Inference Mechanisms for Knowledge Management System in E-health Environment

Krzysztof Goczyła, Teresa Grabowska, Wojciech Waloszek, Michał Zawadzki
Gdańsk University of Technology, Department of Software Engineering,
ul. Gabriela Narutowicza 11/12, 80-952 Gdańsk, Poland
{kris,tegra,wowal,michawa}@eti.pg.gda.pl

Abstract. The paper presents research and development accomplishments achieved so far during work on a Knowledge Management Subsystem (KMS) for a e-health system called PIPS. The paper presents the Semantic Tools layer of KMS, and concentrates particularly on the PIPS Knowledge Inference Engine (KIE). The following issues are discussed: an analysis of freely available knowledge processors, conformance to existing Semantic Web standards, basics of an algorithm called Cartographer applied in KIE, results of application of KIE based on Cartographer in the PIPS Knowledge Base and perspectives for next stages of the project.

1 Introduction

PIPS (Personalised Information Platform for Life and Health Services) is an R&D integrated project carried out within the 6th Framework Programme of European Union, area Information Society Technologies, priority E-health. The project, whose duration spans from 2004 till 2007, is realized by a consortium consisting of 17 partners from Europe (from Poland: GUT and Atena Ltd.), Canada, and China. The main goal of the project is development of a distributed, Web-based platform that would support health care of EU citizens and would help them to keep a healthy life-style. The heart of PIPS are two activities related to development of the “intelligent” part of PIPS that consists of the Decision Support Subsystem (DSS) and the Knowledge Management Subsystem (KMS). Basics assumptions and background that are behind the both subsystem have been addressed in the previous paper [1]. To remind briefly: DSS consists of a set agents that are responsible for interaction with users (doctors, patient, or other citizens), and a database that contains, among others, information about patients (clinical records) in the form of so-called virtual egos. To fulfill the users’ demands or queries, or to react on some facts concerning patients reported by tele-medicine devices, some DSS agents (called Knowledge Discovery Agents, KDa) communicate with the PIPS Knowledge Base (KB) managed by the Knowledge Management Subsystem (KMS). In its Knowledge Base (KB), KMS stores relevant knowledge as ontologies [2], general or especially suited for PIPS.

* Work partially sponsored by the 6. Framework Programme of European Union, Contract No. IST-507019-PIPS
An essential part of KMS is the Knowledge Inference Engine (KIE) that is able to perform reasoning over the ontologies to infer new knowledge from the stored one. The knowledge is transferred from KIE to the KDAs according to a specific protocol and is further used by DSS to take appropriate decisions.

In this paper, we concentrate on the Knowledge Management Subsystem of PIPS, particularly on the Knowledge Inference Engine. The reason for that is that the authors of this paper were assigned a task of design and development of KMS that – according to the Description of Work for PIPS – could: (1) deliver trustworthy and dependable knowledge base covering broad range of knowledge on health and healthy lifestyle, (2) offer an efficient, scalable, flexible, distributed platform for managing knowledge base and ontologies, (3) be compliant with the Semantic Web emerging standards. In [1], we have formulated some problems to be solved in the PIPS KMS, like inferring new knowledge from Description Logics (DL) [3] ontologies or scalability of KMS with respect to the number of individuals stored in the KB. The rest of this paper describes how these problems have been solved in the first, prototype release of PIPS KMS. In Section 2 we present the architecture of Semantic Tools. In Section 3 Cartographer algorithm [4] for inferring over DL ontologies is briefly introduced. Section 4 concludes the paper with perspectives of new functionalities to be included into future releases of PIPS Knowledge Base.

2 Architecture of PIPS Knowledge Base

The PIPS Knowledge Base has been divided into two parts. An ontological part of the Knowledge Base holds data in a form allowing for inferences with respect to defined concepts. This part contains TBox and ABox data and is managed by Knowledge Inference Engine. A data part of the Knowledge Base (called DBox) holds numerical and string values of attributes of individuals and is managed by DBox Manager. From conceptual point of view, DBox is a part of ABox, but it has been separated from the rest of KB for efficiency reasons. Indeed, PIPS ontologies contain a lot of individuals with many numerical and string attributes that are not important for inference process but are important for KDAs as pure

