Logic, Accountability and Design: extended abstract*

Pedro Cabalar[1] and David Pearce[2]

[1] University of Corunna, Spain, cabalar@udc.es,
[2] Universidad Politécnica de Madrid, Spain, david.pearce@upm.es

1 Introduction

This note is a contribution to the methodology of applied, computational logics in light of their potential role in securing the accountability of Artificial Intelligence (AI) systems. A key feature of the idea of accountability is that solutions, actions and decisions made by intelligent systems should ultimately be explainable to the end user in a comprehensible manner. In view of this, explainable AI has recently become a hot topic of research. Much of symbolic AI is supported by logic-based systems whose reasoning mechanisms are, or should be, transparent and comprehensible. But is it really the case that a logic-based system can provide convincing explanations accessible to the non-expert? In practice this is doubtful as such systems may contain many lines of code and numerous computational reasoning steps. Even the expert user or developer may not be able to survey and assimilate the entire reasoning process for a given outcome.

This point has been recognised for quite some time. Already [5] contains an extensive survey of approaches to adding explanations or justifications to answer set programs. Recently the XLoKR workshop series on Explainable Logic-Based Knowledge Representation has featured systems such as ASP, description logics, default logics, argumentation theory and more. However until now attention has mainly focused on how to add human-understandable explanations to the reasoning steps computed by a primary logic-based systems. While these works are valuable, for the most part they implicitly take for granted the adequacy of the primary reasoning formalism that they aim to extend.

If logic is to play a significant role in making AI systems explainable, then logic itself needs to be accountable. Many logics applied in AI systems are in competition with one another. Since logics are an integral part of our approaches to knowledge representation and reasoning, they can no longer be considered as a kind of 'theory-neutral' component, as logic was often treated by philosophers of science in the past. In light of this we should ask ourselves, what kinds of adequacy conditions should applied logics fulfil in order to be candidates to support accountable and ultimately trustworthy AI. In doing so

* Supported by project LIANDA - BBVA Foundation Grants for Scientific Research Projects; additional support from the Spanish Ministry of Science and Innovation, Spain, MCIN/AEI/10.13039/501100011033 (grant PID2020-116201GB-I00), and by Xunta de Galicia, Spain and the European Union (grant GPC ED431B 2022/33)

we hope to dispel the idea that *anything goes*, ie that the engineer has a free hand just to pick off the shelf any reasoning mechanism that appeals to her. Instead we argue for a principled approach to designing applied logics and formal reasoning tools for uptake in AI; an approach that complies with suitable adequacy conditions and respects a sound methodology. One consequence will be to attempt to rule out *ad hoc* solutions to formal reasoning problems.[1]

2 Conceptual analysis and explication

The formal analysis of concepts was a major component of the work of logical empiricist philosophers in the 20th Century, and Rudolf Carnap was perhaps in this respect its most illustrious representative. Carnap's method of *explication* is probably the clearest account of how logic and formal methods should be applied to the rational reconstruction of scientific concepts. Carnap's method is most clearly articulated in the introduction to his *Logical Foundations of Probability* [2]. As he explains, the method

> consists in transforming a more or less inexact concept into an exact one or, rather, in replacing the first by the second. We call the given concept the **explicandum**, and the exact concept proposed to take the place of the first the **explicatum**. The explicandum may belong to everyday language or to a previous stage in the development of scientific language. The explicatum must be given by explicit rules for its use, for example, by a definition which incorporates it into a well-constructed system of scientific either logico-mathematical or empirical concepts. [2]

Part of the task consists in specifying adequacy conditions that the explicatum should satisfy: a pre-formal analysis of the explicandum may suggest a series of properties desirable for the explicatum.

