Argument Exchange Over the Semantic Web*

Alejandro G. Stankevicius

Laboratorio de Investigación y Desarrollo en Inteligencia Artificial
Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Bahía Blanca - Buenos Aires - ARGENTINA
e-mail: ags@cs.uns.edu.ar

ABSTRACT

The *Semantic Web* purports to give computer-accessible meaning to the content of the World Wide Web. The rationale behind this project is to create an environment where all this information could be freely exchanged among diverse entities, yet retaining its intended meaning.

Defeasible Logic Programming is a knowledge representation and reasoning formalism that by combining Logic Programming with Defeasible Argumentation is able to represent incomplete and potentially contradictory information.

In this article we explore what is required in order to provide Defeasible Logic Programming knowledge representation and reasoning over the semantic web.

Keywords: semantic web, knowledge representation, defeasible reasoning, argumentation, ontologies.

1 MOTIVATIONS

The *Semantic Web* purports to give computer-accessible meaning to the content of the World Wide Web [1]. The rationale behind this project is to create an environment where all this information could be freely exchanged among diverse entities, yet retaining its intended meaning. Even though humans have no trouble shopping online, consulting diverse sources of information, or contacting other persons through the web, computers can hardly accomplish any of these tasks. Aware of these shortcomings, Tim Berners-Lee (father of the web and head of the World Wide Web Consortium), suggest the progressive adoption of a set of standards to achieve this goal. The central idea of this approach is to enrich the current *syntactic* web with the missing semantics.

This ambitious task is being tackled in successive stages [2]. The first consist in making explicit the structural information: web developers nowadays tend to focus mainly in how their content will look, neglecting other aspects such as where the data came from or how its parts relate to each other. HTML, the standard in which most web pages are coded, is a markup language proving a set of tags for describing the content of documents with an emphasis on visual presentation. As such, it may not be the best language to capture those missing features. XML, in contrast, is a new markup language that provides a text-based mean to describe them. In fact, we should rather call XML a *meta-markup language*, since we can actually define our own set of tags. This freedom turned XML into a success story, making it the *de facto* standard for sharing structured information, particularly among heterogeneous systems such as those present on internet. In the context of the semantic web, XML provide a medium in which web developers can express information about the information they are providing. This *metadata*, usually available before publishing a given piece of information, is unfortunately discarded when that information gets delivered as an HTML document. For instance, most dynamic web content is generated through some sort of scripting language accessing a repository, say in the form of a database, but all the structuring information one can infer from it is lost once that data reaches the end user.

Then again, having this metadata along the actual data is not enough: the presence of structuring information tells us nothing about its actual meaning. In a sense, we also need to convey how the different parts of the data relate with each other. This constitute the second stage to bring about the semantic web. To this end, several languages have been proposed, such as RDF, and RDF Scheme. These languages indeed allow the knowledge engineer to express relations among data parts. Once we start describing the meaning of our information in such a precise way, we end up characterizing what is called an *ontology*. Briefly stated, an ontology depicts one of the pos-

*Partially supported by Agencia Nacional de Promoción Científica y Tecnológica (PAV 2003 Nro. 076, PICT 2002 Nro. 13096, PICT 2003 Nro. 15043), CONICET (*Consejo Nacional de Investigaciones Científicas y Técnicas de la República Argentina*), and CIC (*Comisión de Investigaciones Científicas de la Provincia de Buenos Aires*).

