Navigating Virtual Information Sources with Know-ME

Xufei Qian[1] **Bertram Ludäscher**[1] **Maryann Martone**[2] **Amarnath Gupta**[1]

[1]San Diego Supercomputer Center
[2]Department of Neuroscience
University of California San Diego

Introduction

In many application domains such as biological sciences, information integration faces a challenge usually not observed in simpler applications. Here, the to-be-integrated information sources come from very different sub-specialties (e.g., anatomy and behavioral neuroscience) and have widely diverse schema, often with very little overlap in attributes. Yet, they can be *conceptually* integrated because they refer to different aspects of the same physical objects or phenomena. In [1], we proposed *model-based mediation* (MBM) as a solution to this problem. MBM is an information integration paradigm where information sources with hard-to-correlate schemas may be integrated using *auxiliary expert knowledge* that provides domain-specific "glue information" to hold together widely different data schemas. The expert knowledge is captured in a data structure called the ***Knowledge Map*** (described below). Aside from the use of global domain knowledge, MBM has an important difference from current approaches to information mediation. In current mediator systems, the integrated view definition (IVD) is defined over the exported *logical schema* of sources (often through an XML query language). In MBM we extend this architecture by lifting exported source data from the level of uninterpreted data to the semantically rich level of ***conceptual models*** (CMs) that represent more local knowledge than a logical schema. The mediator's IVDs are defined in terms of source CMs (global-as-view) and hence make use of a semantically richer model involving class hierarchies, complex object structure and rule-defined properties of relationships (e.g, local domain constraints). Additionally, sources specify ***object contexts***, i.e., formulas that relate their conceptual schema with the global domain knowledge maintained at the mediator. Thus model-based mediation has a hybrid approach to information integration – on the one hand at the mediator IVDs are defined over source CMs and the Knowledge Map using a global-as-view approach; on the other hand, object-contexts of the source are defined as local-as-view.

In this demonstration paper, we present a system called Knowledge Map Explorer (***Know-ME***) that shows how the user can use the Knowledge Map as a backbone structure to explore both the domain knowledge itself, as well as all data sources that have been integrated using it.

The Knowledge Map

Formally, the knowledge map is a multigraph $G=\{V, E, \lambda_{conc}, \lambda_{proc}, \mu_{proc}, \mu_{role}, \mu_{context}\}$ with vertices V, edges E, *concept* and *process* labeling functions λ_{conc} and λ_{proc} respectively, labeling functions for *process edges*, *role edges* and *context edges* μ_{proc}, μ_{role} and $\mu_{context}$ respectively, together with a set of logic rules Φ (a *logic program*) whose atomic formulas are built using symbols in G.

The Knowledge Map (KM) can be viewed as two intertwined subgraphs:

- a *Domain Map* (DM) relates *concepts* through *roles* as shown. The nodes of the DM take their names from the namespace λ_{conc} and the edges take their names from μ_{role}:

  $$map\_kinase \xrightarrow{isa} enzyme$$

  Here, *isa* defines the concept hierarchy and is a special role. Other roles include the "has_a" hierarchy, spatial relationships such as "inside", and domain-specific roles such as "projects_to".

- a *Process Map* (PM) expresses temporal and causal relationships between state-changing *processes* or *events*. Edge labels (like "activates" below) are drawn from the process namespace μ_{proc} and can have

  $$state\_i \xrightarrow{activates(map\_kinase, protein\_kinase\_A)} state\_j$$

  parameters that correspond to DM nodes. *Nodes* denote states and have unique identifiers but no external labels. An *edge* in a process map denotes a state transition which causes a fluent such as active(protein_kinase_A) to change its truth value say from *false* to *true* as a result of the process:

- edges of a PM (processes) and nodes from a DM (concepts) may share the same label L (so $L \in \mu_{proc} \cap \lambda_{conc}$). This means that the PM contains the process description of L and the DM contains the concept representing the phenomenon of L. For example, L-LTP (late long-term potentiation) is a process:

  The DM may have a concept node for L-LTP, with roles like:

$$state\_i \xrightarrow{E\_LTP} state\_j \xrightarrow{L\_LTP} state\_k$$

$$L\_LTP \xrightarrow{occurs\_in} hippocampus$$

  Thus there is an (implicit) *interface link* connecting process edges and concept nodes with identical labels (here: L_LTP).

- *context edges* relate concepts or events to actual data. For example, a source src_1 that has immunolabeling images that may serve as the evidence of the process activates(map_kinase, protein_kinase_A). Then

$$src_1 \xrightarrow{has\_evidence(immuno\_image)} activates(map\_kinase, protein\_kinase\_A)$$

  shows a parameterized context edge in the DM connecting src_1 to the process.

- logic rules Φ formalize edge-properties like "edge label **tc**(**is-a**) is the *transitive closure* of the edge label **is-a**", or edge-derivation rules like "has-a(X,Z) **if** is-a(X,Y) **and** has-a(Y,Z)".

