Chapter 1

Introduction

Ontologies are key components in the development of the Semantic Web, as they provide a shared understanding of a domain. In the field of digital libraries, ontologies support tasks such as the description of resources, integration of information, interoperability, search and browsing. However, acquiring the knowledge to construct ontologies is a costly task that requires much time and many resources. This task can be supported by ontology learning methods, which are often based on machine learning techniques, natural language processing or clustering algorithms.

This work presents the OntOAIr method (Ontologies from Open Archives Initiative Repositories to Support Information Retrieval), a semi-automatic method for the construction of lightweight ontologies called *ontologies of records*. The adjective *semi-automatic* refers to the fact that the method requires minimal human intervention (only to determine input parameters), while the term *lightweight* indicates that the construction of ontologies does not involve domain experts. The OntOAIr method uses simplified representations of documents, an adaptation of the Frequent Itemset-based Hierarchical Clustering algorithm (FIHC) [16], and ontological engineering techniques.

The OntOAIr method allows human and software agents to organize and retrieve information from document collections. We use collections provided by the Open Archives Initiative (OAI) as a testbed. The next sections describe the context and the mechanisms to access these collections.

1.1 OAI background

Digital libraries enable on-line access to valuable document collections of many scientific and technical disciplines. A lot of research has focused on developing standard ways to address data exchange issues. The Open Archives Initiative (OAI) develops and promotes interoperability standards that aim to facilitate the efficient dissemination of content.

OAI is an organization formed by a broad range of institutions, researchers, librarians, publishers and archivists whose aim is to create simple standards of interoperability in digital libraries [30]. With respect to our work, especially the following characteristics of OAI are noteworthy:

- *Autonomy.* Each data provider has its own policies and administration
- *Decentralization.* Data providers do not have to report either collection updates or changes in their records to central coordinators
- *Dynamism and amount of data.* OAI has hundreds of members. New members are incorporated frequently

- *Independent origin.* Collections offered by data providers are built independently as a response to the needs of particular communities

There are two types of participants in OAI: data providers that expose metadata of their resources in semi-structured documents termed records, and service providers that use records to offer value-added services [30]. At the time of this writing, there are 830 data providers that belong to universities, research institutes or libraries [1]

Data providers have become valuable information sources for applications and users of digital libraries. Their records describe all types of resources such as articles, books, magazines, maps, theses, and videos.

Unqualified Dublin Core (DC) is a metadata format recommended by the OAI. DC defines elements to describe the content, intellectual property and other features of resources. It is widely used in federated digital libraries, which offer collections of decentralized data that are accessed via remote services. The use of DC in data providers guarantees that service providers which do not handle any other metadata format will at least be able to get the basic information about resources from their DC renditions.

The elements `<dc:title>`, `<dc:subject>` or `<dc:description>` store content information. Table 1.1 shows a typical record taken from the Tales Collection[2] of Digital Theses. XML[3] is used to encode records.

[1] Registered data providers. Open Archives Initiative. October 14th 2007. $< http://www.openarchives.org/Register/BrowseSites >$.

[2] Tales Data Provider. Universidad de las Américas Puebla. June 16th 2008. $< http://ict.udlap.mx:9090/Tales/Oai_tesis >$

[3] Extensible Markup Language

```
<record>
 <header>
  <identifier>oai:thesisUDLAP:98</identifier>
  <datestamp>2001-08-01</datestamp>
 </header>
 <metadata>
  <oai_dc:dc
     xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc"
     xmlns:dc="http ://purl.org/dc/elements/1.1/"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
     http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
     <dc:title>
       Collaborative Space to Support Research Activities
     </dc:title>
     <dc:description>
       This thesis describes a collaborative space where user
       information needs are addressed by librarians or
       software agents
     </dc:description>
  </oai_dc:dc>
 </metadata>

</record>
```

Table 1.1: Sample record from Tales Collection

Records are associated with a unique identifier. They are encapsulated within a special structure formed by two mandatory parts: metadata and header as illustrated in Table 1.1. The former includes elements of a specific metadata format. The latter uses the elements `<identifier>`, `<datestamp>` or `<setSpec>` for harvesting pusposes.