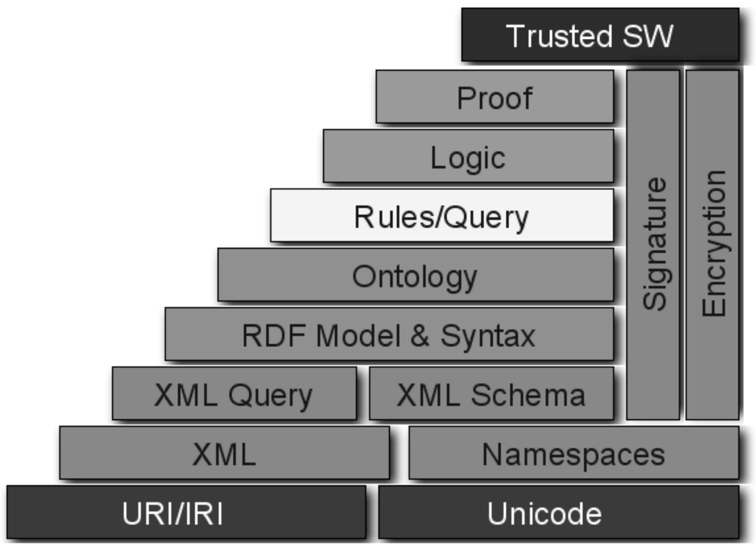

Figure 1: Layers of the Semantic Web

sible views upon a given domain, enumerating the object classes, as well as its individual instances. RDF and RDF Scheme serve this purpose quite well, yet some more abstract properties still cannot be expressed with them. OWL(*Web Ontology Language*), the current standard for defining ontologies, was later introduced to address those limitations.

The third stage in the adoption of the semantic web encompasses putting all this data and its corresponding metadata to a good use. Since one of the goals was to make sure all the content of the web remained computer-accessible, once the semantic web gets underway, this huge knowledge repository suddenly becomes available to any reasoner. For instance, software agents will be able to harvest relevant information from the web, solve complex tasks delegated upon them, etc. To put all these in perspective, the interaction among the different layers that will give birth to the semantic web is briefly sketched in Figure 1. Note that it implies we are going to need powerful formalisms for Knowledge Representation and Reasoning (KR&R, for short) capable of handling both data and metadata. These formalisms constitute the *logic layer*, and their inference engines are the core of the *proof layer*. However, given the bottom-up approach being followed in the implementation of the semantic web, no single theory for KR&R can be deemed today as the standard for representing knowledge.

In this article, it is our objective to lay the foundations required to offer *Defeasible Logic Programming* (DeLP) as a KR&R alternative within the semantic web. In this first installment, we focus mainly on the infrastructure required for exchanging DeLP arguments over the semantic web. To that end, the next section briefly introduce DeLP's key concepts, and then section 3 addresses the actual requirements that are to be met in this first stage. Finally, section 4 presents our conclusions.

2 DEFEASIBLE LOGIC PROGRAMMING

Defeasible Logic Programming (DeLP) is a formalism that by combining Logic Programming with Defeasible Argumentation is able to represent incomplete and potentially contradictory information. Ideas borrowed from Defeasible Argumentation [5] such as representing defeasible reasons as arguments or performing a full dialectical analysis before answering queries are carefully added to a knowledge representation language featuring Prolog-like rules. This section briefly introduces DeLP's essentials following its most recent formulation [4], referring the reader looking for a more comprehensive presentation to [3].

The DeLP language is defined in terms of three disjoint sets: *facts*, *strict rules*, and *defeasible rules*. Literals can be ground atoms (*e.g.*, A), or their strong negation (*e.g.*, $\sim A$). Facts are simply literals. Strict rules are ordered pairs $L_0 \leftarrow L_1, \ldots, L_n$ whose first component, L_0, is a literal, and whose second component, $L_1, \ldots, L_n$, is a finite non-empty set of literals. In a like manner, a defeasible rule is an ordered pair $L_0 \prec L_1, \ldots, L_n$ whose first component, L_0, is a literal, and whose second component, $L_1, \ldots, L_n$, is a finite non-empty set of literals. Strict rules represent undisputed information while, in contrast, defeasible rules represent tentative information.

