Journal of Logic, Language and Information
https://doi.org/10.1007/s10849-026-09473-z

Estimating the cognitive complexity of description logic entailments with a cognitive architecture

Jelle Tjeerd Fokkens[1] · **Fredrik Engström**[1]

Accepted: 23 April 2026

Abstract
Debugging an ontology stated in description logic is a cognitively straining task for humans and there has been a recent increase in literature discussing various methods of facilitating this task. While some of these methods involve selected fragments of cognitive theory, none systematically integrate a cognitive theory. This paper aims to fill that gap by using the cognitive architecture ACT-R to model the ABox consistency task; a task which is essential for debugging ontologies. The resulting model is called SHARP (Simulating Human ABox Reasoning Performance) which simulates the ABox consistency algorithm as if it were run by a human mind. We evaluate the model by comparing its predicted inference times with empirical data gathered from an experiment with 70 participants. The complexity measures based on SHARP prove very accurate in predicting the complexity ordering of the set of ABoxes used in the experiment and outperform other measures found in the literature. These measures can thus be considered cognitively adequate, which opens up opportunities for them being utilised in cognitively optimised debugging of ontologies. Besides the complexity ordering, several other hypotheses are tested, which demonstrate that SHARP displays certain inaccurate consequences on a finer scale. These are considered points of improvement.

Keywords Cognitive modelling · Act-r · Description logic · Ontologies

1 Introduction

Many industries make use of data management systems that are based on so-called *ontologies*. These ontologies are highly structured representations of domain knowledge and allow for efficient data storage, because a myriad of specific facts can be

✉ Jelle Tjeerd Fokkens
tjeerd.fokkens@gu.se

Fredrik Engström
fredrik.engstrom@gu.se

[1] Department of Philosophy, Linguistics and Theory of Science, University of Gothenburg, Box 100, Gothenburg 40530, Sweden

Published online: 14 July 2026

derived from only a few general facts by means of *description logics*. Ontologies can, however, contain mistakes and ontology debugging is a common process that often proves to be cognitively demanding (Warren et al., 2015).

This problem is usually illustrated in the context of the medical ontology Snomed CT, which is widely used as a standardised terminology for medical concepts[1] (Baader & Suntisrivaraporn, 2008). The standard example involves the incorrect reasoning that an amputation of the finger entails the amputation of the upper limb. The culprit is an incorrectly stated axiom and humans generally find it difficult to assess its incorrectness.

Examples like the above illustrate that AI systems are not always understandable for humans, even if they are based on logic as opposed to probabilistic reasoning. In recent years there has been a strong push towards explainable AI systems (Whittaker et al., 2018). Generating comprehensible explanations of automated description logic entailments is part of this trend. The specific benefit in this case is an easier debugging process for ontologies that would result in more reliability of such systems.

One technique to reach better explainability is *axiom pinpointing* (Peñaloza, 2019), where the aim is to find *justifications*, i.e. the smallest set of axioms from the ontology (not necessarily unique) that gives rise to the undesired consequence. There have been attempts to optimise axiom pinpointing by, for example, supplementing the justification with a (partial) proof structure and investigating different methods of visualising these proofs (Alrabbaa et al., 2020). More recently, attention has been given to the cognitive aspects of these techniques to further improve the quality of the explanations of entailments. Among other things, it is highly desirable to have a cognitively adequate complexity measure to make selections among such explanations. Using the complexity measure, the easiest justification can be selected from a given set of justifications. The complexity measure could also be applied to proofs, for example to select the easiest to understand proof in the interactive description logic proof visualisation software EVONNE (Alrabbaa et al., 2022). This paper aims to construct such a measure.

Some concepts from the psychology of human deductive reasoning have previously been applied to construct such measures, but cognitive architectures, despite being state-of-the-art psychological theories that are successful in modelling many experiments, have not been applied in this context. Cognitive architectures are integrated theories of human cognition founded on a wide variety of cognitive experimental results. Such architectures provide frameworks for building specific models and ACT-R (Adaptive Control of Thought - Rational) is arguably the most well-known (Anderson, 2007). One area in which cognitive architectures perform well is the modelling of solving algebraic equations. Researchers in Qin et al. (2004) used the cognitive architecture ACT-R to accurately model solving three types of equations as well as the change in the study's participants' performance over time. Solving algebraic equations is similar to reasoning with a description logic, because both are examples of symbolic reasoning. This similarity is a major motivation for using ACT-R to model the cognitive task of reasoning with a description logic.

[1] And since recently being adopted as a standard in Västra Götalandsregionen in Sweden Eriksson (Clinical terminology SNOMED CT, 2021).

This paper aims to fill the gap in the literature on ontology debugging: understanding the cognitive aspects of description logic entailments in order to generate comprehensible explanations that facilitate ontology debugging processes. The project focusses on ABoxes (sets of assertional axioms only) in the description logic $\mathcal{ALE}$ (Attributive Language with Existential restrictions). The resulting model, SHARP (Simulating Human ABox Reasoning Performance), is an implementation of the abstract $\mathcal{ALE}$ ABox consistent() algorithm into ACT-R, thereby simulating this algorithm as if it were run by a human mind (Fokkens, 2023).

