it is one-to-many relationship, this estimation applies well when a collection has simple or homogeneous structure, with little or no ambiguities in its structural concept for a term. For example, an estimation that "andrew" is an "actor", followed by "title" at a lower probability, is probably still satisfactory under a domain with few structural types like movie domain.

However, as we extend our problem to scenario such as searching a more general collection like web site, which consists of many different page types, or even a collection where there are many possible schematic views; further analysis on the markups and structures usage are required to disambiguate differences of structural concepts a term may have. For example, for collection with different schematic views like bibliography domain, "andrew" can be an "author" of a "journal article", or "proceedings article", or "book chapter" etc. Or, when we look at "andrew" from different sites, he can be a "chairman", "senior pc", "lecturer", etc. Hence, current works have limitations in handling this kind of ambiguous situations.

1.2.2 Constructing Structured Queries

There are two main approaches used by the works on the construction of structured queries, i.e. templates or operations. In the first approach, Woodley & Geva, 2004 and Woodley & Geva, 2006 create a set of grammar templates based on structured queries samples collected from INEX forum. Each grammar template corresponds to an individual information request. Similarly, Tannier, 2005 uses XSL Transformation to generate NEXI structured query from its generic query representation known as DRS.

As template approach may suffer from its coverage of structured query formats, there may be difficulties when new templates need to be added. Hence, in the second approach, a set of transformation operators are used to construct the contents of a structured query, which is mainly used to identify target term and content term in the query (Petkova et al., 2009; J. Li, Liu, Zhou, & Ning, 2009). For example, in Petkova et al., 2009, the operations are used to formulate target and constraint terms identified from keywords query into NEXI query language. However, rules needs to be crafted for every possible operation of the structured query language. As we are trying to look into the possibility of a generic query transformation process, ability to accommodate new structured query has become our concern.

With respect to the motivation and current state of the art of query transformation, the next section presents the problems that make this research challenging.

1.3 Query Transformation as a Structured Retrieval Problem

A practical application of query transformation from unstructured query to structured from is to enable integration of structured retrieval features into web search solution. Consider situations where these retrieval systems are used over the web. Here, different retrieval systems mean that we are dealing with different collections (see Figure 1.3). And, this signifies that different structured collections need to be addressed accord-

ingly based on their own concepts and structures in its query construction. This is due to heterogeneities in terms of information structures, document nature and lexical ambiguity among these collections.

Next, dealing with multiple retrieval systems also means that we are dealing with different retrieval methods, so as the structured queries employed. As these queries could fall into categories of either concept-based, fragment-based, or path-based, therefore they have different levels of complexity. For example, system for a text-centric structured resource collection like Wikipedia may deploy a less strict matching option by using the concept-based query in its retrieval method. Whereas a record-centric one like DBLP may deploy a straight forward matching by using a path-based query.

Therefore, transformation between an unstructured query and a structured one does not only involves a set of transformation rules, but many sets of rules if we want to enable the transformation to more possible structured queries forms. And, it is tedious to create different rules for different pairs of queries. Moreover, there are certainly needs of accommodating new querying interfaces, or variants of the existing one. Thus, the approach of redefining rules each time a new interface is introduced is less flexible in applicability and do not generalize well across new structured forms. E.g. a separate transformation process, *SQT1*, *SQT2* and *SQT3* (see Figure 1.3) are required for each interface. Instead, we attempt to reduce many pairs of rules that link to different structures into a more generic form by generalizing the structuring process.

1.4 Goals of This Thesis

The current query transformation solution for structured retrieval is designed specifically for a single type of query, scoring model as well as collection. However, this solution lacks flexibility to cater for evolving structured retrieval environment, especially multiple query types, collection complexities and query interpretation models. The objective of

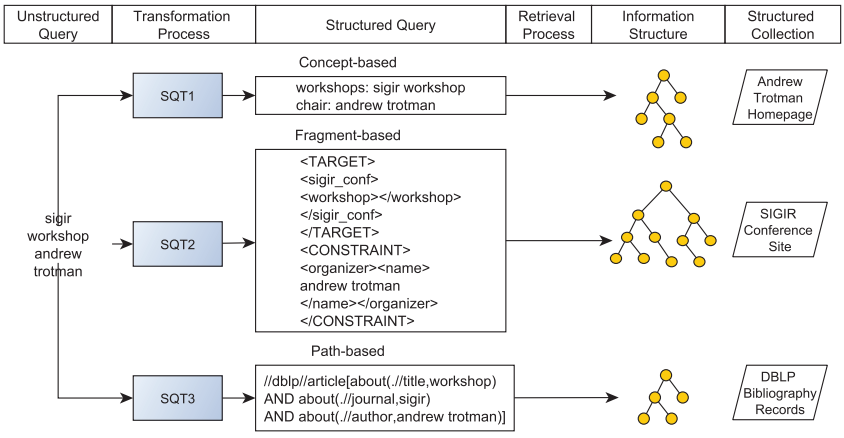

Figure 1.3: A search scenario using structured retrieval systems on the web.

this thesis is to design a flexible framework that can handle the variations or evolutions of these components, with friendlier user interface and better retrieval performance. With respect to this objective, we present a series of goals as follows.