> If a concept is given as explicandum, the task consists in finding another concept as its explicatum which fulfils the following requirements to a sufficient degree.
>
> 1. The explicatum is to be *similar to the explicandum* in such a way that, in most cases in which the explicandum has so far been used, the explicatum can be used; however, close similarity is not required, and considerable differences are permitted.
> 2. The characterization of the explicatum, that is, the rules of its use (for instance in the form of a definition), is to be given in an *exact* form, so as to introduce the explicatum into a well-connected system of scientific concepts.
> 3. The explicatum is to be a *fruitful* concept, that is, useful for the formulation of many universal statements (empirical laws in the case of a nonlogical concept, logical theorems in the case of a logical concept).
> 4. The explicatum should be as *simple* as possible; this means as simple as the more important requirements (1), (2), and (3) permit. [2]

Ultimately the question whether the explicatum is or is not correct is not a factual one, but a question of methodological adequacy [2].

Although logic has formed a prominent part of the design of intelligent systems, most developers of logic-based systems have shied away from specifying

[1] The full version of the paper elaborates more on this issue and includes a case study of the adequacy of logics underlying ASP. In this abridged version, we focus just on adequacy conditions for logics in KR.

a clear methodology of the type that Carnap has proposed. One exception is the programme proposed by Michael Gelfond, a founder of answer set programming and a leading contributor to logic-based AI. The aim of his programme is to reconstruct some of the most basic forms of human knowledge and to exploit this knowledge for practical problem solving. It combines scientific and engineering knowledge of real systems with practical human skills and abilities and commonsense reasoning. It deals with both static and dynamic domains. Gelfond's programme combines the physicalist language of engineering and physical systems with epistemic notions such as belief, agency and action. The new programme of rational reconstruction is much less self-conscious than its predecessor and is less well known. Nevertheless it has clear goals and methodology, even if they are sometimes buried in technical articles and lectures.

Gelfond's programme for KR has two main objectives [6]. First, achieving an understanding of "basic commonsense notions we use to think about the world: beliefs, knowledge, defaults, causality, intentions, probability, etc., and to learn how one ought to reason about them." Secondly it aims "to understand how to build software components of agents – entities which observe and act upon an environment and direct its activity towards achieving goals."([6])

These goals shape the criteria used to evaluate and select languages for KR. In particular, Gelfond [6] endorses four main adequacy criteria: clarity, elegance, expressiveness and relevance. These are further elaborated in [7]:

- *Naturalesness*: Constructs of a formal language L should be close to formal constructs used in the parts of natural language that L is designed to formalize. The language should come with a methodology of using these constructs for knowledge representation and programming.
- *Clarity*: The language should have simple syntax and clear intuitive semantics based on understandable informal principles.
- *Mathematical Elegance*: Formal description of syntax and semantics of the language should be mathematically elegant. Moreover, the language should come with mathematical theory facilitating its use for knowledge representation and programming.
- *Stability*: Informally equivalent transformations of a text should correspond to formally equivalent ones.
- *Elaboration Tolerance*: It should be possible to expand a language by new relevant constructs without substantial changes in its syntax and semantics.

Gelfond's first criterion is close to Carnap's first condition of *similarity*, while his second criterion echoes Carnaps´s fourth condition of *simplicity*. Gelfond's third criterion is close to Carnap's second requirement of *exactness*, while elaboration tolerance and Gelfond's other criterion of relevance (from [6]) clearly relate to Carnap's requirements 2 and 3.

One condition that Gelfond does not entertain is the requirement of *efficiency* understood in a computational sense. Efficiency is evidently an aim in designing computational systems and a requirement of any KR language is that it can eventually be processed by a computer. But Gelfond's methodology suggests that conceptual adequacy should initially at least take preference over computational efficiency.