Among other things, SHARP outputs the simulated inference time. We consider inference time an important aspect of the cognitive complexity of ABoxes, i.e. the difficulty a human might experience while understanding or interacting with ABoxes. Hence, using this simulated inference time, we attempt to accomplish the aim of this paper:

define a measure that reflects the cognitive complexity of the $\mathcal{ALE}$ ABox consistency task

This is expected to be an improvement over several such other measures in the literature, as most of these measures are based on ad hoc assumptions without a firm base in cognitive science. Finding a measure that accurately reflects human performance should make the debugging of ontologies easier, as the human user can then be presented with the cognitively simplest-to-understand justification when a contradiction is found.

The effectiveness of SHARP for modelling cognitive complexity is verified in this paper by testing eight hypotheses that follow from SHARP. This paper also discusses how well the inference time correlates with the difficulty that the subjects report.

The first part of this paper provides the background of this research and discusses in turn the description logic $\mathcal{ALE}$, the cognitive architecture ACT-R, and the model SHARP together with the hypotheses that follow from it. Then a brief description of the empirical data collection is given. The rest of the paper is structured after the hypotheses mentioned, with each hypothesis (or group of hypotheses) being analysed and discussed in turn. Then follows a final conclusion with recommendations for further research.

2 Background

This paper uses the cognitive architecture ACT-R to model reasoning in a description logic with the aim to define a complexity measure on ABoxes that is cognitively adequate. Being able to accurately determine the cognitive complexity of ABoxes allows the selection of the ABox which is easiest for humans to process, which forms an vital step in the debugging process. We concentrate on the description logic $\mathcal{ALE}$.

2.1 The description logic $\mathcal{ALE}$

Description logics form a family of logics that are designed to reason about concepts (Baader et al., 2017). They can be used in knowledge bases to reason about the concepts that specific individual items satisfy. There are many different logics within this family

with every description logic striking a different balance between expressiveness on the one hand and computationally efficiency of reasoning algorithms on the other. $\mathcal{ALE}$ stands for (Attributive Language with Existential restriction); its concepts are formed by:

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

where A is a primitive concept symbol and r is a primitive role symbol. Note that negation ($\neg$) only applies to primitive concepts.

In $\mathcal{ALE}$ we can form the knowledge base Baader et al. (2017):

$$\begin{aligned}
\texttt{Teacher} &\equiv \texttt{Person} \sqcap \exists \texttt{teaches.Course} \\
\texttt{Student} &\equiv \texttt{Person} \sqcap \exists \texttt{attends.Course} \\
\texttt{Martin} &: \texttt{Person} \\
\texttt{LOG011} &: \texttt{Course} \\
(\texttt{Martin}, \texttt{LOG011}) &: \texttt{teaches}
\end{aligned}$$

which can be straightforwardly interpreted and from which it follows that Martin is a teacher. The latter can be seen by observing the definition of teacher (first line) and the facts that Martin is a person, LOG011 is a course and Martin teaches it (last three lines). The benefits of description logics is that many of these inferences (and in the case of $\mathcal{ALE}$ *all* such inferences) can be automated.

More formally, the semantics of $\mathcal{ALE}$ is as follows. The role symbols are interpreted by an interpretation $\mathcal{I}$ with associated domain $\Delta^{\mathcal{I}}$ as relations on that domain: $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The concepts are interpreted by a given interpretation $\mathcal{I}$ as subsets of a given domain $\Delta^{\mathcal{I}}$, so A is interpreted as $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$. The interpretation of $\neg A$ is the complement of A, so $(\neg A)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} | a \notin A^{\mathcal{I}}\}$. Intersections ($\sqcap$) are interpreted as $(C \sqcap D)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} | a \in C \text{ and } a \in D\}$. Existential restrictions ($\exists r.$) are interpreted as $(\exists r.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} | \exists b \in \Delta^{\mathcal{I}} \text{ such that } (a, b) \in r^{\mathcal{I}} \text{ and } b \in C^{\mathcal{I}}\}$ and resemble the indexed diamond operator from modal logic. Universal restrictions ($\forall r.$) are interpreted as $(\forall r.C)^{\mathcal{I}} = \{a \in \Delta^{\mathcal{I}} | \forall b \in \Delta^{\mathcal{I}} \text{ with } (a, b)^{\mathcal{I}} \in r^{\mathcal{I}} \text{ it holds that } b \in C^{\mathcal{I}}\}$ and resemble the indexed box operator from modal logic. Note that in the latter case, there does not have to be a b with $(a, b)^{\mathcal{I}} \in r^{\mathcal{I}}$ for the individual a to satisfy the concept $\forall r.C$ in the interpretation $\mathcal{I}$.

With the concepts defined, the assertional statements, of the form:

- $a : A$ are satisfied by interpretation $\mathcal{I}$ if and only if $a^{\mathcal{I}} \in A^{\mathcal{I}}$, and
- $(a, b) : r$ are satisfied by interpretation $\mathcal{I}$ if and only if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$.

Fig. 1 The syntax expansion rules for $\mathcal{ALE}$ ABox consistency. After: (Baader et al., 2017, p.73). Each rule derives simpler consequences from a more complex formula

$\sqcap$-rule: if $a : C \sqcap D \in \mathcal{A}$ and
$\{a : C,\ a : D\} \nsubseteq \mathcal{A}$,
then $\mathcal{A} \rightarrow \mathcal{A} \cup \{a : C,\ a : D\}$.
$\exists$-rule: if $a : \exists r.C \in \mathcal{A}$ and
there is no b such that $\{(a, b) : r,\ b : C\} \subseteq \mathcal{A}$,
then $\mathcal{A} \rightarrow \mathcal{A} \cup \{(a, d) : r,\ d : C\}$, where d is new in $\mathcal{A}$.
$\forall$-rule: if $\{a : \forall r.C,\ (a, b) : r\} \subseteq \mathcal{A}$ and
$b : C \notin \mathcal{A}$,
then $\mathcal{A} \rightarrow \mathcal{A} \cup \{b : C\}$.

