KBB: A Knowledge-Bundle Builder for Research Studies

David W. Embley[1⋆], Stephen W. Liddle[2], Deryle W. Lonsdale[3], Aaron Stewart[1], and Cui Tao[4]

[1] Department of Computer Science,
[2] Information Systems Department,
[3] Department of Linguistics,
Brigham Young University, Provo, Utah 84602, U.S.A.
[4] Mayo Clinic, Rochester, Minnesota 55905, U.S.A.

Abstract. Researchers struggle to manage vast amounts of data coming from hundreds of sources in online repositories. To successfully conduct research studies, researchers need to find, retrieve, filter, extract, integrate, organize, and share information in a timely and high-precision manner. Active conceptual modeling for learning can give researchers the tools they need to perform their tasks in a more efficient, user-friendly, and computer-supported way. The idea is to create "knowledge bundles" (KBs), which are conceptual-model representations of organized information superimposed over a collection of source documents. A "knowledge-bundle builder" (KBB) helps researchers develop KBs in a synergistic and incremental manner and is a manifestation of learning in terms of its semi-automatic construction of KBs. An implemented KBB prototype shows both the feasibility of the idea and the opportunities for further research and development.

1 Introduction

In many domains, the volume of data is enormous and increasing rapidly. Unfortunately, the information a researcher requires is often scattered in various repositories and in the published literature. Researchers need a system that can help efficiently locate, extract, and organize available information so they can analyze it and make informed decisions.

We address this challenge with the idea of a *Knowledge Bundle* (*KB*) and a *Knowledge-Bundle Builder* (*KBB*). Active conceptual modeling for learning (ACM-L) is at the core of our approach. As we explain below, a KB includes an extraction ontology, which allows it to both identify and extract information with respect to a custom-designed schema. (This constitutes the conceptual-modeling part of ACM-L.) Construction of a KB itself can be a huge task—but one that is mitigated by the KBB. Construction of the KB under the direction of the KBB proceeds as a natural progression of the work a researcher does

⋆ Supported in part by the National Science Foundation under grant #0414644

to manually identify and gather information of interest. As a researcher begins to work, the KBB immediately begins to synergistically assist the researcher and quickly "learns" and is able to take over most of the tedious work. (This constitutes the active-learning part of ACM-L.)

In describing our KBB approach to building KBs, we first give a motivating example of a bio-research study (Section 2). We then explain how the KBB plays its claimed role in the bio-research scenario (Section 3) by defining what a KB is and giving specific examples of KBB tools for building and using KBs. Finally, we give the status of our implementation and mention current and future work needed to enhance KBs and the KBB (Section 4) and then draw conclusions (Section 5).

2 Motivation Scenario

Suppose a bio-researcher B wishes to study the association of TP53 polymorphism and lung-cancer. To do this study, B wants information from the NCBI dbSNP repository[5] about SNPs (chromosome location, SNP ID and build, gene location, codon, and protein), about alleles (amino acids and nucleotides), and about the nomenclature for amino-acid levels and nucleotide levels. B also needs data about human subjects with lung cancer and needs to relate the SNP information to human-subject information.

To gather information from dbSNP, B constructs the form in the left panel in Figure 1. Form construction consists of selecting form-field types—e.g., single-value fields, multiple-value fields, multiple-column/multiple-value fields, radio buttons, and check boxes—and organizing and nesting them so that they are a conceptualization of the information B wishes to harvest for the research study. B next finds a first SNP page in dbSNP from which to begin harvesting information. (The created form and located page need not have any skpecial correspondence—no schema correspondence, no name correspondence, and no special structure requirements—but, of course, the page should have data of interest for the research study and thus for the created form.) B then fills in the form by cut-and-paste actions, copying data from the page in the center panel in Figure 1 to the form in the left panel.

