Structured Knowledge Source Integration and its applications to information fusion

James Masters
Cycorp, Inc.
3721 Executive Center Dr.
Austin, TX

Abstract

Structured Knowledge Source Integration, or SKSI, is an ongoing research and development project at Cycorp intended to enable the Cyc knowledge base to integrate (access, query, assimilate, and merge) a variety of external structured knowledge sources, such as databases, spreadsheets, XML or DAML tagged text, GIS datasets, and queryable web pages. With SKSI, the Cyc knowledge base will be able to draw upon information obtained from multiple knowledge sources when answering complex queries, to assimilate (transform and store) the contents of the knowledge sources directly into the Cyc knowledge base, and to mediate between several semantically similar knowledge sources. These capabilities will extend the flexibility and power of the Cyc product to serve as the universal ontology and knowledge repository in any application requiring knowledge based reasoning. This article discusses some of the main technical issues of knowledge source integration, reviews some of the literature on the subject, describes some elements of the SKSI approach, illustrates two example Cyc queries that use two structured knowledge sources already mapped into Cyc, and proposes a Schema Modeling Toolkit of applications we are designing to leverage the core SKSI development.

Key words: Cyc, data fusion, data management, data mining, information fusion, information management, information mining, knowledge bases, knowledge fusion, knowledge management, knowledge mining, multi–database Federation.

1 Introduction

Many analysts and decision makers in the government, military, and business sectors work amidst a sea of available information presented to them by several disparate sources that do not interoperate, or do so only through data visualization techniques or some other shallow level. Interpreting the meaning of the relevant information from different sources is left to the analyst, who must spend valuable time cognitively sorting through the available sources to get at the truly relevant information. Clearly, some intelligent middlewear is needed to help the analyst perform this time consuming task. This need presents both a challenge and an opportunity to create an information management system that can incrementally add new sources to its knowledge base and support much more complex queries than any one of the knowledge sources alone.

To answer this need, Cycorp is developing the technology to incrementally integrate a general class of knowledge sources (not just databases) into the Cyc knowledge base. We use the Cyc ontology and the CycL language to declare the structural and semantic mappings necessary to translate knowledge represented in an external knowledge source into the Cyc KB, as well as to represent the knowledge itself. The mappings for a single knowledge source are declared using a toolkit designed to aid the user through the process of identifying concepts represented in the knowledge source with their correspondences in the Cyc KB. Once the mappings have been defined in the Cyc knowledge base (KB), the Cyc inference engine can treat the contents of the knowledge source as if the knowledge resided in the Cyc knowledge base, or its contents could be batch translated into CycL and stored either in the Cyc KB or in another external data structure.

The rest of this article is organized as follows: Section 2 reviews several concepts of central importance to knowledge source integration; Section 3 reviews some of the related literature concerning the integration of multiple databases and notes some of the main technical issues identified in them; Section 4 introduces the Cyc knowledge base and the

CycL language, and describes how our approach addresses the technical issues raised in the literature; Section 5 presents an example query in CycL which is answered using two different structured knowledge sources; Section 6 introduces ideas we have for a *Schema Modeling Toolkit*, a set of potential utilities and applications which will unleash the power of SKSI in existing and future Cyc projects and products.

2 Some technical details

We assume the reader is familiar with standard database terminology and data models, including the entity–relationship model, the relational database model, and object–oriented database models. See [9] for definitions and details. We use the language of the entity–relationship model in this article, since conceptual schemata in this model are easily tranformed to conceptual schemata in the other models, and since the schemata of more general structured knowledge sources can be modelled using entity–relationship diagrams.

A *Structured Knowledge Source*, or simply a knowlege source, is an information repository in which knowledge is represented and stored in a systematic way that is easily characterized by some type of formal structure, or conceptual schema. The schema should provide the user with enough detail of the logical structure and physical design of the knowledge source to allow access and querying by some type of management system. Examples of structured knowledge sources include databases, spreadsheets, xml or daml tagged documents, GIS datasets, and queriable web pages, among others.

The *conceptual schema* of a structured knowledge source contains the logical and physical specification of all the *entity sets*, *attributes*, *attribute domains*, and *relationships* which may appear in a physical copy of the knowledge source. These are collectively the elements of the schema, or its *schematic elements*.