Algorithm consistent()
Input: an $\mathcal{ALE}$ ABox $\mathcal{A}$.
repeat
  if $\mathcal{A}$ contains a clash **then**
    return "inconsistent"
  end if
  select a rule R that is applicable to $\mathcal{A}$, and a (pair of) formula(s) $\alpha \in \mathcal{A}$ to which R is applicable
  set $\mathcal{A} \rightarrow \mathsf{exp}(\mathcal{A},\ R,\ \alpha)$
until $\mathcal{A}$ is complete
if $\mathcal{A}$ contains a clash **then**
  return "inconsistent"
else
  return $\mathcal{A}$
end if

Fig. 2 The tableau algorithm consistent for $\mathcal{ALE}$ ABox consistency. After: (Baader et al., 2017, p.75). $\mathsf{exp}(\mathcal{A},\ R,\ \alpha)$ is the ABox obtained from applying rule R to α

2.1.1 ABox consistency

A set of assertional statements is called an *ABox*; the last three lines of the example in the previous section form such an ABox. In $\mathcal{ALE}$ ABoxes can be either *consistent* or *inconsistent*. The former means that there exist an interpretation $\mathcal{I}$ that satisfies all the formulas of the ABox; in the latter there exists no such interpretation. There exists a decision procedure on the consistency of ABoxes, shown in Figure 2. The algorithm makes use of the *syntax expansion rules*, shown in Figure 1. The syntax expansion rules preserve truth and are applied exhaustively to the ABox to derive an expanded ABox. In this expanded ABox, the algorithm searches for an obvious contradiction called a *clash*. A clash is the situation when an individual a is assigned to two contradictory concepts C and $\neg C$, i.e. the ABox has a subset of the form $\{a : C, a : \neg C\}$ for some individual name a and concept name C. In the presence of a clash the original ABox is inconsistent, while in absence of one it is consistent.

For example, the ABox:

$$\{\texttt{Martin:Teacher} \sqcap \texttt{Person}\}$$

can be expanded by applying the $\sqcap$-rule to yield the ABox:

$$\{\texttt{Martin:Teacher} \sqcap \texttt{Person},\ \texttt{Martin:Teacher},\ \texttt{Martin:Person}\}$$

The decision problem of consistency of an $\mathcal{ALE}$ ABox – extensively studied from a formal perspective – is the task we aim to study from a cognitive perspective. To this end, we translated the above algorithm into chunks and production rules so that

it can be run in the cognitive architecture ACT-R, thereby simulating a human mind performing the algorithm; details of this translation will follow in section 2.5.

2.2 Capturing cognitive complexity

Cognitive complexity is – like creativity (Simonton, 2012), intelligence (Neisser, 1979) and other psychological constructs – difficult to define both conceptually and operationally. The concept of cognitive complexity is multifaceted, consisting of factors such as: energy consumption, error rates, time and possibly more. The operational definition of the concept seems, moreover, dependent on the theoretical lens through which one is to explain the subject's internal processes during the task. Regarding deductive reasoning, several such theories exist, the most well-known ones being the *mental rules* theory and the *mental models* theory.

2.2.1 Mental Rules

The mental rules theory is expounded in Lance Rips' book *The Psychology of Proof* (Rips, 1994). The basic idea is that humans have certain deduction rules stored in their memory, the application of which to given premises yields certain conclusions. Each deduction rule has a certain probability of firing and the probability of correctly judging the validity of an argument is calculated by combining the probabilities of its deduction rules while taking guessing into account as well as situations where the conclusion can be deduced in multiple ways. This theory was extensively tested by experiments in which subjects were presented propositional logic arguments in natural language for which they were asked to judge the validity.

By measuring error rates on judging a set of given arguments, the error rates of the deduction rules could be found by a model fitting procedure. After that, the error rates of the arguments were computed to be very close to the measured error rates. So the way in which Rips operationalises cognitive complexity of arguments relies, informed by his mental rules theory, on the measurable error rate of human performance.

2.2.2 Mental Models

Another theory is the mental models theory, defended by Philip Johnson-Laird in (Johnson-Laird, 2001). This theory states that the reasoner creates mental representations of the premises of a given argument using both the semantic content of the premises and background knowledge. This can result in a unique mental model, or a set of mental models representing different possibilities. The theory explains semantic effects in reasoning by the subjects using their background knowledge for constructing models. Certain reasoning mistakes are explained by the failure of creating all models necessary for a successful deduction.

The mental models theory informs the operational definition of cognitive complexity of deductions not only by considering the error rate of judging arguments, but also by the time that these processes take: creating more models, besides being more error-prone, requires more time.

2.2.3 Inference time and attitudes

The above theories are examples of cognitive complexity of deduction being operationalised by the error rate that humans show on various tasks related to deductive reasoning, as well as the time that deductive inferences take. Without having fully conceptually defined the cognitive complexity of deductive reasoning, we will focus on both the time that deductive inferences take and the attitude that humans have towards these inferences. The latter aspect is included because we think that humans have a certain degree of introspective insight into their performance on these tasks. We expect both included operational aspects to converge to the same construct. In this paper the error rate is not used to operationalise cognitive complexity, because high error rates cast doubt on whether humans are actually performing the required task, i.e. whether they correctly reason towards an interpretation of the logical symbols (Stenning & Lambalgen, 2012).