To harvest similar information from the numerous other dbSNP pages, B gives the KBB a list of URLs, as the right panel in Figure 1 illustrates (although there would likely be hundreds rather than just the six in Figure 1). The KBB automatically harvests the desired information from the dbSNP pages referenced in the URL list. Since one of the challenges bio-researchers face is searching through the pages to determine which ones contain the desired information, the KBB provides a filtering mechanism. By adding constraints to form fields, bio-researchers can cause the KBB harvester to gather information only from pages that satisfy the constraints. B, for example, might only want coding SNP data

[5] The Single Nucleotide Polymorphism database (dbSNP) is a public-domain archive for a broad collection of simple genetic polymorphisms hosted by National Center for Biotechnology Information (NCBI) at www.ncbi.nlm.nih.gov/projects/SNP/.

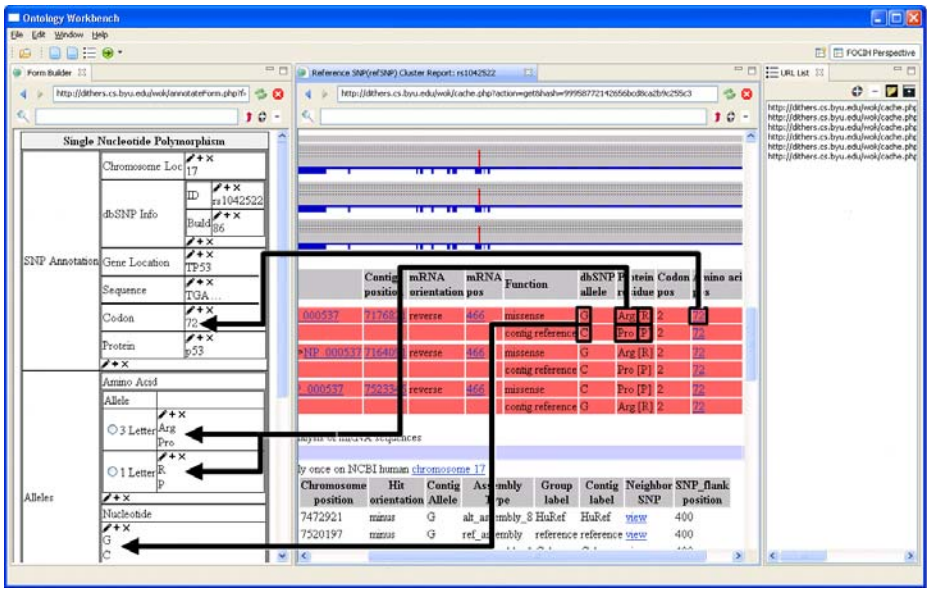

Fig. 1. Form Filled in with Information from an SNP Page.

with a significant heterogeneity (i.e., minor allele frequency $> 1\%$). Because of this filtering mechanism, B can direct the KBB to search through a list of all pages without having to first limit them to just those with relevant information.

For the research scenario, B may also wish to harvest information from other sites such as GeneCard. B can use the KBB with the same form to harvest from as many sites as desired. Interestingly, however, and as an example of the learning that takes place, once the KBB harvests from one site, it can use the knowledge it has already gathered to do some of the initial cut-and-paste for B. In addition to just being a structured knowledge repository, the KB being produced also becomes an extraction ontology capable of recognizing data items it has already seen. It can also recognize data items it has not seen but are like the data it has seen—e.g., numeric values or DNA snippets.

Using KBs as extraction ontologies also lets bio-researchers search the literature. Suppose B wishes to find papers related to the information harvested from the dbSNP pages. B can point the KBB to a repository of papers to search and cull out those that are relevant to the study. Using the KB as an extraction ontology provides a sophisticated query of the type used in information retrieval resulting in high-precision document filtering. For example, the extraction ontology recognizes the highlighted words and phrases in the portion of the paper in Figure 2. With the high density of not only keywords but also data values and relationships all aligned with the ontological KB, the KBB can designates this paper as being relevant for B's study.

cer
rt-
DV
cer
tio
tlv
in various cancers. The gene is located on chromosome 17p13 and is one of the most commonly mutated genes in all of the human cancers (27, 28). The codon 72 *p53* polymorphism is a result of a single bp substitution: guanine is replaced by cytosine leading to an arginine (*Arg*) replaced by proline (*Pro*). The wild-type *p53* gene operates by

Fig. 2. Paper Retrieved from PMID Using an Extraction Ontology.

