Cognitively adequate complexity of reasoning in a description logic: extended abstract

Jelle Tjeerd Fokkens[1], Fredrik Engström[1]

[1]University of Gothenburg, Universitetsplatsen 1, 405 30, Göteborg, Sweden

Abstract

The problem of finding automated and cognitively adequate explanations for entailments in knowledge bases is tackled by modelling a deductive reasoning process with the cognitive architecture ACT-R. This results in the model SHARP which simulates the reasoning process of a human executing an algorithm for deciding inconsistency of an $\mathscr{ALE}$ ABox. With SHARP one can make certain predictions about the inference time of this reasoning process. Based on the inference time two complexity measures are defined that are expectedly cognitively adequate by design.

Keywords

cognitive modelling, description logic, complexity measure

1. The Description Logic $\mathscr{ALE}$

An $\mathscr{ALE}$ ABox is a set of expressions of the forms $a : C$ or $(a, b) : r$, where C denotes a concept and r a role. Roles are primitive in this logic, but the concepts are constructed from a set of primitive concepts $A \in \mathbf{C}$ according to [1, p.48]:

$$C ::= A \mid \top \mid \bot \mid \neg A \mid C \sqcap C \mid \forall r.C \mid \exists r.C.$$

Note that negation only applies to primitive concept names. An interpretation $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is the domain, interprets $(\forall r.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \forall b.(a, b) \in r^{\mathcal{I}} \rightarrow b \in C^{\mathcal{I}}\}$ and $(\exists r.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} \mid \exists b.(a, b) \in r^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}$, while the other constructors are straightforwardly interpreted. $\mathcal{I} \models a : C$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$ and $\mathcal{I} \models (a, b) : r$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$.

2. The ABox Inconsistency Algorithm

The tableau-based algorithm inconsistent() decides inconsistency for a given ABox, as proved in [2, pp.78-81]. inconsistent() applies *syntax expansion rules* until either none can be applied anymore, or a *clash* is derived, i.e. for some individual name a and concept name A both $a : A$ and $a : \neg A$ derived.

CAKR'23: The 2nd International Workshop on Cognitive Aspects of Knowledge Representation, September 4, 2023, Rhodes, Greece

tjeerdfokkens@gu.se (J. T. Fokkens); fredrikengstrom@gu.se (F. Engström)

https://www.gu.se/om-universitetet/hitta-person/tjeerdfokkens (J. T. Fokkens);

https://www.gu.se/om-universitetet/hitta-person/fredrikengstrom (F. Engström)

© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

CEUR-WS.org/Vol-3548/paper6.pdf

For example, an application of the ∃-rule to $\{a : \exists r.C\}$ amounts to deriving both $(a, d) : r$ and $d : C$, where d is new; the other syntax expansion rules function straightforwardly. If a clash is found, inconsistent() outputs 'inconsistent' and else it outputs 'consistent'.

Each next assertion to apply the rule to is nondeterministically selected; the list of assertions in order of being selected until decision is called the *run*.

3. The Cognitive Architecture ACT-R

The cognitive architecture ACT-R is an integrated theory of human cognitive behaviour, as well as a software with which this theory can be used to make various models.

ACT-R works by manipulating chunks, which are lists of slot-value pairs that codify information. These chunks can be temporarily stored in buffers that are connected to the procedural memory where manipulations take place by the application of production rules. The latter are condition-action rules, where the conditions apply to the chunks present in the buffers and the actions modify them and create new ones.

Each chunk has an associated *activation* which is a numerical value that determines the chunk's retrieval process as well as possible learning effects. The value of a chunk's activation is determined by which times a chunk is used as well as some random noise.

More information can be found in, for example, the ACT-R tutorial [3], or in [4].

4. The Model SHARP

We implemented the algorithm inconsistent() into ACT-R; the resulting model is called SHARP, which stands for **S**imulating **H**uman **A**Box **R**easoning **P**erformance. The idea is that SHARP can be used to estimate the cognitively adequate complexity of the $\mathscr{ALE}$ ABox inconsistency task, so that justifications of inconsistencies in knowledge bases can be selected according to this measure.