2.3 Existing complexity measures

Another question related to the cognitive complexity of deciding $\mathcal{ALE}$ ABox consistency – however conceptually or operationally defined – is whether it is possible to estimate the cognitive complexity of such a task from the ABox's formal properties and if so, how this is done.

In Warren et al. (2015) both the mental models theory and the relational complexity theory are used to define measures on description logic formulas. It seems that the relational complexity theory is not much different from the mental models theory as the proponents of the relational complexity theory write in Halford et al. (2010): "Mental models embody the core properties of relational knowledge". Moreover, in practice, the two theories quantify the complexity of reasoning quite similarly.

It should be noted that complexity measures are sometimes dependent on the logic and/or proof system used and do not always directly translate to $\mathcal{ALE}$; in many cases, however, a simple adaptation can be found and is assumed in the following. For example, Strannegård et al. (2013) (Section 6, bullet point 1) and Alrabbaa et al. (2020) (Introduction, bullet points 2, 3 and 7) mention formal measures related to the *size* of the theory/ABox, ($<_{size}$) as correlating to the cognitive complexity. The latter reference (bullet point 8) and Alrabbaa et al. (2021) (Introduction) also mention the *proof depth* ($<_{pd}$) as such a formal measure. The *proof length* ($<_{pl}$) is mentioned by all three sources (bullet point 8, bullet point 4, Introduction, respectively).

Furthermore, Strannegård et al. (2013) mentions:

- Number of (universal or existential) restrictions ($<_{quan}$)
- Cardinality, i.e. the number of elements in the canonical model ($<_{card}$)
- Number of edges in the canonical model ($<_{edge}$)
- Proof size (total size of all expressions in a minimal proof) ($<_{ps}$)

The novel approach of this paper is the use of a cognitive architecture to model the $\mathcal{ALE}$ ABox consistency reasoning task. ACT-R allows us to model the inference time of this reasoning task, which we think is an important operational aspect of its cog-

nitive complexity. The simulation results of this model translate into two complexity measures that are firmly founded in cognitive theory by design.

2.4 ACT-R

This section contains a short and simplified overview of the ACT-R cognitive architecture, with its most important aspects highlighted to understand the rest of the paper; more detailed information can be found in (Anderson, 2007), (Whitehill, 2013) and (Anderson & Byrne, 2004).

The basic idea of ACT-R is that the brain has different regions which are each highly specialised in certain tasks and can perform these tasks in massively parallel fashion. ACT-R represents these regions by *modules*, each of which has a *buffer* connecting the module to the *procedural memory*. The buffers act as communication bottlenecks and can contain only a limited amount of information.

2.4.1 Knowledge representation

Knowledge in ACT-R is encoded in *chunks* and *productions*. A chunk is a collection of attribute-value pairs that represents a unit of factual knowledge that a person can process. The attributes are called *slots*, each of which can contain at most one string (that has no additional structure otherwise). An example of a chunk is:

```
Formula 1
  formula (a,b):r
  element1 a
  element2 b
  role r
```

`Formula 1` is the name of the chunk and helps it to distinguish it from other chunks, but it has no significance in the modelling. The production rules encode procedural knowledge, which corresponds to skills that people themselves generally cannot easily express in words. They are condition-action rules and are stored in the procedural module. An example is:

```
Rule 1
IF
  goal inference
  and buffer1 (a,b):r
  and buffer2 a:∀r.C
THEN
  create chunk for b:C
```

`Rule 1` is the production rule's name and helps to distinguish it from other production rules, but it has no significance in the modelling. The production rules operate on the chunks to produce other chunks and actions in the modules, resulting in behaviour intended to model the cognitive behaviour of a human performing the same task. The modules relevant for SHARP will be discussed in turn below.

2.4.2 The Goal and Imaginal modules

The *goal* and *imaginal* modules are used to monitor the state of the process, store context-relevant information and plan the next step to be executed. The goal module has one associated buffer (*goal buffer*) and the imaginal module has two associated buffers (the *imaginal* and the *imaginal-action* buffer).

2.4.3 The Declarative module

The *declarative* module (alternatively called the declarative memory) contains all the factual information of the system, encoded in chunks. It has one buffer, named the *retrieval* buffer which can store one chunk. The module responds to a retrieval request by searching through all its stored chunks stored for a match and places the matching chunk in the buffer; in case multiple chunks match a random selection among them is made.

2.4.4 The Procedural module

The *procedural* module does not have a buffer. It contains all the model's production rules and is continuously monitoring all the buffers' contents for matching the conditions of any production rule. In case of a match, one rule is *fired*, which means that the action of that rule is performed.

2.4.5 Activation

Each chunk i has a certain *activation* A_i, a real numbered value that is modelled by: the number of *presentations*, the time passed since each presentation, the *decay* parameter d, which is usually set to 0.5 and a random *noise* component. The simplest expression of the activation is:

$$A_i = \ln\left(\sum_{j=1}^{n} t_{ij}^{-d}\right) + \epsilon_i$$

The (instantaneous) noise ϵ_i is a random number drawn from a logistic distribution with mean 0 and standard deviation $\sigma^2 = \frac{\pi^2}{3}s^2$, where s is the noise value which can be set by the modeller. The number of uses (creation and re-creation) of the chunk is denoted n and t_j is the time at which use j occurs. This expression describes *Base-level learning* and it models the retrieval of chunks from the declarative memory in a way that corresponds to empirical results (Anderson & Byrne, 2004). ACT-R allows for spreading activation as well as other activation-related mechanisms; these are ignored in the following.