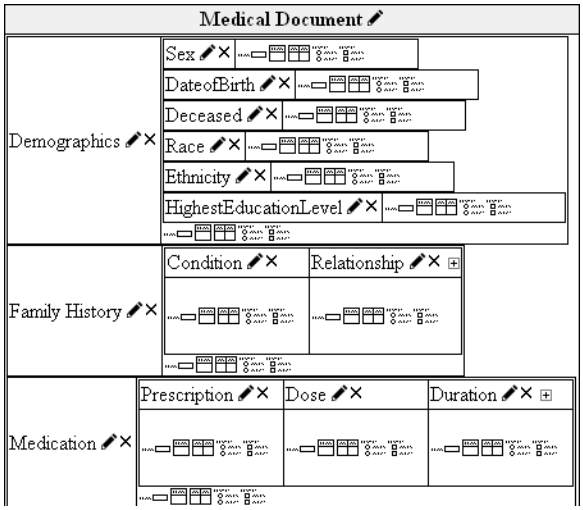

Fig. 3. Some Human Subject Information Reverse-Engineered from INDIVO.

For the human-subject information and to illustrate additional capabilities of the KBB, we suppose that a database exists that contains the needed human-subject information. The KBB can automatically reverse-engineer the database to a KB, and present B with a form representing the schema of the database. B can then modify the form, deleting fields not of interest and rearranging fields to suit the needs of the study. Further, B can add constraints to the fields so that the KBB only gathers data of interest from the database to place in its KB. Figure 3 shows an example of a form reverse-engineered from the INDIVO database, tailored to fit our research scenario. The icons in the form let a user tailor the form: modify a form-field title (pencil icon), delete a form field ($\times$ icon), insert or nest a new form field (choice list of icons to insert respectively a single-value form field, a multiple-value form field, a multiple-column/multiple-value form field, and radio-button and check-box selection fields).

With all information harvested and organized into an ontology-based knowledge bundle (the KB), B can now issue queries and reason about the data to do some interesting analysis. Figure 4 shows a sample SPARQL query over the data harvested from the pages referenced by the six URLs listed in Figure 1. The query finds three SNPs that satisfy the query's criteria and for each, returns the

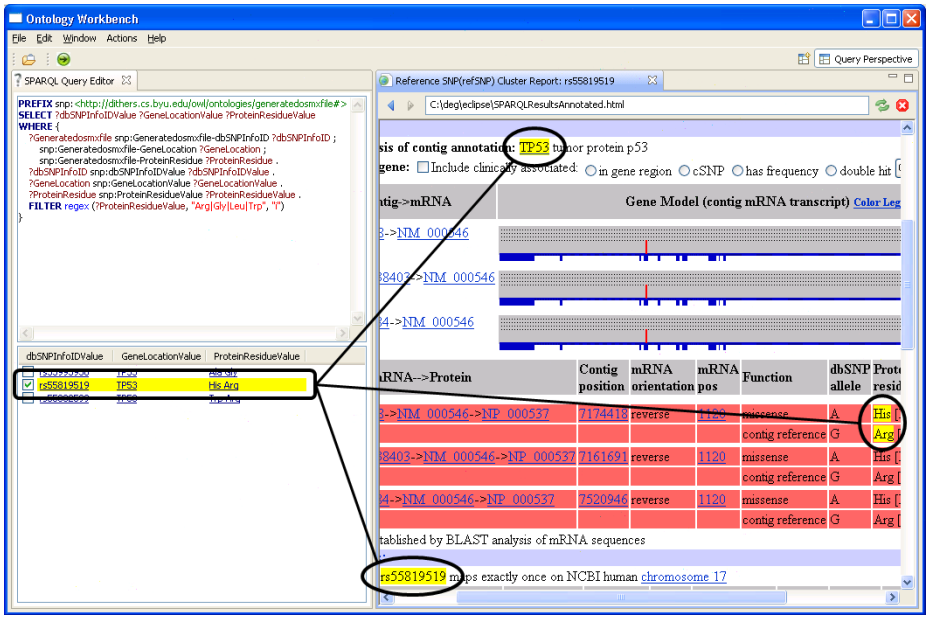

Fig. 4. Screenshot of our Web of Knowledge Prototype.