A similar point has been recently made by Jones, Artikis and Pitt in their proposed methodology for the design of socio-technical systems [11]. They are concerned with the way in which social concepts are reconstructed and represented in computational, socio-technical systems. [11] deals mainly with social concepts such as *trust*, *role* and *normative power*. But their reconstruction in computational systems will typically involve a strongly logical component. [11] proposes a multi-stage process of representing and implementing these concepts. The first stage involves theory construction, passing from some observed social phenomena S to pre-formal representations. Then, Step2-Phase1 representations provide an analysis of conceptual structure "constrained primarily by considerations of expressive capacity, not those of computational tractability"[11] Later we come to the stage of implementation where simplifications may have to be made to achieve comptational tractability.. For [11], a primary requirement for assessing the adequacy of a conceptual characterisation is expressive capacity. As criteria, they list the capacity to (i) identify the principle elements; (ii) test for consistency; (iii) articulate specific, characteristic aspects of the concept; (iv) 'place' the concept in relation to its near relatives. (iv) is clearly related to Carnap's second and third requirements, and also to Gelfond's criteria of *naturalness* and *relevance*.

These three approaches to the formal analysis of concepts come from very different backgrounds and yet display important commonalities. They each urge a principled approach to formal reconstructions, based on a clear methodology. They propose a preliminary, informal analysis of concepts, preferably informed by scientific or philosophical reflection, suggesting that this may lead to specifying criteria of adequacy that the formal concepts should satisfy. Then there is the shared idea that the formal characterisations fit into the broader scheme of scientific concepts covering related domains. Lastly, there is the idea of fruitfulness or relevance for problem solving, as well as the aim of expressive capacity. These are all considerations that we may bring to bear on the study of possible conditions for the adequacy of logical systems in AI. We will focus on applied logics for KR, trying to extract some formal adequacy conditions.

3 Nonmonotonic reasoning and strong equivalence

Logic-based systems for KR in AI are typically *nonmonotonic* in character, to allow for the representation of defaults and to be able to express exceptions to general rules. Since these systems depart considerably from ordinary, bread and butter logics, classical or otherwise, it is not immediately obvious that they fulfil the needs of explainability and accountability for AI in practical cases. Can all such systems really be considered logics? Do they lend themselves to support explainable AI? How can we choose between rival solutions to specific kinds of reasoning?

One way to approach these questions is by way of some concepts that were studied already in the early years of nonmonotonic reasoning (NMR). In particular, to ask what constitutes a (monotonic) logical basis for an NMR sys-

tem. If our logic for KR, despite nonmonotonicity, is clearly anchored to a standard, monotonic logic, this may help to clarify and even legitimate its reasoning mechanism. This suggests that we might focus initially on how an NMR system extends and relates to a given, underlying monotonic logic. To consider what it means saying that a logic L forms a well-behaved monotonic basis for a given nonmonotonic consequence relation, three main conditions come to light.[2]

Definition 1. *Let C be a (possibly nonmonotonic) consequence relation and let C_L be the consequence relation for a monotonic logic L. We say that L forms a* deductive base *for C if the following conditions hold:*

$$\textit{Sublogic}: \quad C_L \leq C \quad (\textit{ie} \quad C_L(\Gamma) \subseteq C(\Gamma) \quad \textit{for all} \quad \Gamma) \tag{1}$$

$$\textit{Left absorption}: \quad C_L C = C \tag{2}$$

$$\textit{right absorption}: \quad C C_L = C \tag{3}$$

Absorption guarantees that if theories are equivalent in L they remain equivalent at the nonmonotonic level (ie under C). Moreover closing the nonmonotonic consequences of a theory under L-consequence does not produce anything new. One characteristic of standard, monotonic logics is the presence of replacement theorems that guarantee when equivalent formulas or theories are interchangeable *salva veritate* in any context. In nonmonotonic logics, replacement properties are more complex, since equivalence may also be derived from the absence of information. For instance, two theories Π_1 and Π_2 may yield the same C-consequences but these may differ after adding new information Γ. This motivates:

Definition 2. *In the context of a nonmomontonic consequence relation C, two theories Π_1 and Π_2 are said to be* strongly equivalent *if for all Γ, $C(\Pi_1 \cup \Gamma) = C(\Pi_2 \cup \Gamma)$*