In case a chunk's activation drops below the retrieval threshold τ, the chunk cannot be retrieved. The probability that the activation of chunk i is above τ (and hence can

be retrieved) is described by:

$$P = \frac{1}{1 + \exp\left(\frac{\tau - A_i}{s}\right)}$$

The larger the noise value s, the more gradual is the transition from 0% to 100% recall, while smaller values of s make this transition more abrupt. If the decay parameter d has a large value all chunks' activations drop very rapidly.

The retrieval latency is also modelled by the chunks' activation and is given by the formula:

$$T_i = Fe^{-A_i}.$$

Where i is the chunk's name and F is the latency factor, which is set by the modeller; its default value is 1. Higher activations allow chunks to be retrieved more quickly. The time duration of a retrieval failure is given by the same expression with the retrieval threshold τ substituted for the activation A_i.

2.4.6 Compilation

There are two mechanisms related to learning at the symbolic level of the production rules. Firstly, there is *proceduralisation*. After the firing of an instantiation of a production rule that has a variable, this instantiation is stored in the procedural memory. The next time the same instantiation can fire, it competes with the original production rule and the one with the highest utility is selected.

Secondly, there is production rule *composition* where two rules that fire after one another combine into one rule. This new rule has the conditions of the first and the actions of the second. Possible actions of the first rule that are not negated by the second rule will also be put in right hand side of the newly created rule.

The two mechanisms can create production rules that skip certain steps compared to the original reasoning process, thereby reducing the time needed to execute that process.

2.4.7 Sub-symbolic parameters

ACT-R has many sub-symbolic parameters, the default values of which are based on previous experiments (Anderson & Lebiere, 1998) and later supported by fMRI (functional Magnetic Resonance Imaging) data (Qin et al., 2004), (Borst and Anderson 2013) and (Borst and Anderson 2015). However, these parameters can, and often will be, tweaked to make the model more accurate in its simulations. Only few sub-symbolic parameters are relevant for SHARP:

- noise parameter s
- retrieval threshold τ
- decay parameter d
- latency factor F.

2.4.8 Application: algebraic equations

ACT-R was used to model human performance on solving simple algebraic equations in (Qin et al., 2004) and (Anderson, 2005). In these studies, children were asked to solve linear algebraic equations mentally. The equations came in three different categories of difficulty, depending on the number of arithmetic operations that were needed to be performed to solve the equation. By encoding the equations as chunks and the arithmetic operations as production rules, researchers were able to relate the children's performance to brain scan data, model their performance on this task, as well as their learning over time. The production rules modelled basic processes such as observing the equation and retrieving the sum of two numbers from the declarative memory. The model's efficiency increased over time due to ACT-R's rule compilation: several production rules combined into one production rule that skips some time-consuming steps of the process. Using these newly created production rules, the model accurately simulated human's performance on learning the task, showing a 0.986 correlation coefficient between the empirical data and formulas that were derived from the model which had parameters fitted parameters to that data. To corroborate the findings, the activity of each ACT-R modules was used to predict brain activity as measured by an fMRI scan in terms of the BOLD (Blood Oxygen Level Dependent) response in the corresponding brain regions.

In analogy to these studies, the model SHARP is based on encoding $\mathcal{ALE}$ ABox formulas as chunks and the syntax expansion rules in Figure 1 as production rules, thereby translating the algorithm in Figure 2 into ACT-R with the aim of accurate modelling of human performance of the $\mathcal{ALE}$ ABox consistency task.

2.5 SHARP

Algebraic equations allow practically only one method of solving: firstly a possible addition or subtraction and secondly a division or multiplication. In the case of deciding whether an ABox is consistent – as in other cognitive tasks of sufficient complexity – the situation is quite different in that multiple strategies and heuristics can be used, which is partly confirmed by private communication with subjects in the current study. In such cases it is unclear which steps the reasoner is making and it is therefore impossible to model these intermediate steps with the same detail as modelling the intermediate steps of solving the algebraic equations. Moreover, these strategies seem difficult to model with ACT-R's learning mechanisms discussed above.[2] Rather, these strategies appear to emerge from careful reflection on the task. We argue, however, that the desired measure on ABoxes can be approximated by modelling a subject mentally performing the tableau algorithm in Figure 2, because the average over a sufficiently large set of strategies is expected to remove the bias that one strategy has over another with respect to the relative complexity of two given ABoxes. Also, the tableaux algorithm is relatively simple, does not involve clever tricks that seem to

[2] One example of a strategy is performing a quick scan for negations; if none are present, the ABox has to be consistent.

short-cut the reasoning process and should therefore not be biased towards a specific class of ABoxes.

In translating the tableau algorithm into ACT-R, $\mathcal{ALE}$ formulas are translated to chunks and the syntax expansion rules from Figure 1 are translated to ACT-R's production rules, similar to the examples in section 2.4.1. The latter examples are highly simplified, because the formal machinery to guarantee termination of the algorithm requires keeping track of many extra variables such as: if a formula is derived, if a formula has already been used to make an inference on, etc. This requires the formula chunks and the production rules to contain many more slots. The source code is constructed with the pyactr package (Dotlacil, 2024) and can be found in (Fokkens & Engström, 2024).

2.5.1 SHARP's simulations

To facilitate the description of SHARP's simulations, the code is split up into five components that each execute a subtask of the consistent() algorithm; this way of organising the code has no other meaning. An visual overview of the code can be seen in Figure 3.