![Fig. 1. Components of PIPS Semantic Tools](image-url)
data. Storing them directly in ABox of the Knowledge Base would considerably increase the size of the database and affect performance. In the current version of the system, both parts of the Knowledge Base are populated exclusively from ontologies, but in future releases they will be also populated from external data sources via Syntactical Tools layer of KMS. Basic components of Semantic Tools for the PIPS Demonstrator are presented in Figure 1. A KDA passes its query to Query Manager. Query Manager does a preliminary analysis of the query and determines the class of the query. Four classes of queries have been identified: 

**TBox class queries** are related only to terminology (i.e. to TBox exclusively). These queries are directed to Knowledge Inference Engine. An example of such query is the subsumption problem, like: "Is Pneumonia an IndustrialDisease?" (Pneumonia and IndustrialDisease are concepts). **ABox class queries** correspond to queries about individuals with respect to terminology. These queries are also directed to Knowledge Inference Engine. An example of such a query is the retrieval problem, like: “Give all instances of CertifiedDrug” (CertifiedDrug is a concept). **DBox class queries** correspond to queries about individuals with respect to their attributes. An example of such a query is “Give John Smith’s age” (John Smith is an individual and age is a numerical attribute). These queries are directed to DBox Manager. **ABox - DBox class queries** correspond to queries about values of attributes of individuals with respect to terminology. An example of such query is “Give all instances of Drug that contain at most 2.5 grams of paracetamol” (Drug is a concept and containsParacetamol is a numerical attribute). Execution of this kind of queries requires participation of both KIE and DBox Manager, because both ontological and data part of the Knowledge Base are involved. In each case, Query Planner prepares an execution plan for a query, which is then interpreted by Query Manager by issuing appropriate calls to KIE and DBox Manager. Returned results are merged by Query Integrator and passed to the calling KDA. The KBAdmin and DBAdmin modules are responsible for creation and initialization of TBox, ABox and DBox databases to store an ontology, and are able to load an ontology into appropriate components of KB.

### 3 The Algorithm

In this section we briefly introduce main ideas of Cartographer that constitutes an algorithmic basis for Knowledge Inference Engine in PIPS. Cartographer is an algorithm for processing DL ontologies and for reasoning over DL ontologies. Cartographer aims at storing in the knowledge base as many conclusions about concepts and individuals as possible. The conclusions can be quickly retrieved from the knowledge base in the process of query answering and remain valid due to the assumption that terminology (TBox) cannot be updated. By proper organization of the knowledge base the same conclusions can be applied to any number of individuals, facilitating efficient and scalable information retrieval and reducing the size of the knowledge base. The idea of Knowledge Cartography takes its name after a map of concepts. A map of concepts is basically a
description of interrelationships between concepts in a terminology. The map is created in the course of knowledge base creation, i.e. during ontology loading. A map of concepts can be graphically represented in a form similar to a Venn diagram (see Fig. 2).

