Towards Reasoning Pragmatics

Pascal Hitzler

Kno.e.sis Center, Wright State University, Dayton, Ohio
http://www.pascal-hitzler.de/

Abstract. The realization of Semantic Web reasoning is central to substantiating the Semantic Web vision. However, current mainstream research on this topic faces serious challenges, which force us to question established lines of research and to rethink the underlying approaches.

1 What Is Semantic Web Reasoning?

The ability to combine data, mediated by metadata, in order to derive knowledge which is only implicitly present, is central to the Semantic Web idea. This process of accessing implicit knowledge is commonly called reasoning, and formal model-theoretic semantics tells us exactly what knowledge is implicit in the data.

Let us attempt to define reasoning in rather general terms: Reasoning is about arriving at the exact answer(s) to a given query. Formulated in this generality, this encompasses many situations which would classically not be considered reasoning – but it will suffice for our purposes. Note that the definition implicitly assumes that there is an exact answer. In a reasoning context, such an exact answer would normally be defined by a model-theoretic semantics.

Current approaches to Semantic Web reasoning, however, which are mainly based on calculi drawn from predicate logic proof theory, face several serious obstacles.

- Scalability of algorithms and systems has been improving drastically, but systems are still incapable of dealing with amounts of data on the order of magnitude as can be expected on the World Wide Web. This is aggravated by the fact that classical proof theory does not readily allow for parallelization, and that the amount of data present on the web increases with a similar growth rate as the efficiency of hardware.
- Realistic data, in particular on the web, is generally noisy. Established proof-theoretic approaches (even those including uncertainty or probabilistic methods) are unable to cope with this kind of data in a manner which is ready for large-scale applications.

1 It is rather peculiar that a considerable proportion of so-called Semantic Web research and publications ignores formal semantics. Even most textbooks fail to explain it properly. An exception is [7].

2 Simply referring to a formal semantics is too vague, since this would also include procedural semantics, i.e. non-declarative approaches, and thus would include most mainstream programming languages.
It is a huge engineering effort to create web data and ontologies which are of sufficiently high quality for current reasoning approaches, and usually beyond the abilities of application developers. The resulting knowledge bases are furthermore severely limited in terms of reusability for other application contexts.

The state of the art shows no indications that approaches based on logical proof theory would overcome these obstacles anytime soon in such a way that large-scale applications on the web can be realized. Since reasoning is central to the Semantic Web vision, we are forced to rethink our traditional methods, and should be prepared to tread new paths.

A key idea to this effect, voiced by several researchers (see e.g. [3,23]) is to explore alternative methods for reasoning. These may still be based more or less closely on proof-theoretic considerations, or they may not. They could, e.g., utilize methods from statistical machine learning or from nature-inspired computing.

Researchers who are used to thinking in classical proof-theoretic terms are likely to object to this thought, arguing that a relaxation of strict proof-theoretic requirements on algorithms, such as soundness and completeness, would pave the way for arbitrary algorithms which do not perform logical reasoning at all, and thus would fail to adhere to the specification provided by the formal semantics underlying the data – and thus jeopardize the Semantic Web vision. While such arguments have some virtue, it needs to be stressed that the nature of the underlying algorithm is, effectively, unimportant, as long as the system adheres to the specification, i.e. to the formal semantics.

Imagine, as a thought experiment, a black box system which performs sound and complete reasoning in all application settings it is made for – or at least up to the extent to which standard reasoning systems are sound and complete. Does it matter then whether the underlying algorithm is provably sound and complete? I guess not. The only important thing is that its performance is sound and complete. If the black box were orders of magnitude faster than conventional reasoners, but somebody would tell you that it is based on statistical methods, which one would you choose to work with? Obviously, the answer depends on the application scenario – if you’d like to manage a bank account, you may want to stick with the proof-theoretic approach since you can prove formally that the algorithm does what it should; but if you use the algorithm for web search, the quicker algorithm might be the better choice. Also, your choice will likely depend on the evidence given as to the correctness of the black box algorithm in application settings.

This last thought is important: If a reasoning system is not based on proof theory, then there must be a quality measure for the system, i.e., the system must

---

3 Usually, they are not sound and complete, although they are based on underlying algorithms which are, theoretically, sound and complete. Incompleteness comes from the fact that resources, including time, are limited. Unsoundness comes from bugs in the system.