A simulation by SHARP starts by directly loading the formulas of the ABox into the declarative memory; visual processes are not modelled. The first component inspects only those chunks in its declarative memory that represent assertional $\mathcal{ALE}$ formulas with atomic concepts (primitive and negated); among them it searches for a clash.

If no clash is found and not all formulas are inspected yet, the second component is invoked, which searches for a complex formula to make an inference on; it corresponds to the 'select' step in the algorithm in Figure 2. After selecting a conjunction, existential restriction or a universal restriction, the third, fourth or fifth component is invoked respectively.

The third component creates two chunks representing the formulas corresponding to the conjuncts of the formula selected by the second component; this component corresponds to the $\sqcap$-rule of the expansion rules in Figure 1.

The fourth component derives a formula with a new element and the concept from the the existential restriction formula selected by the second component; this component corresponds to the $\exists$-rule of the expansion rules in Figure 1.

Lastly, the fifth component only fires after no formulas of the previous two kinds can be selected; the reason for this implementation is explained in section 2.5.2. The component searches for a chunk representing a universal restriction formula and a chunk representing a role formula, with the role being the same in both formulas and the first element in the role formula being the same as the element of the first. After finding these two formulas it creates the chunk representing the appropriate derived formula; this component corresponds to the $\forall$-rule of the expansion rules in Figure 1.

SHARP outputs, first of all, the simulated inference times. These inference times show randomness due to the random selection processes related to the retrieval of chunks based on their activations. Second, SHARP outputs the *run* which is the list of formulas that are consecutively inspected during the simulation process. Formulas in this list are not necessarily used for inference steps, because Component 5 may inspect a universal restriction formula on which no inference can be made. Runs for ABoxes

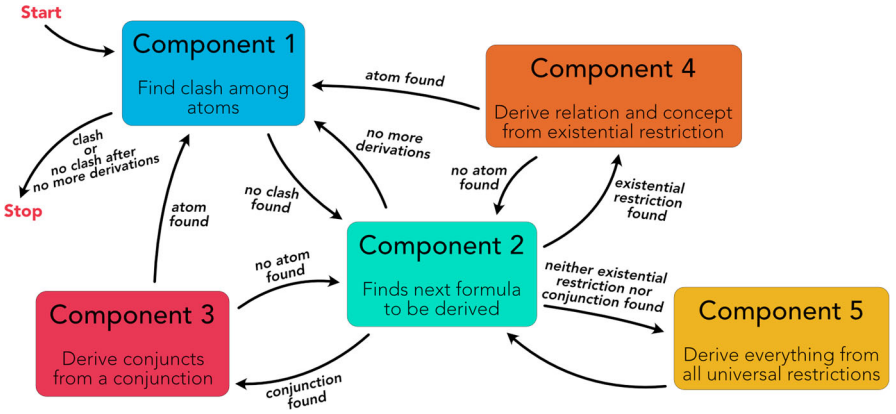

Fig. 3 To facilitate the code's description SHARP's production rules are categorised into 5 components. The components focus on specific subtasks of the ABox consistency decision procedure but have no other meaning. Component 1 looks for a clash among atomic concept formulas. Component 2 selects the next formula to make an inference on. Components 3, 4 and 5 correspond to the $\sqcap$-rule, the $\exists$-rule and the $\forall$-rule respectively

with universal restrictions, therefore, generally vary in length. The chosen values for the sub-symbolic parameters are:

- noise parameter, $s = 0.005$
- retrieval threshold, $\tau = -0.05$
- decay parameter, $d = 0.005$
- latency factor, $F = 1.0$.

2.5.2 Implementational Issues

The chunks in ACT-R's declarative memory cannot be removed. This poses problems for executing the algorithm, because there is no obvious way to prevent SHARP from making the same inference twice. Indeed, retrieving chunks increases their activation, which makes them easier to retrieve next time, thereby risking infinite looping. Certain chunks keep therefore track of which formulas have been already inspected, thereby disqualifying them from being selected.[3] This solves most problems, except in the case of the fifth component, because universal restriction formulas may need to qualify for selection after a corresponding role formula is derived. To counter this problem, the universal restriction formulas are all derived after one another and the number of times the fifth component is invoked is counted; this number is used to prevent the same inference step on a universal restriction formula to be performed again. Another consequence of this is that for consistent ABoxes – i.e. ABoxes that do not give rise to a clash – the fifth component is always the last component to be invoked and necessitates some undesired overhead in simulated inference time. Because of the issues discussed, the algorithm in 2 could not be implemented completely faithfully.

[3] This mechanism may have also been implemented by declarative finsts, but this would require some reverse engineering to determine the number of the finsts as well as how long they persist. On top of that, other bookkeeping mechanisms in SHARP cannot be simulated by finsts.

Another problem with SHARP is related to base-level learning. The activation of a chunk decays over time and if it drops below the retrieval threshold, it becomes impossible to retrieve that chunk. This poses practical problems for simulations on ABoxes that involve long runs, because by time that the chunk needs to be retrieved its activation may have decayed below the retrieval threshold. With certain chunks not being able to be retrieved, SHARP fails to draw conclusions on the consistency of the ABox. An example of this is discussed in section C.

3 Data

In this paper two sorts of data are used: simulated data by SHARP and empirical data Fokkens and Engström (2023) gathered through an online questionnaire. The simulated data consists of a list of ABoxes together with their inference times and their runs (list of formulas being inspected consecutively by SHARP).