In other words, Π_1 and Π_2 remain equivalent in whatever context Γ they are embedded. One property of deductive bases is immediate but very powerful: if L is a deductive base for C, then L-equivalence of Π_1 and Π_2 is a sufficient condition for strong equivalence. A given nonmonotonic relation C may have several monotonic deductive bases and equivalence in any of them is a sufficient condition for strong equivalence, but not always a necessary condition. To guarantee a suitable replacement property one can add a further refinement and say that a deductive base is *strong* if it satisfies:

$$C_L(\Pi_1) \neq C_L(\Pi_2) \Rightarrow \quad \text{there exists} \quad \Gamma \quad \text{such that} \quad C(\Pi_1 \cup \Gamma) \neq C(\Pi_2 \cup \Gamma).$$

If a deductive base is strong we obtain what is known as a strong equivalence theorem, namely that two theories are interchangeable in any context under the nonmonotonic inference *if an only if* they are equivalent in the monotonic base.[3] This has important consequences for simplifying nonmonotonic theories

[2] The term deductive base defined below is taken from [3,4]; however similar ideas can be found in [12] and elsewhere.

[3] Equivalence usually means same intended models; but if consequence is defined in terms of intended models, then this will imply equivalence wrt C.

and programs and studying their properties. For example if the base logic has a suitable proof theory we can use it to test for program equivalences.

4 Methodologies for applied logics

Given the discussion above, let us try to compile some guidelines for the design and development of logic-based systems for KR and AI. We propose an initial list of criteria that we group in the following three kinds of conditions.

Type I. General requirements for good design and sound methodology

1. Is it logic?
2. Is the reasoning based on a known underlying logic?
3. Is it a combination of known logics?

The first condition may at first sight appear to be circular. But if understood correctly, it does make sense. In KR there are formal reasoning methods that appear to be logic-based yet fail some natural properties that one would expect to hold. Computational logics have vastly extended the boundaries of what the repertoire of logic, in its mathematical paradigm, was formerly supposed to include. Nevertheless, some properties seem to be constitutive and basic to logic, especially if we consider the KR context. We sometimes find formalisms defined only for syntactic fragments or under syntax restrictions and with *ad hoc* semantic definitions that do not rely on any standard method for defining a logic. So we keep Condition 1 as an imprecise but useful first test of adequacy.

Condition 2 is inspired by the discussion of the previous section. If our system is based on a known logic whose reasoning mechanisms are well understood and appropriate for the domain, we have advanced on the path to verifying its adequacy. Condition 3 is related to Carnap's third and Gelfond's fifth requirements. Much work has been done on combining logics, and in KR many opportunities arise for their application. One may think of combinations such as knowledge with belief, tense and modality, space and time, nonmonotonicity combined with epistemic reasoning, and others. A primary logic that can be combined with other logics to gain new functionalities can be a very fruitful conceptual tool. Criteria for combining logics are also important. For instance, a combined formalism should have a clear connection to its constituent logics – for instance, a modal extension of ASP should collapse to ASP when no modal operators are used. Also, the combined formalism should inherit and generalise recognisable properties from its constituent logics.

Type II. Specific adequacy conditions for the logical concepts to be formalised
These include adequacy conditions that pertain to a specific concept and context, perhaps based on a pre-formal analysis of the concept.

1. Does it adequately reconstruct/formalise the intended concepts?
2. Does it offer suitable reasoning mechanisms for those concepts?

3. Does it accommodate new cases in a clear and natural manner?
4. Does it possess desirable metatheoretic properties?

The first of these may include for instance the expressive capacity of the formal language and be based on prior analysis of the concept. The second condition relates to semantics and inference. The third requirement relates to Carnap's and Gelfond's ideas of simplicity and clarity. A successful formalisation should yield a general approach that goes beyond just a few isolated cases of reasoning. Moreover, it should handle new examples in a natural manner without needing *ad hoc* adjustments and revisions. 4 may include matters of tractability of reasoning, properties that are generally regarded as 'good' for a logic, or properties that are desirable in a given KR context. These will tend to change from domain to domain and so can best be made precise given a specific context.