- Improve template-based or rule based method for incorporating new structured queries instead of using fix templates or transformation rules.
- Optimize information utilization from collection side for better query interpretation by
  - extending the probabilistic method to include context factor for query interpretation of complex collection with heterogeneous structures.
  - allowing the probabilistic method to incorporate various type of basic term scoring methods to suit its collection.
- Optimize information utilization from query side for better interpretation using structural keywords in query, in which these keywords are used in indirect manner.

Research Questions In order to justify the feasibility of our research goals, there are several questions that need to be answered.

Q1 Can the proposed flexible query transformation framework scales to differnet structured collections and structured queries types?

Q2 Can the proposed flexible query transformation framework generates a structured query that gives a better retrieval performance compare to the original query?

Q3 Can the extension of probabilistic method with context factor helps in improving the accuracy of query transformation for collection with higher structural complexities?

Q4 Can the proposed extension of probabilistic method be used with existing term weighting models?

Q5 Can query with certain features of information needs such as *"more specific"*, *"longer in size"* and *"inclusion of structural keywords"* give better accuracy for its translated query?

From the stated research goals and questions, we proceed with the following methodologies:

1. Review and identify literatures and their limitations related to various aspects of query transformation in structured retrieval environment (Chapter 2).

2. Develop a formal framework for flexible query transformation (Chapter 3).

3. Instantiate the framework on real information needs and structured resources. A set of algorithms for query interpretation, representation and ranking are designed based on the theory of formal framework (Chapter 4).

4. Evaluate the performance of flexible query transformation framework based on the raised research questions. Evaluation is carried out at from multiple aspects such as query interpretation algorithms, query representation and retrieval outcome (Chapter 5).

1.5 Proposed Framework

The principle underlying our proposed query transformation framework is to have a generic process that could be easily adapted to changes in structured retrieval environment. As such, we propose an intermediate query representation, that can represent the interpreted query in a generic query structure form. This query structure is independent of a specific query language. To convert this structure to a query language, a structure to syntax mapping is defined. The mappings of query structure to syntax are defined using an example-based knowledge base method. This method does not refine the creation of mappings for a single query language, but it is flexible to be applied to more than one query types.

In addition, the proposed solution also handles current limitations of query term interpretation that may occur in complex collection. In complex collection, there exists deeper structure path for a term, hence, it is insufficient if a term is only associated to its immediate parent for term interpretation. Our solution handles this by considering additional good ancestor structures besides its parent. As complex collection also contain heterogeneous elements instead of fewer types as in homogeneous collection, the ambiguity of term is higher. To handle this, our solution introduces context within a collection, where a term will be associated to these contexts during interpretation. This makes the selection of structure/concept context-specific. To capture the probability of a term with its structural under a particular context, a context-based probabilistic model that combines statistical information of contents occurrences and hierarchical information of schema/structures is used.

We present an illustration of the proposed query transformation framework in Figure 1.4. There are three major parts in the proposed framework.

- *the query interpretation process* that consists of two sub modules, i.e. query inter-

pretation and query construction,

- *the query representation model* that represents interpreted queries in generic form and a set of mappings for conversion to structured queries,

- *the knowledge modules* that consists of a context-based term weighting model for query interpretation, a query template knowledge base for storing created templates and mappings for structured queries conversion,

In a query transformation process, an unstructured query will go through the query interpretation module for analysis of information needs. The interpretation and optimization of the query will be carried out using the collection knowledge generated by context-based probabilistic model. Then the interpreted contents will be constructed and represented in a generic query form, known as intermediate query representation. This intermediate form is represented using two separate query structures, that capture the semantic and syntax of queries separately. The interpreted contents in this structure are transformed to query syntax via pre defined mappings. The generation of these mappings are obtained from a set of structured query examples.