In this formalism, the state of the world is modelled as a *Defeasible Logic Program* (*de.l.p*), essentially a possibly infinite set of facts, strict rules and defeasible rules. In a given *de.l.p* $\mathcal{P}$, the subset of facts and strict rules is referred as Π, and the subset of defeasible rules as Δ. When required, the *de.l.p* $\mathcal{P}$ can also be noted as (Π, Δ). Since the set Π represent non-defeasible information, it is assumed that it is non-contradictory, that is, no pair of complementary literals can be derived at the same time. As usual, literals that may be derived are obtained chaining as many rules and facts as required.

Even though Π must be non-contradictory, Δ, and hence $\mathcal{P}$ itself (*i.e.*, $\Pi \cup \Delta$), may be contradictory. However, out of the literals that can be defeasible derived in a given *de.l.p*, only those able to stand the dialectical analysis are entailed. This procedure is borrowed from the field of Defeasible Argumentation, where people speak in terms *arguments* instead of derivations. An argument is a tentative piece of reasoning supporting a given conclusion, that satisfy some restrictions such as minimality, consistency, etc. An argument structure $\langle \mathcal{A}, h \rangle$ denotes that $\mathcal{A}$ is an argument for h.

This framework allows the construction of argument structures for conflicting conclusions. In DeLP, this conflict is settled defining under which conditions one argument structure has enough strength to *warrant* its conclusion. In short, a given literal is warranted if we are able to find an argument structure for that literal that remains undefeated. Unfortunately, the precise conditions under which an argument structure defeats another argument structure is beyond the scope of this short article. Suffice it to say that this relation must break the attack cycle existing between conflicting argument structures. Now, since defeaters are in turn argument structures, there may exists defeaters for the defeaters, and so on. This sequence of argument structures, each one defeating the previous one, is called in this context *argumentation line*, in the sense that this exchange of reasons seems to be exploring a given topic of the controversy. Not withstanding, not every exchange of arguments actually constitutes a valid pattern of reasoning. For instance, circular argumentation is a particular case of *fallacious reasoning* which should be avoided at all cost (since it compromises termination). The occurrence of cycles and other undesired situations are prevented imposing a set of conditions over the potential argumentation lines. Those argumentation lines not incurring in any sort of fallacious reasoning are called *acceptable*.

In order to determine whether a given literal is warranted, all the acceptable argumentation lines that have that literal as their origin must be considered. This dialectical analysis is usually structured as a tree, called *dialectical tree*. Finally, even though the notion of warrant characterizes a set of literals that could be interpreted as the semantic of a given *de.l.p*, this formalism takes into account more outcomes as answers to a given query h:

- YES, when h is warranted.
- NO, when $\overline{h}$ is warranted, where $\overline{h}$ denotes the complement of h with respect to strong negation.
- UNDECIDED, when neither h nor $\overline{h}$ are warranted.
- UNKNOWN, when h is not present in the signature of the *de.l.p* under consideration.

After this succinct overview of DeLP, the next section outlines the requirements to be fulfilled in order to have access to DeLP KR&R over the semantic web.

3 AN ARGUMENT ONTOLOGY

Providing DeLP KR&R over the semantic web is an ambitious goal. This complex task should be approached in successive phases, such as:

1. Make explicit all the metadata surrounding DeLP programs, queries, answers, argument structures and dialectical trees.
2. Engineer an ontology suitable for characterizing the aforementioned concepts.
3. Adapt DeLP to serve as a knowledge representation alternative within the logic layer of the semantic web.
4. Embed DeLP inference engine into the proof layer of the semantic web.

In what follows, we consider each of these phases in detail.