A central issue of interoperability between multiple knowledge sources is the need to model the *semantic proximity* of schematic elements represented in different knowledge sources, as well as the *structural heterogenity* of their physical representation. Semantic similarity between schematic elements rising from different knowledge sources is the degree to which they are alike with respect to the "real world" objects to which they refer. Structural heterogeneity refers to the schematic differences in the physical representation of the schematic elements. At the structural level, transformation rules are often used to reconcile the differences between the physical structure of different knowledge sources. At the semantic level, similarity between schematic elements is usually captured by comparing the "real–world meanings" of the schematic elements from different knowledge sources in a common context of comparison.

Another central issue of interoperability between multiple knowledge sources is the need to perform *natural joins*[1] between entity sets taken from different schemata which share one or more common attributes. In essence, a natural join produces a new entity set with the attributes of both joined sets. The elements of the new entity set are the concatenations of all pairs of elements from the two original entity sets which have the same values for all their common, or *joinable*, attributes. In general, joining entity sets across different knowledge sources requires accounting for their structural heterogeneity and semantic similarity.

A *structural join*, or *syntactic join* between two entity sets is possible if they share one or more attributes which are structurally the "same", that is they have identical physical representations in both of their conceptual schemata, or if they both share the same conceptual schema. This is equivalent to the natural join operation for a single knowledge source.

Alternatively, if the entity sets share attributes which are semantically equivalent but are not identical with respect to their physical representation, then a *semantic join* is necessary. Semantic joins require mapping the semantic meaning of the sources' schematic elements into a common context in order to compare values between semantically equivalent attributes.

Most existing models for knowledge source integration offered in the literature limit the type of sources they consider integrating to databases. A *multi–database federation* is an information management system which supports a common set of operations over multiple databases. There are two commonly recognized types of federated systems. In a *tightly–coupled federation*, data is accessed using a global schema specification to which all the federated databases must adhere, while in a *loosely–coupled federation*, each database maintains its own local schema specification; the management system must be able to translate between specifications in order to share data across the federation. Structural heterogeneity does not occur in tightly coupled federations, since by design the databases in the federation all share a common representation schema. Management systems for loosely–coupled federations, however, must

[1] see [9], volume I, chapter 2, section 4

address both problems.

The management system of a federation of multi–databases is often called a *mediator*[8]. Mediators manage the interoperability between members of the federation and provide services like evaluating queries which extract information from several members of the federation. In general, each database may be designed, maintained, and updated independently of (and unaware of) each other. Also, the databases are usually loosely–coupled; they don't share a common conceptual schema that enforces common data structures and semantic meanings of the schematic elements (entity sets, relationships, attributes, etc.) among all the cooperating databases. Furthermore, the databases are often distributed throughout the global information system (Internet), were never designed to interoperate, and may dynamically change contents. Mediators for such federations must be able to deal with these problems as well.

3 Related literature

Several information systems researchers have written on various architectures for mediators of multi–databases. They have identified many of the technical difficulties which arise when designing mediators capable handling all of these management tasks for a loosely–coupled federation.

The authors of [4] and [5] develop a taxonomy of the degrees and types of semantic similarity which two schematic elements may share, as well as a broad class of the representational differences which may occur between the schematic elements from different members of the federation. They present a formal model of the semantic/structural dichotomy and try to reconcile it by introducing *contexts of comparison* and *schema correspondences*. In short, schema correspondences map semantically similar schematic elements from structurally different knowledge sources into a common context of comparison.

The authors of [8] present a typical example involving two semantically equivalent, yet structurally heterogeneous GIS (Geographic Information Systems) databases. Their implementation method develops some of the ideas in [4] and [5]. They consider a method to transform records represented in the schema one database into records represented in the schema of the other database. They rectify the schematic differences between the two databases using their rule–based mediatior, MECOTA, and identify the semantic equivalence between them using OIL, or ontology inference layer.[1] Furthermore, they provide details of how their method deals with one class of schematic heterogeneity that is described in [5].

In [3], the authors argue that constructing federation mediators that rectify schematic differences and semantic similarities is not sufficient for databases dispersed throughout the global information system and which are subject to periodic updates or extensions. They point out that in these cases, it is essential that the federation mediator be complemented by static and dynamic information about the location and accessibility of physical copies of the databases. For example,

- What is the access path (e.g. url address) to the database?
- What are the permissions needed to access the database?
- How often is the database updated?

These considerations motivate the use of a common ontology which evolves incrementally as the federated databases are updated, and as new databases join the federation. The authors present a data model which addresses these requirements.