The questionnaire was distributed by email among researchers, teachers and their students from several Swedish, German and Dutch universities that offered (mathematical, philosophical and/or computer science) logic education. It was running from June to November 2023 until 70 *usable* (see next subsection on what is meant by this) data points were collected. The questionnaire consisted of:

- Three questions about previous experience in logic.
- Questions about age, gender and nationality.
- A 4-minute introduction video with instructions on the reasoning task.
- Two practice ABoxes for which the subjects decided the consistency.
- Twenty ABoxes for which the subjects also decided the consistency; these are listed in the A.
- Four difficulty rating questions.

The first three questions related to previous logic experience ensured that the subjects had the required knowledge to complete the questionnaire; failing all questions denied access to the rest of the questionnaire, passing at least one question is considered sufficient for understanding the rest of the experiment.

The demographic information was gathered to monitor possible bias in the sample. The subjects were aged between 19 and 49 years old, with an average age of 28. The sample had 45 Male and 23 Female persons, one transgender person and one person who preferred not to share information on their gender. Most subjects were Swedish (20), German (12) and Indian (8), with other nationalities represented by fewer than 4 subjects. The above demographics were not considered problematic for the analysis and ensuing conclusions.

The introduction video [28] discussed the description logic $\mathcal{ALE}$ and how inconsistency works within this logic, illustrated with examples; the syntax expansion rules were not discussed as we did not want to force the subjects into using a single solving strategy.

The concept element, and role names in the ABoxes are chosen to be letters and not English words. The reason is to avoid semantics-related modulation on logical reasoning, which is a well-known phenomenon (Stenning & Lambalgen, 2012). The

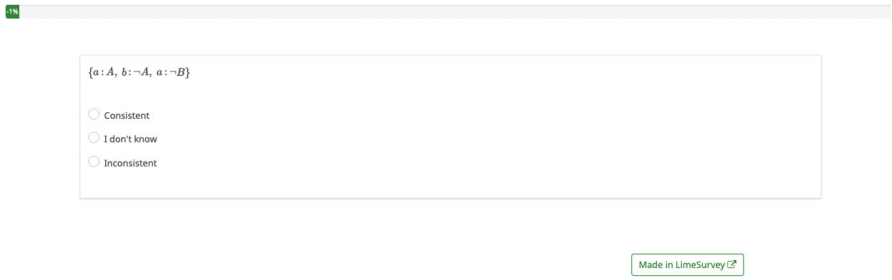

Fig. 4 Screenshot of the questionnaire's interface. Shown is the (consistent) ABox $\{a : A, b : \neg A, a : \neg B\}$.

subjects were presented the ABoxes on the left side in the middle of the screen and were asked to find an inconsistency using a 3-choice response format: 'Consistent', 'I don't know' and 'Inconsistent', which were displayed directly under the ABox. An example can be seen in Figure 4. On the top of the screen was a progress bar which indicated how much progress the subject made in the questionnaire. The response times were recorded. The first two practice ABoxes (one very easy and one very hard) were part of the familiarisation phase; the format of this phase was the same as the rest of the questionnaire and thus probably helped the subjects get familiar with the interface and the task. The subjects were told that these first two ABoxes were not part of the actual experiment; data on these ABoxes was not used for the analysis to compensate for possible learning effects. The 20 ABoxes were chosen with the aim of testing the hypotheses mentioned. The ABoxes were presented to every participant in randomised order to compensate for a learning effect that was indeed found in the data post hoc. In the difficulty rating questions, subjects were asked to rate the difficulty of the previous ABox; these questions were presented after every five ABoxes of the set of 20. Different subjects, therefore, generally rated different ABoxes.

Most participants could complete the questionnaire in half an hour. They were each compensated by a 100 SEK or 10 EUR voucher, for which we got financial support through the Stiftelsen Karl Langenskiölds minnesfond. To send these vouchers, we asked consent to use the participant's phone numbers. This data was deleted after closing the survey.

3.1 Exclusion

In total, we had 84 completed responses on the survey. To ensure good quality of the data, we consider a subject's data *usable* iff the ratio of correct versus incorrect responses (so excluding the 'I don't know' responses) exceeds the threshold of 75%; data not meeting this criterion is excluded from the analysis. This resulted in data from 70 subjects. Of the included data, there were two subjects that had a rate of 'I don't know' responses higher than 25% (namely 6 and 7 out of 20 answers); this was considered unproblematic.

The subjects' environments were uncontrolled and random distractions from the survey can therefore be expected to arise. Private (and unprompted) communication with some subjects revealed this to be the case indeed. The data has some inference

time outliers that can hardly be explained by non-distracted subjects that are thinking slowly, especially because those subjects were not *consistently* thinking slowly and had more typical inference times for the other ABoxes. Based on these considerations, it was decided that each inference time that lies more than 3 standard deviations away from the mean inference time for that ABox was excluded from the analysis; this amounted to removing between 0 and 3 data points for each ABox (32 in total).

4 Analysis and Results

This section discusses the research hypotheses, the data analysis and the corresponding empirical results. The methods of analysis differ among the hypotheses, so hypotheses that require the same analysis are discussed together; the common denominator of the analyses is, however, discussed first. After that, we show that the complexity measures defined with SHARP accurately represent the relative cognitive complexity of the $\mathcal{ALE}$ ABox consistency task, making it therefore suitable for application purposes. Then follow – on a more critical note – five hypotheses concerning inference times of certain pairs of ABoxes, which show that on a fine-grained scale certain areas that SHARP might need to be improved upon. Then there are two hypotheses that require specialised methods of analysis. Finally, the correlation between the inference times and the experienced difficulty scores is inspected, supporting the claim that these two quantities converge to the same concept.