The Open Archives Initiative (OAI) proposes a low level mechanism called the *Protocol for Metadata Harvesting* (OAI-PMH) to support application-independent in-

Request verb	Retrieved objects
`GetRecord`	A unique metadata record
`Identify`	Information about the data provider
`ListIdentifiers`	Headers of records
`ListMetadataFormats`	Available metadata formats
`ListRecords`	Harvested records
`ListSets`	Set structure

Table 1.2: *Request verbs* of OAI-PMH protocol

teroperability.

1.1.1 OAI-PMH protocol

The OAI-PMH protocol provides external online access to records. There are three versions of this protocol: version 1.0, 1.1 and 2.0 [10]. Our work considers the current version, 2.0. In this version, the GET/POST methods of the Hypertext Transfer Protocol (HTTP) and XML are used to encode requests and responses, respectively. DC is the default metadata for OAI-PMH Version 2.0.

The OAI-PMH protocol defines *request verbs* which are utilized by service providers to harvest records. Table 1.2 shows the objects retrieved by each request verb. Data providers respond to these requests with XML documents.

In order to access records in a collection, first it is necessary to process the values returned by the `Identify` verb, as it contains data such as the identifier of the data provider, the date of creation or the last update of the records, the list of available metadata formats and the definition of sets. Afterwards, the remaining verbs can be used for harvesting.

As a way of illustration, the next request uses the `GetRecord` verb to retrieve the record of Table 1.1:

```
http://ict.udlap.mx:9090/Tales/Oai_tesis?verb=GetRecord&
   identifier=oai:thesisUDLAP:98&metadataPrefix=oai_dc
```

The components of this request are interpreted as follows:

- There is a data provider that handles OAI requests available at:
  `http://ict.udlap.mx:9090/Tales/Oai_tesis`
- The request verb tells the data provider what to do:
  `verb=GetRecord`
- The identifier indicates which record is being requested:
  `identifier=oai:thesisUDLAP:98`
- There is a specification of the metadata format:
  `metadataPrefix=oai_dc`

There are two elements for selective harvesting: *datestamps* and *sets*. Datestamps contain the creation date or the last modification date of a record, whereas sets are optional constructions to cluster records. Sets allow data providers to expose their internal structure to service providers. It is not mandatory for any data provider to support set constructs. Indeed, the protocol does not establish any kind of clustering by default. Thus, the use of sets requires explicit agreement between data providers and service providers. Therefore, selective harvesting often relies on datestamps.

As noted in Table 1.2, the OAI-PMH protocol does not offer content retrieval mechanisms, that is, *request verbs* do not implement similarity measures between records and requests in order to harvest only relevant records. As a consequence, a harvesting task often implies the retrieval of a specific record or the retrieval of all the records from a data provider.

Therefore, data providers are free to choose a search service to facilitate discovery and retrieval of relevant resources. Thus, when multiple data providers are involved, users would require the propagation of queries to each data provider in order to produce an overall results list.

1.1.2 OAI retrieval tools

The typical ways to find relevant records on OAI-compliant data providers are search engines and service providers that implement search mechanisms. Both of them produce lists of potentially relevant records.

Record lists thus produced exhibit two general drawbacks. The first one is that these lists do not satisfy the information needs of all users. Instead of lists of records, some users would be more interested in obtaining a view that enables them to explore data providers through the construction of groups of similar records.

The second drawback is that most search engines and service providers search on a single data provider. Thus, when multiple data providers are involved, users must ini-

tiate several requests. Therefore, records are harvested, selected and organized before determining their relevance. The complexity of this task increases with the volume of data providers and the large number of metadata formats.

The incorporation of document clustering techniques into the information retrieval models to OAI repositories should help overcome the first disadvantage. Clustering would generate groups of similar records instead of lists of independent records. A distributed search engine might be a solution for the second drawback. It would require the propagation of queries to several search engines in order to produce an overall results list.

The quality of an overall list would depend on factors such as the search quality provided by the participating search engines and the integration mechanism of partial results. If the quality of this list is high, then document clustering techniques could transform the list into groups of similar records. The search of relevant records would be a time-consuming task due to the volume of data providers. Consequently, tools are needed to support users and applications in this task.

In addition to the use of search engines, some members of the OAI community have developed tools to retrieve information from data providers. With respect to our work, especially the following keyword-based tools are noteworthy:

- *Arc*: This is a federated search service based on a harvester and a DataBase Management System (DBMS). The search mechanism of the DBMS is used to retrieve records. Arc is implemented in the Java language with Java Servlet Technology. It is compliant with Version 1.x and Version 2.0 of OAI-PMH [3].