4 Cyc, CycL, and SKSI

The issues raised in the literature motivate design of a mediator which uses a large and easily extensible universal ontology and schema modeling language rich enough to

- Resolve the conflicts between the physical structures of the knowledge sources;
- Translate and compare the semantic meaning of the logical structures of multiple knowledge sources so that semantic joins are possible in the context of comparison;
- State the access dependencies (location, permissions, update schedule) of physical copies of the knowledge sources, so that the system can dynamically account for these factors when querying the knowledge sources.

We believe that the Cyc technology is uniquely qualified to be the basis of a mediator which address all of these needs. *Semantic Knowledge Source Integration* is the name of a core development project at Cycorp intended to provide these and other knowledge fusion capabilities.

Cyc is a very large, multi-contextual knowledge base and inference engine.[6] Cyc contains a vast

body of knowledge about the world. The inference engine reasons over this knowledge, infers new assertions and either adds them to the KB, returns them as answers to a query, or both. CycL, the declarative representation language of Cyc, is based on n^{th} order predicate calculus. CycL supports the occurance of variables in relation arguments, the creation of variable-arity functions, relations, and meta–relations, default and monotonic reasoning, truth maintenance, contextualization of knowledge, and reasoning by argumentation (finding and weighing pro/con arguments).

We are progressively developing a Schema Mapping Language (SML) within CycL which we use to declare the physical and logical schema of a knowledge source, as well as its access paths, privledges, and update frequency in the Cyc KB. The inference engine uses these declarations to construct heuristic level modules which are optimized to access the knowledge source according to these schema mappings. Once this mapping is established, it can be queried through Cyc as if its content were part of the knowledge base.

The knowledge base is used to store both the mapping declarations establishing connectivity, as well as the semantic content of the knowledge itself. CycL models both the schematic structure of the knowlege source and the semantic meaning of its content. The schema correspondences and the contexts of comparison are kept separate by placing them in separate microtheories.[2] For each integrated knowledge source, there are one or more *mapping microtheories* containing all of the structural declarations necessary to translate to and from the knowledge source and Cyc. Also, there are one or more *content microtheories* which allow access to the contents of the knowledge source. Any microtheory that can "see" a content microtheory has access to the part of the knowledge source that is mapped into the content microtheory. If two content microtheories that access different knowledge sources can both be seen by a common context of comparison microtheory, then any queries asked in the common microtheory can access and combine knowledge from both of the knowledge sources. Furthermore, semantically similar entity sets from the different sources can be semantically joined in the common microtheory. In the next section, we provide two working examples of queries that illustrate our current level of SKSI development by by extracting information from two structurally different sources.

Table 1: Example record from E_PERSON

Attribute name	Value
_ID	619
_SURNAME	"Aleksandr"
_LASTNAME	"Fogel"
_NICKNAME	"The Fox"
_COUNTRY_RESIDENCE	198

5 Example Cyc Queries

Cycorp's SKSI effort is already showing promising results at its current level of development, even though much of the work is yet to be completed. Our approach has been to provide the SKSI functionality needed to meet specific project goals and deliverables and then generalize the work. We are currently working with a seedling knowledge source developed for "link discovery experimentation" among collaborators in the DARPA Evidence Extraction and Link Discovery project. (see www.darpa.mil) The Veridian Seedling Schema[7] is a database containing information about assassinations committed by the Russian mafia. The data includes entity sets for people, events, locations, material, and others. With the Veridian Seedling Schema mapped into Cyc, we can ask queries that access the seedling data as well as any other knowledge in the KB.

The Veridian Seedling Schema contains the entity set E_PERSON with the attributes

_ID
_SURNAME
_FIRSTNAME
_NICKNAME
_COUNTRY_RESIDENCE

as well as others. At the schematic level, each instance of E_PERSON is a tuple of attribute values taken from primitive data types. The values of _SURNAME, _NICKNAME, and _LASTNAME are all strings with no more than 50 characters each, while _ID and _COUNTRY_RESIDENCE are positive integers. For _COUNTRY_RESIDENCE, its integer value is actually a reference to an instance of another entity set L_COUNTRIES whose physical instances are strings of no more than 255 characters. An example record appears in Table 1. Note that in the example record, the value of _COUNTRY_RESIDENCE is the integer 198. It is an indexical reference to the string "Russia," an instance of the L_COUNTRIES entity set.