dbSNP ID, the gene location, and the protein residue it found. In our prototype, users may click on any of the displayed values to display the page from which the value was extracted and to highlight the value in the page. As Figure 4 shows, users may alternatively click on one or more check boxes to access and highlight all the values in checked rows. The values *rs55819519*, *TP53*, and *His Arg* are all highlighted in the page in the right panel of Figure 4.

3 KBs and KBBs

Having provided a scenario in which a researcher can use KBs built synergistically through a KBB, we now explain exactly what a KB is and how a KBB synergistically builds them. In doing so, we emphasize that although our research-study scenario specifically targets bio-research, our definitions and explanation here do not. It should be clear that a KBB can assist intelligence-gathering researchers in all areas—scientific, business, military, and government.

We define a *knowledge bundle* (KB) as a 7-tuple (O, R, C, I, S, A, F):

- O is a set of intensional object sets, and are one-place predicates—sometimes called concepts or classes; they may also play the role of properties or attributes. (Examples: $Person(x)$, $Amino\ Acid(x)$, $Country(x)$, $Color(x)$.)
- R is a set of intensional relationship sets among the object sets, and are n-place predicates ($n \geq 2$). (Examples: $Person(x)$ *is citizen of* $Country(y)$, $Sample(x)$ *taken on* $Date(y)$.)

- C is a set of constraints over O and R, limited so that (O, R, C) constitutes a decidable fragment of first-order logic. (Examples: $\forall x(Student(x) \Rightarrow Person(x))$, $\forall x(Sample(x) \Rightarrow \exists^1 y(Sample(x)\ taken\ on\ Date(y)))$
- I is an instantiation of the object and relationship sets in O and R; when I satisfies C, I is a model for (O, R, C). (Examples: $Sample$("SMP9671") $taken\ on$ $Date$(2009-03-25), $Color$("green").)
- S is a set of inference rules, horn-clause statements. (Example: $BrotherOf(x, y)$:- $Person(x)$, $Person(y)$, $SiblingOf(x, y)$, $Male(x)$.)
- A is a set of annotations for data-value instances in object sets in O; each data value v may link to an appearance of v in a source document. (Example: $Codon(72)$ may link to the appearance of 72 in the SNP page in Figure 1.)
- F is a set of data frames [Emb80]. Data frames are abstract data types, linguistically augmented to include recognizers for object and relationship instances and operation instantiations as they appear in documents and free-form user queries. (Examples: the instance recognizer [ACGT]([ACGT])$^+$ for a DNA snippet, (Country | Nation | Republic | ...) as keywords indicating the presence of a country concept.)

The triple (O, R, C) is an *ontology*.[6] In our implementation, we use OWL to represent ontologies. Adding the I component allows us to populate the ontology. The quadruple (O, R, C, I) characterizes *information* and is an information system or database. In our implementation, we use RDF for storing instances with respect to OWL ontologies. The quintuple (O, R, C, I, S) characterizes a *computational view of knowledge*. Adding the S component allows us to reason over the base facts in the information system. In our implementation, we use SWRL rules and the Pellet reasoner. The sextuple (O, R, C, I, S, A) characterizes a *platonic view of knowledge*. Adding the A component provides a form of authentication since users can trace knowledge back to its source; it thus provides a form of "justified true belief," which Plato insists is part of the definition of knowledge [PlaBC]. Completing the septuple by adding the F component, *linguistically grounds* the knowledge [BCHS09,HLF$^+$08],[7] making the KB also be an extraction ontology. Further, having an extraction ontology enables a KB to be an active learner, where we consider *active learning* to be the ability to automatically find facts in source documents that pertain to the KB's ontology, annotate them, and add them to the KB.