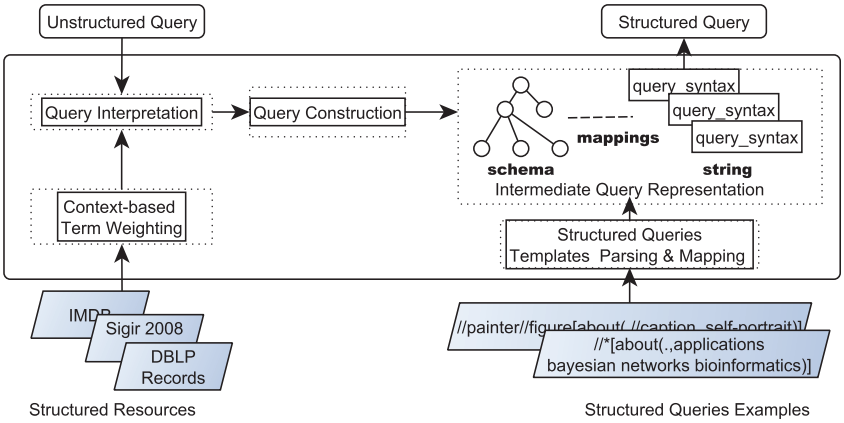

Figure 1.4: The Proposed Query Transformation Framework

1.6 Thesis Contributions

The contributions of the thesis are as followed.

The main contribution of this thesis is its flexible framework that makes query transformation process generic and adaptable for different settings of structured retrieval tasks, such as the collections type, structured querying interface, and query interpretation scoring method. Compare to existing query transformation solutions which have been developed for single transformation, our framework meant to be easily extensible and customized to any domains and retrieval systems. Although existing solutions may have stated that the model they employed are extensible, e.g. from simple hierarchy to complex hierarchy for its query interpretation scoring model, however, it has only been stated briefly without a detailed discussion. In contrast, our framework defines in detailed these aspects and proves them in both theoretical and practical manners. This differentiates the framework from current solutions as the framework targets to be a generic solution to query transformation rather than a one time solution.

Within this framework, we have made three sub contributions.

1. A context-based term weighting approach for query interpretation. This approach focuses on capturing a more precise concept for query term interpretation based on its usage under different contexts in collections. This approach overcomes the limitation of current concept weighting approaches as they still weigh concepts based on the entire collection view, where a term may only have one best concept per collection. This works fine for homogeneous collection but not heterogeneous collection. Whereas, our approach addresses this problem by introducing one best concept per context in collection. In this approach also, intermediate concepts factor is introduced to extend the current immediate concept binding for term. This is to address the issue of non-meaningful immediate concept node.

2. An interchangeable term scoring model for query interpretation. This flexibility allows incorporation of external term scoring model in its concept weighting based on

user preferences. Instead of using only one scoring model as in current works, this flexibility allows usage of simple to complex scoring models based on the nature of collections.

3. A novel intermediate query representation structure is used to represent interpreted query. Using this uniform structure, only single query construction operations set is required for generating the interpreted query. This structure is used to overcome the needs of individualized operations when constructing structured queries. The framework will only require the same operations set rather than individualized one for each structured query type.

4. A structure-syntax template is used to reconstruct a structured query string. For this, we propose a simple annotation and parsing method to generate the template. This method enables the incorporate new or modified structured query syntax with predefined query examples.

In addition, there are additional contributions.

1. We have formalized the query transformation framework. Such formalization is necessary to ensure its applicability and reusability.

2. The practicality of the framework is shown via the instantiation of the framework using both homogeneous and heterogeneous collections, various complexities of information needs and term scoring models.

3. The evaluation of the framework from algorithm aspect and application aspect. The result of the evaluation shows that context-based concept weighting approach is able to suggest better constraint concept and target concept for query interpretation. For application in retrieval task, the structured query generated by the framework out performs its original query in unstructured form. Overall, this thesis changes the

way of designing query transformation solutions from rigid manners to a customizable way under a flexible framework. Although the framework may not always give the best combinations of its components, it is always adjustable without affecting other parts. This characteristic is very desirable when as we are dealing with an evolving environment. Up to date, we are not aware of any work that considers this characteristic under a flexible framework.

1.7 Thesis Outline

The rest of the thesis is organized as follows. In Chapter 2 we provide background on structured resources and querying methods. We also include a detailed analysis of related works with respect to the problems of this thesis. Chapter 3 describes our query transformation framework that uses intermediate query representation to overcome limitations of the inflexibility of conventional transformation approach. This chapter also describes the knowledge modules, i.e. a probabilistic model for context-based query interpretation and a structure query template knowledge base for the mapping of interpreted query to query language. Chapter 4 presents the algorithms used in query transformation. It shows how a query is interpreted and constructed into structured query form. In Chapter 5 we present the evaluation of the framework based on its algorithm, application and representation. Chapter 6 concludes the thesis with discussions and outlines directions for future research.