At the semantic level, an instance of E_PERSON *is* a person, while the values of _SURNAME, _LASTNAME, and _NICKNAME together determine the

first, last, and nick names of the person, and the value of _COUNTRY_RESIDENCE is the country in which the person resides. Each attribute of E_PERSON (or any other entity set) is associated with one or more *meaning sentence templates*, which translate instances of the entity set, and its corresponding attribute value, into a CycL sentence that may be directly asserted in the approprate content microtheory. For example, the record above would be translated at the semantic level into the CycL sentences

(givenNames
(SchemaObjectFn E_Person-LS 619) "Aleksandr")

(lastName
(SchemaObjectFn E_Person-LS 619) "Fogel")

(residesInRegion
(SchemaObjectFn E_Person-LS 619) Russia)

These sentences may be added to the content microtheory for E_PERSON or saved in a batch translation file which can be added or removed from the knowledge base as needed. The physical and logical encodings, decodings, and CycL meaning sentence formulae are asserted in the mapping microtheory for E_PERSON. Note that (SchemaObjectFn E_Person-LS 619) is an indexed reference in the Schema Mapping Language to Alexandr Fogel, *the* person denoted by the unique id 619 in the logical schema for E_PERSON. Also note that for _COUNTRY_RESIDENCE, the indexical reference to an instance of L_COUNTRIES has been replaced by the actual referent, Russia.

Exam le #1

With the encodings, decodings, and meaning sentences for E_PERSON in place, we can extract records from the Veridian database from Cyc in the CycL language using the meaning sentences for E_PERSON with variables replacing the literals:

```
(thereExists ?PERSON
  (and
    (givenNames ?PERSON ?FIRSTNAME)
    (lastName ?PERSON ?LASTNAME)
    (residesInRegion ?PERSON
   ?REGION)))
```

Some results from asking this query in a context which can access the E_PERSON content microtheory are given in Table 2.

Table 2: Bindings for Example Query #1

?FIRSTNAME	?LASTNAME	?REGION
"Igor"	"Pilipchuk"	Ukraine
"Nikolai"	"Lysenko"	Russia
"Dmitry"	"Polevoi"	Belarus
"Alexi"	"Polevoi"	France
...	...	...

Exam le #2

In a previous project, we used a precursor database integration design to enable Cyc to query the IMDB on–line movie database (see us.imdb.com) so that Cyc queries about movies (concerning their titles, genres, lead actors, directors, etc.) are evaluated by querying the IMDB web site.

With both the Veridian database and the IMDB website now queriable through a common context in Cyc, we can ask queries that return data from both: Suppose we wish to find all (if any) persons named in the Veridian database who have nicknames that are also the titles of movies. The corresponding query in CycL that returns the person's last name and nickname is:

```
(thereExists ?ACTOR
  (thereExists ?PERSON
    (and
      (givenNames ?PERSON
        ?FIRSTNAME)
      (lastName ?PERSON ?LASTNAME)
      (nicknames ?PERSON ?NICKNAME)
      (movieActors
        (MovieNamedFn ?NICKNAME)
        ?ACTOR))))
```

Some results from asking this query in a context which can access both the E_PERSON content microtheory and the IMDB content microtheory are given in Table 3. Note that the table does not include the bindings for ?PERSON or ?ACTOR. These could be returned if desired, however there is not enough room between the column margins to include them in this example.

6 Schema Mapping Toolkit

Our SKSI efforts so far have been focused on developing the Schema Mapping Language and heuristic modules needed to make Cyc interoperate with structured knowledge sources. However, our long term

Table 3: Bindings for Example Query #2

?FIRSTNAME	?LASTNAME	?NICKNAME
"Aleksandr"	"Fogel"	"The Fox"
"Sergey"	"Timofeyev"	"Sylvester"
"Sergie"	"Kruglov"	"The Beard"
"Vitaliy"	"Roshchin"	"Tomcat"
"Mikhail"	"Besfamilnyy"	"Bes"
...	...	

goal is to use this core capability to develop marketable tools for database administration and information mining and analysis that exceed the power and utility of the current state of the art software. The *Schema Modeling Toolkit (SMT)* is the first step in that direction. The SMT will provide a core set of utilities needed to enable rapid knowledge source integration. It will include the following general capabilities:

- Add new knowledge sources to the Cyc knowledge base using graphical and dialog driven user interaction.
- Browse/visualize the schemata and contents of knowledge sources mapped into Cyc using graphical and dialog driven user interaction.
- Translate the contents of all or part of a knowledge source into CycL.
- Answer queries using supports from multiple knowledge sources as if they were already part of the Cyc KB.
- Mediate queries between loosely–federated knowledge sources using the Cyc knowledge base as a universal ontology.