Knowledge-Map Explorer (KNOW-ME) enables a user to navigate a knowledge map KM by performing exploratory graph queries over KM. In our setting, the user always has some part of the KM on the screen, and explores this graph by performing relativized queries. A *relativized query Q* is one that is applied to the *visible fragment* of the KM. As a result of a query, the visible part of the graph gets modified – edges from the Knowledge Map or source data get added, or parts of the originally visible graph are removed from the screen.

The Demonstration System

System Architecture. A block diagram of the prototype MBM system is shown in Figure 1. The Knowledge Map database stores the graph structure of the domain and process maps. Currently, the database consists of the Unified Medical Language System[1] ontology from the National Library of Medicine and the Gene Ontology from the Gene Ontology Consortium[2]. The two ontologies together store about 8 million concepts and 10 million relationships (edges) represented as relational tables in an Oracle 8 database. In the demonstration system, the wrappers connect to Neuroscience information sources that are either web-pages containing published experimental results, or XML documents storing experimental data at UCSD. The logic engine is XSB Prolog v2.4 from SUNY at Stony Brook that comes with FLORA, an F-Logic preprocessor implementing an F-Logic engine on top of XSB Prolog. The rules pertaining to the Knowledge Map are implemented in F-Logic. The graph processor and coordinating query engine have been developed by the authors of the paper. The query engine is the central evaluation unit of the mediator and uses the graph processor and the logic engine as required. Integrated views are defined over the source schemas and the Knowledge Map using F-Logic. The KNOW-ME tool is a query formulation front-end that drives the query engine by sending it relativized and possibly parameterized queries.

---

[1] http://www.nlm.nih.gov/research/umls/index.html

[2] http://www.geneontology.org

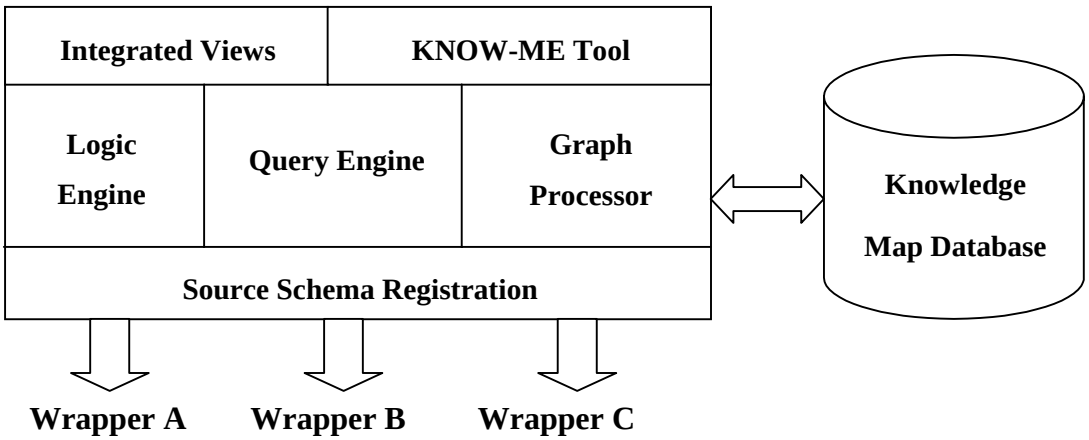

Figure 1. The architecture of the Model-based Mediator

The KNOW-ME Tool. The KNOW-ME tool presents to the user a two-window interface showing the DM and the PM respectively. In either window, the user needs to create an ***initial graph*** from which she starts exploration. In the default case, the system presents the user with only the nodes that represent root-level objects (i.e., a generic node called OBJECT and a generic edge called PROCESS between two unnamed nodes) and the user starts expanding the graph by querying on their neighboring nodes and edges. In a more involved case, the user selects from a number of concepts, roles and processes, and asks the system to compute a connected subgraph that contains the chosen nodes and edges. For this request, the graph processor computes an approximate solution to the directed Steiner tree problem [2] to construct the initial graph.

Once the initial graph is obtained, the user may select a node, edge or subgraph to launch the next relativized query. The KNOW-ME tool internally maintains a list of all ***visible***, ***active*** and ***response*** subgraphs (which can be just a single DM node or a PM edge). A ***visible*** subgraph is a fragment of a Knowledge Map that appears on the screen at the time of the next operation; an ***active*** subgraph is one that has been highlighted by the user as the "argument" of the next operation; and a ***response*** subgraph is returned as the answer to a query. These are represented with different shapes (for nodes) and colors (for edges) on the interface. The user may manually deactivate an active subgraph or set a preference to automatically deselect the active subgraph after a query has been evaluated and make the response subgraph active. After selecting the active subgraph the user can right-click to open a query menu and perform either a basic operation or a predefined parameterized query. An active subgraph of the PM can be seen (in yellow) in the prototype interface shown in Figure 2, while the diamonds in the DM shows evidence edges leading to data (not shown for clarity).