To begin with, making explicit the metainformation present in a DeLP program is somewhat straight. In fact, given its highly regular syntax, DeLP programs can be easily parsed to elicit rules, facts and literals out of plain text. Granted, why rebuild all that metadata when you can instead put it there right from the beginning. For instance, nowadays we exchange DeLP programs over internet as plain text files, but we could rather encode them using a custom set of XML tags:

Example 1 $fly(X) \prec bird(X)$ can be codified in XML as:

```
<defeasible-rule>
  <head>
    <positive-literal>
      fly(X)
    </positive-literal>
  </head>
  <body>
    <positive-literal>
      bird(X)
    </positive-literal>
  </body>
</defeasible-rule>
```

Regarding the second phase, we already mentioned that making explicit the structuring infomation is not enough. We also need an ontology capable of capturing the subtle interaction among parts of DeLP programs, such as argument structures, relashionships among them (*e.g.*, defeat), and warranted literals. Languages such as RDF and RDF Scheme are suitable for modelling the highly regular aspects of DeLP syntax (for instance, that strict and defeasible rules are both rules). However, some features only available to OWL (such as cardinality restrictions or disjointness of classes), may also come in handy when characterizing other concepts (*e.g.*, dialectical trees). We are in the process of refining the documents formally defining an ontology that meets all these requirements.

The third phase appears to be more challenging than initially expected. Note that most of the standards being sanctioned by the W3C closely match the point of view of the Description Logic community when it comes to represent knowledge and reason about it. In fact, one of the OWL dialects, OWL-DL, is directly equivalent to a well known description logic. DeLP, in contrast, follows a more classical approach, unfortunately not entirely compatible. This accounts for the difficulties faced when trying to incorporate case-based reasoning to the semantic web (or any rule-based knowledge representation for that matter). As it has been discussed elsewhere [1], some aspect impossible to capture under one approach are easily modelled under the other, and vice-versa. We believe, in turn, that both approaches can coexists in harmony: lets keep the stack of layers we already have (and their corresponding languages), and use those modelling tools to express DeLP programs, argument structures, dialectical trees, etc. The ontology under development constitutes the first step in this direction.

Finally, we have to admit that the fourth phase still looks somewhat blurry. Recall that the adoption of the semantic web is following a bottom-up approach, so even though the technologies associated with the lower layers are well known standards, those associated with the higher layers are still under development. That is to say, it is still unclear how one should make a given inference engine available within the semantic web. We are inclined to believe that web services provide a suitable framework for delivering such a service, yet the W3C may think otherwise. For the time being, our research laboratory has decided to advertise a DeLP inference engine as a web service. This allows developers to easily add DeLP KR&R to their software agents or web services.

4 SUMMARY

Tim Berners-Lee put forth an huge challenge called Semantic Web. If only a part of what has been promised ever gets delivered, that would be enough to radically change the way we interact with internet. Naturally, time will tell, but in the meantime, we can start profitting from the initial work in this regard. For instance, we have decided to provide DeLP knowledge representation and reasoning to agent developers or programmers in general. To do so, our research laboratory has implemented a prototype of a web service, which is available at `http://lidia.cs.uns.edu.ar`. Throughout this article we have also discussed what we need to make this service available within the semantic web, identifying and briefly discussing the required phases to do so. From that analysis, we conclude that the first two phases can be addressed right now, unlike the last two, where we should wait until the involved parties (*i.e.*, the W3C members), manage to agree on the standards to be used.

References

[1] Antoniu, G., and van Harmelen, F. *A Semantic Web Primer.* The MIT Press, 2004.

[2] Berners-Lee, T., Hendler, J., and Lassila, O. The semantic web. *Scientific American 284*, 55 (2001), 28–37.

[3] García, A. J. *Programación en Lógica Rebatible: Lenguaje, Semántica Operacional, y Paralelismo.* PhD thesis, Departamento de Ciencias de la Computacin, Universidad Nacional del Sur, Bahía Blanca, Argentina, Dec. 2000.

[4] García, A. J., and Simari, G. R. Defeasible Logic Programming: An Argumentative Approach. *Theory and Practice of Logic Programming 4*, 1 (2004), 95–138.

[5] Simari, G. R., and Loui, R. P. A Mathematical Treatment of Defeasible Reasoning and its Implementation. *Artificial Intelligence 53*, 1–2 (1992), 125–157.