Finding facts in source documents and adding them to the bundle of collected knowledge is the essence of building KBs for research studies. Letting KBs themselves assist in the task goes a long way toward automating the KB-building process. This automation is non-trivial, and full automation is likely

[6] Researchers disagree about the definition of an ontology, but we adopt the view that an ontology is a formal theory captured in a model-theoretic view of data within a formalized conceptual model. Since the elaboration of our triple (O, R, C) is a predicate-calculus-based, formalized, conceptual model, we call it an ontology.

[7] Both LexInfo [BCHS09] and OpenDMAP [HLF$^+$08] are independently developed complementary, efforts, aimed at linguistically grounding ontologies. As both their work and ours explores this wide-open research area, the projects have much to contribute to and learn from each other.

impossible. Hence, we aim to construct KB-building tools that synergistically work well with users and incrementally take on more and more of the burden of KB construction.

A *KB-Builder* (*KBB*) is a tool suite to aid in the construction of KBs. More specifically, it is a tool suite to largely automate the building of KBs. In our approach to providing a KBB tool suite, we focus on tools (1) to build KBs via form specification and automated information harvesting and (2) to reverse-engineer structured and semi-structured information sources into KBs.

Form-based Ontology Creation and Information Harvesting. While we do not assume that bio-researchers and other decision-making researchers are expert users of ontology languages, we do assume that they can create ordinary forms of the kind people routinely use for information gathering. A KBB interface lets users create forms by adding various form elements as the clickable icons in the data and label fields of the form in Figure 3 indicates. Users can specify any and all concepts needed for a study, can specify relationships and constraints among the concepts, and can nest, customize, and organize their data as they wish. From a form specification, the KBB generates a formal ontological structure, (O, R, C). Each label in a form becomes a concept of O. The form layout determines the relationship sets in R among the concepts and determines the constraints in C over the concepts and relationship sets. Given a form, a user can cut-and-paste data from source documents into the form fields to create the I and A components of a KB. When harvesting from sites like the NCBI dbSNP repository which have hundreds of pages all formatted in a similar way, the KBB can infer from the user's cut-and-paste actions the patterns it needs to harvest the desired information from all pages on the site. These patterns consist of paths in DOM trees of HTML pages along with left and right context and list delimiters to locate data within DOM-tree nodes. To build the F component of a KB, the KBB creates instance recognizers in two ways as it harvests information: (1) by creating lexicons and (2) by identifying and specializing data frames in a data-frame library. For lexicons, the KBB simply makes a list of names of identifiable entities, which it can then later recognize and classify. For data-frame recognizers, we initialize a data-frame library with data frames for common items we expect to encounter—e.g., all types of numbers, currencies, postal codes, and telephone numbers, among many others. When recognizers in these data frames recognize harvested items, they can classify the items with respect to these data frames and associate the data frames with concepts in the ontology. Some automatic specializations are possible, such as numbers with as-yet-unknown units. For more complex pattern recognition, experts can add recognizers.

Reverse-Engineering Structured and Semi-structured Data to KBs. Structured repositories (e.g., relational databases, OWL/RDF triple stores, XML document repositories) and semi-structured repositories (e.g., human-readable tables and forms, hidden-web display pages) may contain much of the information needed for a research study. For structured repositories, reverse-engineering processes (e.g., for relational databases [MH08]) can turn these repositories into knowledge bundles. Further, the results of reverse engineering can be nested form

schemas like the one in Figure 3. In this case, researchers can use the techniques mentioned in the previous paragraph to custom-tailor reverse-engineered KBs by restructuring the generated forms to become the (O, R, C)-ontologies they want. They can also limit the data extracted from the database to the I-values they want, and they can use the techniques mentioned in the previous paragraph to produce F-component lexicons and data frames. For structured repositories such as relational databases that allow view definitions, S-component construction is possible, yielding rules for reasoning. Although the reverse-engineering process for semi-structured repositories is even more challenging than for structured repositories, it is nevertheless feasible for many documents (e.g., for human-readable web tables [GBH+07,PSC+07]).