The basic operations of the KNOW-ME tool are graph queries expressed as generalized path expressions (GPE); see [3] for the details of the semantics:

an edge with some label L or *any* is a GPE

if M and N are GPEs then M.N, M|N, M*, M^{+}, $M^{?}$, M^{k}, M^{-1} are GPEs

if φ is a relation symbol then **if**(φ) is a GPE. Thus if edge *e* connects node n_1 to node n_2 and n_1 is the active node, then the system will return n_2 only if *e* satisfies the condition **if**(φ).

if *e* is a process edge, then **elab**(*e*) is a GPE. The operation **elab**(*edge*) (i.e., elaboration of *e*) substitutes the edge e with a path $<e_1, e_2, \ldots e_k>$ that has been logically defined as a more elaborate description of the process denoted by e. Thus the edge

$$state\_i \xrightarrow{E-LTP} state\_j \xrightarrow{L-LTP} state\_k$$

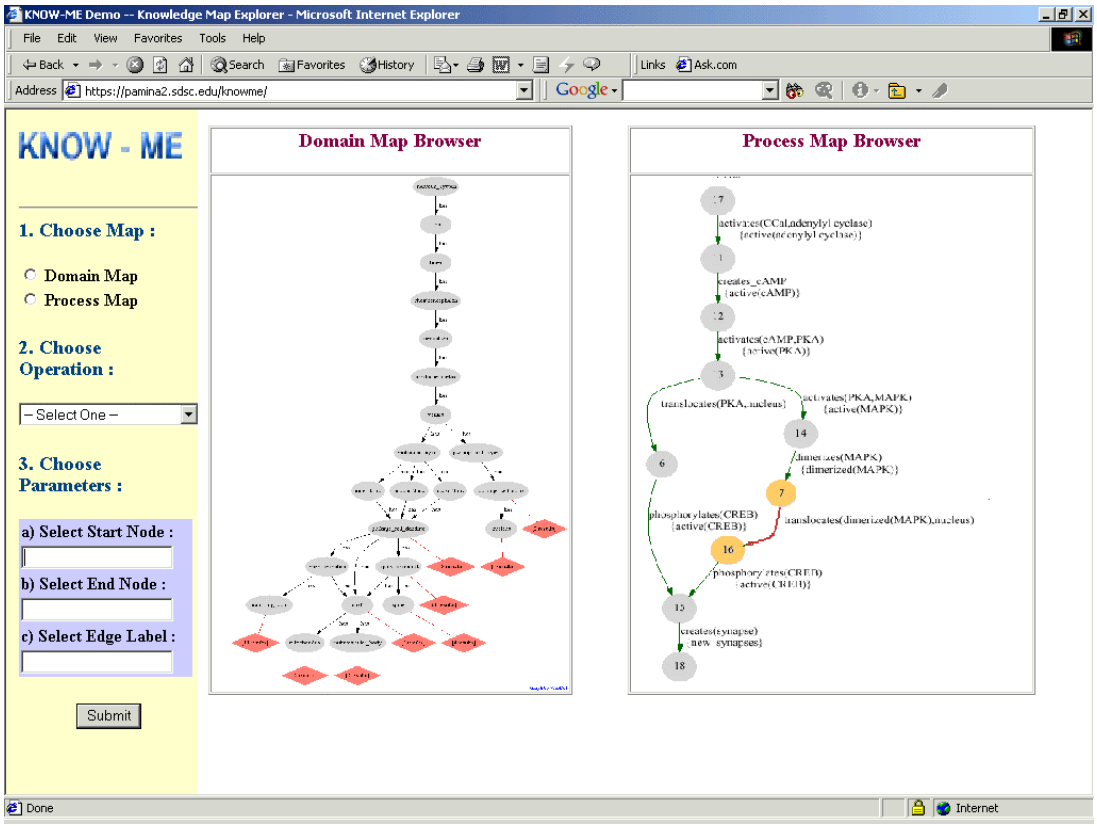

Figure 2. The interface of the prototype implementation

is an elaboration of the edge

if $\theta = \langle e_1, e_2, \ldots e_k \rangle$ is a path in the process map then **abst**(θ) is a GPE. The operation **abst**(*path*) is called *abstraction* and is the reverse of process elaboration.

The user can formulate more complex queries by selecting the "create new query" option in the query menu. Once created a query can be saved for future use. The demo will exhibit a number of pre-defined complex queries.

References

1. B. Ludäscher, A. Gupta, M. E. Martone, "Model-Based Mediation with Domain Maps", *Proc. ICDE*, 2001, pp. 81-90.

2. M. Charikar, C. Chekuri, T. Cheung, Z. Dai, A. Goel, S. Guha, M. Li, "Approximation algorithms for directed Steiner problems". *J. Algorithms*, (1):33, 1999, pp. 73-91.

3. B. Ludäscher, R. Himmeröder, G. Lausen, W. May, C. Schlepphorst, "Managing Semistructured Data with FLORID: A Deductive Object-Oriented Perspective", *Information Systems*, 23(8), 1998, pp. 589-613.