Type III. Methods of reasoning that may lead to explainable AI and support the rational acceptability of conclusions Lastly we may consider requirements that will allow for reasoning steps to be displayed in a way that can explain the outcome of a logic-based, computational system in practical cases.

1. Can it be combined with methods of explanation?
2. Can explanations be broken down into simple steps for human comprehension and rational acceptance?

This is currently an very active field of inquiry. It may involve the ability to apply a secondary type of logic that can add justification steps, or perhaps argumentation trees, to computations carried out in the primary logical system. Such methods should be convincing to a rational agent and, if possible, graspable by a human user.

5 Related and Future Work and Conclusions

We have argued that for logic to play a key role in making AI systems more accountable, we also have to analyse and question the adequacy of the primary reasoning system itself. Here there is much territory still to be explored. As examples of works that *have* initiated a critical analysis and discussion of the adequacy of logical systems in AI, we can mention [8,10,9], focused on epistemic reasoning and multi-agent systems in particular.

We have tried to raise awareness of the need to to take a principled approach to the design of logical systems, to reject the assumption that *anything goes* when proposing a new reasoning formalism, and to avoid *ad hoc* solutions designed to 'save the phenomena'. We have also looked at some general requirements for the formal reconstruction of concepts and proposed some preliminary desiderata for logics to be applied in AI systems.

There is much more to be done. This is the first of a three-part work in progress. The second part will treat extensions of logic programming, such as those dealing with aggregates, temporal logic or epistemic reasoning. The third part will extend work already started on explanatory ASP [1].

References

1. Cabalar, P., Fandinno, J., Muñiz, B.: A system for explainable answer set programming. In: Ricca, F., et al (eds.) Proceedings 36th International Conference on Logic Programming (Technical Communications), ICLP Technical Communications 2020, UNICAL, Rende (CS), Italy, 18-24th September 2020. EPTCS, vol. 325, pp. 124–136 (2020)
2. Carnap, R.: Logical Foundations of Probability. University of Chicago Press (1950)
3. Dietrich, J.: Deductive bases of nonmonotonic inference operations. Ntz report, University of Leipzig (1994)
4. Dietrich, J.: Inferenzframes, University of Leipzig. Ph.D. thesis (1995)
5. Fandinno, J., Schulz, C.: Answering the "why" in answer set programming - A survey of explanation approaches. Theory Pract. Log. Program. **19**(2), 114–203 (2019). https://doi.org/10.1017/S1471068418000534, `https://doi.org/10.1017/S1471068418000534`
6. Gelfond, M.: Personal perspective on the development of logic programming based KR languages (2011), unpublished draft, available online at `http://www.depts.ttu.edu/cs/research/krlab/papers.php`
7. Gelfond, M., Zhang, Y.: Vicious circle principle and logic programs with aggregates. Theory and Practice of Logic Programming **14**(4-5), 587–601 (2014)
8. Herzig, A.: Logics of knowledge and action: critical analysis and challenges. Autonomous Agents and Multi-Agent Systems **29**, 719–753 (2014)
9. Herzig, A.: Dynamic epistemic logics: promises, problems, shortcomings, and perspectives. Journal of Applied Non-Classical Logics **27**, 1–14 (2018)
10. Herzig, A., Lorini, E., Perrussel, L., Xiao, Z.: Bdi logics for bdi architectures: Old problems, new perspectives. KI - Künstliche Intelligenz **31**(1), 73–83 (2017)
11. Jones, A.J.I., Artikis, A., Pitt, J.: The design of intelligent socio-technical systems. Artif. Intell. Rev. **39**(1), 5–20 (2013)
12. Makinson, D.: General patterns in nonmonotonic reasoning. In: (ed.) Handbook of Logic in Artificial Intelligence and Logic Programming (Vol. 3): Nonmonotonic Reasoning and Uncertain Reasoning, vol. III, pp. 35–110. Oxford: Clarendon Press (1994)