4 Implementation Status and Future Work

We have implemented an initial prototype of our KBB as part of our Web-of-Knowledge (WoK) project [ELL+08]. Currently, as Figure 1 shows, our prototype lets users create ontologies via forms, fill in the form from a machine-generated web page in a hidden-web site, and harvest information from the remaining sibling pages of the hidden-web site [Tao08,TEL09]. We have not yet, however, added constraint filtering to forms. Our WoK prototype can also automatically reverse-engineer machine-generated sibling tables from hidden-web sites into forms and automatically establish the beginnings of a KB extraction ontology [Tao08]. Although not yet integrated into our WoK prototype, we have implemented a way to reverse-engineer an XML-Schema document into a conceptual model, which is compatible with our KB ontologies [AKEL08]. Using extraction ontologies coded by hand, we can successfully do high-precision filtering of semi-structured web documents [XE08], but we have not yet brought this up to the level we need for high-precision document retrieval for free-running text as indicated in Figure 2. In another WoK subproject we have developed a way to generate an ontology from a collection of human-readable tables. We can interactively interpret tables [Pad09], semantically enhance them [LE09], and merge them into a growing ontology [Lia08] using automated schema integration techniques [XE06]. We have yet to make all these components work together to achieve the overall goal of automatically growing ontologies by reverse-engineering coordinated collections of human-readable tables. The current implementation of our WoK prototype also allows users to access and query the data in a KB as the screenshot in Figure 4 shows.

Although some of our work is complete, we still have much to do to solidify and enhance what we have already implemented and to extend it to be a viable research-study tool. We plan further research as follows. (1) We have defined and implemented data frames for concepts corresponding to nouns and adjectives, but we should also define data frames for relationships in connection with verbs and prepositions. (2) Our current system expects source documents divided into distinct records, but to extract selected information from free-running text, we need to relax the record-boundary constraints and be able to recognize a

record of interest, and its extent, without any boundary information. (3) Our reverse-engineering efforts have proven to be successful, but we should take these approaches even further, for instance, by inferring schemas from general semi-structured data like the dbSNP page in Figure 1. (4) Although not a scientific workflow system by itself, a KBB can become an integral part of a workflow system; embedding a KBB inside of a workflow system being used to gather information for research studies (e.g., scientific workflow systems [LAB+06]) could greatly enhance and help automate the information harvesting facilities of these systems.

5 Concluding Remarks

Several related fields of research are at the heart of our work: information extraction [Sar08], information integration [ES07], ontology learning [Cim06], and data reverse engineering [Aik98]. The KB/KBB approach discussed here is a unique, synergistic blend of techniques resulting in a tool to efficiently locate, extract, and organize information for research studies. (1) It supports directed, custom harvesting of high-precision technical information. (2) Its semi-automatic mode of operation largely shifts the burden for information harvesting to the machine. (3) Its synergistic mode of operation allows research users to do their work without intrusive overhead. The KB/KBB tool is a helpful assistant that "learns as it goes" and "improves with experience."

References

[Aik98] P.H. Aiken. Reverse engineering of data. *IBM Systems Journal*, 37(2):246–269, 1998.

[AKEL08] R. Al-Kamha, D.W. Embley, and S.W. Liddle. Foundational data modeling and schema transformations for XML data engineering. In *Proceedings of the 2nd International United Information Systems Conferences (UNISCON'08)*, pages 25–36, Klagenfurt, Austria, April 2008.

[BCHS09] P. Buitelaar, P. Cimiano, P. Haase, and M. Sintek. Towards linguistically grounded ontologies. In *Proceedings of the 6th European Semantic Web Conference (ESWC'09)*, pages 111–125, Heraklion, Greece, May/June 2009.

[Cim06] P. Cimiano. *Ontology Learning and Population from Text: Algorithm, Evaluation and Applications*. Springer Verlag, New York, New York, 2006.