SHARP differs slightly from the original inconsistent() algorithm to accommodate for certain characteristics of ACT-R. The main challenge is to keep SHARP from making the same inference more than once, which is ensured by the use of a variety of different lists that store certain expressions already derived. One important consequence of this is that SHARP differs from the inconsistent() algorithm in that it cannot process ABoxes that are too large, because the above mentioned lists would overflow. Other issues are discussed in [5].

For convenience SHARP's production rules are grouped into five components with each its own function: finding a clash, selecting the next assertion(s) and applying the respective syntax expansion rules.

5. The Predictions

With SHARP we can predict the inference times that correspond to a given $\mathscr{ALE}$ ABox. An example of the simulated inference times of the ABoxes $\mathscr{A}_0 = \{a : (A \sqcap (B \sqcap (C \sqcap (D \sqcap \neg A))))\}$ and $\mathscr{A}_1 = \{a : (A \sqcap B),\ a : (B \sqcap C),\ a : (C \sqcap D),\ a : (D \sqcap \neg A)\}$ is given in figure 1.

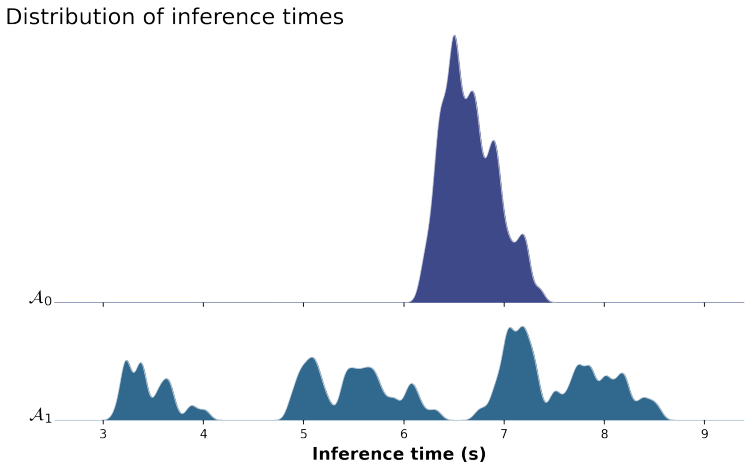

Figure 1: 300 simulations were run on each ABox. $\mathscr{A}_1$ shows a bigger spread in inference times than $\mathscr{A}_0$

The inference times for $\mathscr{A}_1$ exhibit a bigger spread than for $\mathscr{A}_0$ Likewise, with SHARP one can predict that a change of element or concept name does not change the inference time in certain situations. Moreover, a certain class of ABoxes displays exponential scaling of inference time with syntactic complexity because of *AND-branching* [2, p.107]. Finally, the order of presenting the ABox formulas and the order of conjuncts within conjunctions does not affect the inference time according to SHARP.

The paper then defines two complexity measures on ABoxes based on SHARP. One is the mean inference time $C_{mean}(\cdot)$, the other is the mean inference time of the fastest run $C_{least}(\cdot)$; both taken from a large enough sample of simulations to achieve a desirably robust accuracy. The latter measure typically has no assertions that are irrelevant for deriving the clash.

The two definitions correspond, respectively, to humans not using or using non-trivial heuristics in selecting which assertion(s) to make an inference on. In the near future, experiments are planned to test the predictions made by SHARP.

References

[1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. Patel-Schneider, Description Logic Handbook, Cambridge University Press, Cambridge, 2003.

[2] F. Baader, I. Horrocks, C. Lutz, U. Sattler, An introduction to Description Logic, Cambridge University Press, Cambridge, 2017.

[3] Act-r software, http://act-r.psy.cmu.edu/software/, 2023. Accessed: 2022-12-15.

[4] J. Whitehill, Understanding act-r - an outsider's perspective (2013). URL: https://doi.org/10.48550/arXiv.1306.0125.

[5] J. T. Fokkens, Modelling the logical mind, 2023. URL: https://hdl.handle.net/2077/74797.