As indicated above, the SMT will support a variety of methods for adding new knowledge sources to the system, as well as browsing their contents once they are integrated. The user will be able to:

- Generate UML diagrams, entity–relationship diagrams, etc., to assist in creating or viewing schemata and the contents of knowledge sources;
- Describe conceptual schemata using natural language and dialog tools created under the Rapid Knowledge Formation project at DARPA.

The Schema Mapping Toolkit may be extended to create applications which utilize the power of SKSI for multi–database management and information monitoring. For example,

- The SMT could be part of a comprehensive software product for database and multi–database management. In addition to the usual database management capabilities offered by existing products, an SKSI enabled database management product would have capabilities not found in the current state of the art.
- We could build more robust data monitoring applications on top of the SMT to work in data monitoring domains in which the content of one or more knowledge sources is changing over time, or when one or more of the knowledge sources is periodically publishing new information. The main extensions of the SMT would include the capabilities to:
  - batch translate updated knowledge into Cyc, maintaining a current state of a scenario of interest;
  - create and run sets of queries over the knowledge sources which monitor for interesting patterns in the data, or extract specific knowledge;
  - generate alerts when certain conditions (determined by the periodically asked monitior queries) are met, gather relavent supporting evidence for the alert, and post it to a client that handles the alert.

  This application concept grew out of our work on DARPA's Command Post of the Future project for which we are building a prototype Cyc Battle Monitor. This prototype has all three of the capabilities above, however is not scalable to general scenarios in its current form. See [10] for more information on using knowledge based applications in situation analysis.

References

[1] I. Horrocks et al. The ontology inference layer OIL.

[2] R. V. Guha. Micro–theories and contexts in Cyc part I: Basic issues. Technical Report ACT-CYC-129-90, Microelectronics and Computer Technology Corporation, Austin, TX, June 1990.

[3] D. D. Karunaratna, W. A. Gray, and N. J. Fiddian. Organising knowledge of a federated database system to support multiple view generation. In Alexander Borgida, Vinay K.

Chaudhri, and Martin Staudt, editors, *Proceedings of the 5th International Workshop on Knowledge Represenation Meets Databases (KRDB '98): Innovative Application Programming and Query Interfaces, Seattle, Washington, USA, May 31, 1998*, number 10 in CEUR Workshop Proceedings, pages 12.1–12.10, 1998.

[4] Vipul Kashyap and Amit P. Sheth. So far (schematically) yet so near (semantically). In *Proceedings of the IFIP TC2/WG2.6 Conference on Semantics of Interoperable Database Systems*, volume DS-5 of *IFIP Transactions*. North Holland, November 1992.

[5] Vipul Kashyap and Amit P. Sheth. Semantic and schematic similarities between database objects: A context-based approach. *VLDB Journal: Very Large Data Bases*, 5(4):276–304, 1996.

[6] D. B. Lenat and R. V. Guha. *Building Large Knowledge Based Systems.* Addison-Wesley, 1990.

[7] Steven J. McKay, Paul N. Woessner, and Trifin J. Roule. *Evidence Extraction and Link Discovery Seedling Project: Database Schema Description, Version 1.0.* Veridian Systems Division, 1400 Key Blvd, Suite 100, Arlington, VA 22209, August 2001.

[8] Heiner Stuckenschmidt and Holger Wache. Context modeling and transformation for semantic interoperability. In Mokrane Bouzeghoub, Matthias Klusch, Werner Nutt, and Ulrike Sattler, editors, *Proceedings of the 7th International Workshop on Knowledge Representation meets Databases (KRDB 2000), Berlin, Germany, August 21, 2000*, number 29 in CEUR Workshop Proceedings, pages 115–126, 2000.

[9] Jeffrey D. Ullman. *Principles of Database and Knowledge–Base Systems.* Principles of Computer Science Series. Computer Science Press, 1988.

[10] Amanda Vizedom, Raymond A. Liuzzi, and Mark Foresti. Knowledge based support for decision making using fusion techniques in a C2 environment. In *Proceedings of the Fourth International Conference on Information Fusion.* International Society of Information Fusion, 2001.