Each new concept added to TBox divides the domain (denoted by ⊤) into two disjoint regions: one that contains individuals that belong to the concept, and the other that contains individuals that do not belong to the concept (the rest of the world). In that way, after defining atomic concepts A, B, and C the domain has been divided into eight disjoint regions, numbered from 1 to 8 (see Fig. 2a). Each region is assigned a corresponding unique position in a string of bits called signature. In that way, we can represent each concept by its signature with “1”-s at positions that correspond to regions that the concept includes, and with “0”-s elsewhere. Signatures of complex concepts (like D and E in Fig. 2a) are created by applying appropriate Boolean operators to signatures. Let us add two new axioms to our exemplary TBox (see Fig. 2b). The first axioms states that A and B are disjoint, which eliminates regions 3 and 4 from the map (these regions become unsatisfiable, which means that they cannot have instances). The second axiom states that C is included by B, which additionally eliminates regions 5 and 8. Finally, we obtain four regions, renumbered as from 1 to 4 in Fig. 2b. In the same way, each individual can be assigned a signature that defines its “region of confidence”, which actually represents what KB “knows” about the individual. The more “1”-s in an individual signature, the less KB “knows” about it. For instance, assume that we know that a new individual x is an instance of B. So, it is assigned the same signature as B: “0011”. At this point we do not know whether x is an instance of C. If - after some time - KB gains additional information that x is not an instance of C, its signature is changed to “0010” reflecting the fact that our knowledge about x has become more precise. Even this simple example illustrates remarkable potentials of Cartographer. Indeed, the Knowledge Cartography approach: (1) allows for efficient handling of basic Description Logics inferences: subsumption, disjointness, equivalence, and satisfiability, as well as non-standard ones, like least common subsumer or most specific concept [3]; (2)
allows for application of constructs postulated for expressive Description Logics that are useful for reasoning in real-life applications, like role chaining, role intersection, role complement, etc.; (3) offers possibility to formulate complex assertions, e.g. ¬C(x) or A ∪ B(x); (4) enables to store inferred conclusions in a relational database, which is of crucial importance for scalability of KB; (5) in a natural way supports three-valued logic (true, false, don’t know) inherent to the Open World Assumption (OWA) in knowledge-based systems. The last point needs some clarification. In databases, the Closed World Assumption (CWA) is followed. According to CWA, only those assertions (facts) are true (exist) that are stored in the database. Indeed, if we formulate any query on instances as an SQL statement, the result set will be either non-empty or empty. As a result the answer to any instance-related query will either be “true” or “false”. In contrast, according to OWA, if an assertion is not stored in the knowledge base, this assertion may be either true or false: the knowledge base simply “does not know” whether the assertion holds. Let us go back to the above example with instance x. After interpreting assertion B(x), the KB still “does not know” whether x is an instance of C or not, so the answer to the query: “Is x an instance of C?” should be “don’t know”. Cartographer is able to respond in such a way because signatures of B and C partly overlap and x can lie in any region indicated by “1”-s in the signature of B. Full consequences of applying CWA instead of OWA may be fundamental for results of reasoning and detailed discussion is outside the scope of this paper. In the current implementation, Cartographer conforms to OWA only in some classes of queries.

Table 1. Results of efficiency experiments. Hyphens denote that the activity could not be completed within 2 hours.

<table>
<thead>
<tr>
<th>Size of ABox</th>
<th>Loading time [s]</th>
<th>Query-processing time [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>400</td>
<td>1000</td>
</tr>
<tr>
<td>Jena</td>
<td>1</td>
<td>22</td>
</tr>
<tr>
<td>RACER</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Cartographer</td>
<td>43</td>
<td>122</td>
</tr>
</tbody>
</table>

The Cartographer-based KIE performance for ABox reasoning was compared with Jena 2 [5] and Racer [7] (FaCT [6] was not included in the tests because of its lack of support for ABox). Some results of the tests are presented in Table 1. The tests were performed on a PC with Celeron 2.4GHz and 256MB RAM. The back-end of KIE was a PostgreSQL v7 database. The main difference between analyzed reasoning algorithms is related to time of loading an ontology (TBox and ABox). The time of loading ontology is longer for Cartographer. In return a very short time of response is obtained. While Racer was unable to answer a query after 1000 individuals have been loaded to KB, our KIE could process the same query for 11000 individuals in 1.4 second. The results seem to be important for real-life applications, like PIPS, where changes in terminologies are infrequent and response time is much more critical for the overall performance of the system than ontology loading time.
4 Summary

Capabilities of PIPS Knowledge Inference Engine and Knowledge Base have been verified and validated against first versions of PIPS ontologies concerning the problem of diabetes. They contain concepts of Person, Food, Product, ClinicalRecord and Diabetes, and their related subconcepts, axioms and assertions. They together form an integrated (via import mechanism) ontology of approx. 1000 concepts, 1000 assertions and 4000 axioms. The ontologies have been used for the first release of PIPS system called Demonstrator. Experiments performed with Demonstrator showed that the Cartographer-based KIE fully meets DSS expectations. The PIPS ontology can be loaded within several minutes, which means that the forward-chaining of Cartographer is performed efficiently and the limitation that TBox cannot be modified on-line is not severe. Asks issued by KDAs can be processed in short time (fractions of a second to several seconds, depending on kind of query), and capability to respond to an ABox query does not depend on the size of ABox. These results encourage us to further enhance functionality of the engine. Directions of enhancement that are both interesting from research point of view and useful for PIPS system purposes include non-monotonic reasoning, multi-ontology queries [8] and trust issues [9].

Cartographer Approach to knowledge representation will be also further developed towards support of expressive DL constructs, particularly in querying, and towards uniformity in handling roles and concepts.

References