- *myOAi*: This is a harvester that works as a search interface for a selected list of metadata databases. myOAI is implemented in the Perl language. It is compliant with Version 1.1 and Version 2.0 of OAI-PMH [4] .

- *Net::OAI::Harvester*: This is a harvester that allow users to select a data provider for searching. Net::OAI::Harvester provides an object-oriented interface of retrieved data. It uses an XML:SAXPerl parser to support the stream of OAI-PMH *request verbs*. It is compliant with Version 2.0 of OAI-PMH [5] [52].

- *OAI-PMH Pack*: This is a service to browse and search harvested records from a data provider. It offers high-level access to data providers through indexing abilities. OAI-PMH Pack is implemented in the Python language. It is compliant with Version 2.0 of OAI-PMH [53].

These tools produce independent lists of potentially relevant records, but only with respect to a set of given keywords. As mentioned earlier, our work is particularly focused on retrieving groups of similar records based on their content. Our interest in clustering records is motivated by the fact that in any data provider there is a lot of content information hidden in the records.

In our work we posit that ontologies can help overcome the drawbacks of lists of independent records and the retrieval from a single data provider. The following section introduces some definitions and types of ontologies that make up the framework for ontologies of records.

[4] *http* : *//myoai.org/oai*
[5] Available at: *http* : *//www.myoai.org/*

1.2 An overview of ontologies

In the context of information systems, the literature contains many definitions of ontologies. One of the most quoted definitions establishes that an ontology is an explicit specification of a conceptualization[20]. An extended version of this definition suggests that the conceptualization must be shared[5]. [15] describe ontologies as formal explicit descriptions of concepts in a domain, properties of each concept that describe various attributes and restrictions on properties.

Ontologies describe objects, types of objects (classes), attributes and relationships between objects. It is a form of shared knowledge representation or a data model about a specific domain [18].

Knowledge management and interoperability are salient uses of ontologies. Common applications are found in medicine, linguistics, bioinformatics, engineering and digital libraries. In the latter field, they have been widely used to index or access document collections.

The literature distinguishes lightweight from heavyweight ontologies according to the degree of formality involved in their encoding [31], [49]. The scope of our work is limited to lightweight ontologies. They range from an enumeration of terms to a graph or taxonomy of concepts with well-defined relationships among them, which provide a representation of an information space.

In lightweight ontologies, there is not a strong distinction between terms and concepts. However, they make controlled vocabularies available for the classification of

content and they are use-dependent. Web search engines use lightweight ontologies [18], [55].

Ontologies of records are a kind of lightweight ontologies. They are aimed to create a data model able to (1) provide a shared terminology for accessing data providers that human and software agents can understand and use, (2) define the meaning of each term in an unambiguous manner, (3) implement the semantics of the data model in a machine-accessible way, (4) indexing data providers to support information retrieval and (5) support a semantic exploration mechanism.

1.3 Contributions

The main contribution of our work to the field of digital libraries in the Semantic Web is the OntOAIr method for constructing lightweight ontologies. The ontologies constructed by this method can be used to support exploration and information retrieval from digital collections. Other contributions include the following:

- An adaptation of the FIHC algorithm
- The representation of hierarchies of documents in a machine-accessible language
- An algorithm to establish similarity between queries and groups of records
- An agent-based architecture to support a keyword-based retrieval model and an ontology-based exploration model
- The definition of schemas and namespaces to construct valid ontologies

- The formalization of ontologies in diverse markup languages
- A prototypical system that implements the OntOAIr method and demonstrates its feasibility and applicability

Although the conversion of the information currently available to systems into formal ontological knowledge at an affordable cost is an unsolved problem in general, the construction of ontologies of records is a step towards the formal encoding of the content of OAI-compliant data providers.

1.4 Outline of the document

The remainder of the document is organized as follows. Chapter 2 describes the related work. Chapter 3 presents the OntOAIr method for constructing lightweight ontologies. Then Chapter 4 proposes a keyword-based information retrieval model and an ontology-based exploration model as applications of the OntOAIr method. Chapter 5 describes a prototypical system which implements the proposed models. Next, Chapter 6 describes the evaluation of OntOAIr method. Finally, Chapter 7 includes conclusions and suggests future directions of our work.