[ELL+08] D.W. Embley, S.W. Liddle, D. Lonsdale, G. Nagy, Y. Tijerino, R. Clawson, J. Crabtree, Y. Ding, P. Jha, Z. Lian, S. Lynn, R.K. Padmanabhan, J. Peters, C. Tao, R. Watts, C. Woodbury, and A. Zitzelberger. A conceptual-model-based computational alembic for a web of knowledge. In *Proceedings of the 27th International Conference on Conceptual Modeling*, pages 532–533, Barcelona, Spain, October 2008.

[Emb80] D.W. Embley. Programming with data frames for everyday data items. In *Proceedings of the 1980 National Computer Conference*, pages 301–305, Anaheim, California, May 1980.

[ES07] J. Euzenat and P. Shvaiko. *Ontology Matching*. Springer-Verlag, Heidelberg, Germany, 2007.

[GBH+07] W. Gatterbauer, P. Bohunsky, M. Herzog, B. Krüpl, and B. Pollak. Towards domain-independent information extraction from web tables. In *Proceedings of the Sixteenth International World Wide Web Conference (WWW2007)*, pages 71–80, Banff, Alberta, Canada, May 2007.

[HLF+08] L. Hunter, Z. Lu, J. Firby, W.A. Baumgartner Jr., H.L. Johnson, P.V. Ogren, and K.B. Cohen. OpenDMAP: An open source, ontology-driven, concept analysis engine, with applications to capturing knowledge regarding protein transport, protein interactions and cell-type-specific gene expression. *BMC Bioinformatics*, 9(8), 2008.

[LAB+06] B. Ludascher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E.A. Lee, J. Tao, and Y. Zhao. Scientific workflow management and the Kepler system. *Concurrency and Computation: Practice and Experience*, 18(10):1039–1065, 2006.

[LE09] S. Lynn and D.W. Embley. Semantically conceptualizing and annotating tables. In *Proceedings of the Third Asian Semantic Web Conference*, pages 345–359, Bangkok, Thailand, February 2009.

[Lia08] Z. Lian. A tool to support ontology creation based on incremental mini-ontology merging. Master's thesis, Department of Computer Science, Brigham Young University, Provo, Utah, March 2008.

[MH08] N.A. Mian and T. Hussain. Database reverse engineering tools. In *Proceedings of the 7th WSEAS International Conference on Software Engineering, Parallel and Distributed Systems*, pages 206–211, Cambridge, United Kingdom, February 2008.

[Pad09] R.K. Padmanabhan. Table abstraction tool. Master's thesis, Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, New York, May 2009.

[PlaBC] Plato. *Theaetetus*. BiblioBazaar, LLC, Charleston, South Carolina, about 360BC. (translated by Benjamin Jowett).

[PSC+07] A. Pivk, Y. Sure, P. Cimiano, M. Gams, V. Rajkovič, and R. Studer. Transforming arbitrary tables into logical form with TARTAR. *Data & Knowledge Engineering*, 60:567–595, 2007.

[Sar08] S. Sarawagi. Information extraction. *Foundations and Trends in Databases*, 1(3):261–377, 2008.

[Tao08] C. Tao. *Ontology Generation, Information Harvesting and Semantic Annotation for Machine-Generated Web Pages*. PhD dissertation, Brigham Young University, Department of Computer Science, December 2008.

[TEL09] C. Tao, D.W. Embley, and S.W. Liddle. FOCIH: Form-based ontology creation and information harvesting. In *Proceedings of the 28th International Conference on Conceptual Modeling (ER2009)*, Gramado, Brazil, November 2009. (in press).

[XE06] L. Xu and D.W. Embley. A composite approach to automating direct and indirect schema mappings. *Information Systems*, 31(8):697–732, December 2006.

[XE08] L. Xu and D.W. Embley. Categorization of web documents using extraction ontologies. *International Journal of Metadata, Semantics and Ontologies*, 3(1):3–20, 2008.