It should be noted that this section – as mentioned before – analyses two types of data: the empirical data from the questionnaire and the simulation data from SHARP. Which type is under discussion is indicated.

4.1 Analysis

The analysis of the following hypotheses are using Bayesian statistical methods for which we used the PyMC Python library [29]. These methods allow for accepting null-hypotheses and are hence required for many of the following hypotheses. In contrast, standard frequentist methods only allow for the rejection of null-hypotheses and can therefore not be used for our purposes. Generally in Bayesian data analysis, before observing the data, a statistical model is chosen, as well as prior distributions on the model's parameters, after which the empirical data updates the prior through Bayes' theorem to result in a posterior distribution that can be used for drawing conclusions. Bayes' theorem is expressed as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}.$$

In this context, A is the event that a parameter has a certain value and B is the event that a measurement has a certain value. The expression calculates the posterior probability as a conditional probability of a parameter having a certain value, given the data. The right-hand side of the equation contains three probabilities:

- The first factor in the numerator is the *likelihood*; it denotes the probability of the data, given that the parameter assumes a certain value.
- The second factor in the numerator is the *prior*; it denotes the estimated probability of the parameter value, before observing the data.
- The denominator is the marginal probability of the data; it usually takes the form of an integral $\int P(B|A)P(A)dA$, which often cannot be analytically evaluated, in which case the posterior probability is difficult to normalise (i.e. to make sure the total probability equals 1).

The five hypotheses in section B below are, more specifically, analysed by 'Bayesian null-hypothesis testing' (Kruschke, 2014, Chapters 11-12). In Bayesian null-hypothesis testing, a ROPE (Region Of Practical Equivalence) is specified around the null value, which the HDI (Highest Density Interval) is compared with. The ROPE is an interval around the null value that contains the values considered equivalent to the null value. The HDI is the region of the most likely probability density of the posterior distribution (in the following the 94% most likely region); despite the name it is not necessarily an interval, e.g. a bimodal posterior distributions usually does not result in an HDI which is one interval, but two. We distinguish three cases in which a decision regarding accepting/rejecting the null-hypothesis can be made:

- If the HDI lies completely outside the ROPE, the null-hypothesis is rejected.
- If the HDI completely lies inside the ROPE, the null-hypothesis is accepted. Note that this conclusion cannot be drawn within the frequentist statistics framework.
- If the HDI partially overlaps with the ROPE and partially lies on the negative (positive) side of the ROPE, the hypothesis of the underlying effect being positive (negative) is rejected.

In other cases no conclusion shall be drawn. A Bayesian paired t-test is defined in analogy to a frequentist paired t-test. The differences of each subject's inference times on two ABoxes are modelled by:

$$\begin{aligned} \nu &\sim \text{Exponential}(\lambda = 1/30) \\ \mu &\sim \text{Cauchy}(\alpha = 0, \beta = 20) \\ \sigma &\sim \text{HalfNormal}(\sigma = 30) \\ y &\sim \text{StudentT}(\nu, \mu, \sigma) \end{aligned} \tag{1}$$

The Student t-distribution is chosen because it can flexibly model outliers (Andrade, 2022) and matches the overall shape of the data. This distribution is a generalisation of the Gaussian distribution because the thickness of its tails can be adjusted by the parameter ν and in the limit of $\nu \to \infty$ the distribution coincides with the Gaussian distribution. The prior distribution on ν is the exponential distribution with $\lambda = 1/30$; its mean and standard deviation are both 30. The prior distribution on the location parameter μ is given by Cauchy distribution, centred around $\alpha = 0$ with scale parameter $\beta = 20$; this distribution has no defined mean or standard deviation. The parameter σ is assumed positive and has a half-normal prior with standard deviation equal to 30; the mean of this distribution is $30\sqrt{\frac{2}{\pi}} \approx 23.9$. The above priors are weakly informative:

they are chosen to make the prior predictive distribution close to the data to achieve high informativeness of the posterior distribution while imposing few constraints on the data.

Because the posterior distribution is often impossible to normalise, a Markov-Chain Monte-Carlo approximation is used to create samples that approximate the normalised posterior distribution. In the following analysis specifically, NUTS (No U-Turn Sampling) Hoffman and Gelman (2014) is used to sample from the posterior distribution. This algorithm forms samples by first picking a random value for the parameter and adding it to the sample. Then, a second parameter value is picked, selected randomly by using information about the gradient of the numerator evaluated at the current point. The process is repeated until the sample is sufficiently large. It is important that each parameter value that is guessed, does not depend too strongly on the previously guessed value so that the entire distribution is sampled from. To ensure the sample is independent of the initial parameter value, several such values are selected at once, creating multiple samples called chains. Ideally, these chains converge to the same distribution after some time. Their similarity is expressed in a convergence measure called $\hat{R}$, which equals 1.0 in case of perfect convergence.

4.2 Comparison to Other Complexity Orderings

This section compares the performance of different complexity orderings and contains the most significant results of the paper. The Bayesian t-tests were performed on the empirical data from every ABox pair. The quality of the sampling is high, because the chains yielded very similar posterior distributions, showed no divergences and the $\hat{R}$-values where lower than 1.05. Furthermore, the sampling values showed very low auto-correlation: typically disappearing in the range of $[-0.05, 0.05]$ after 4 draws, with only 26 exceptions of the autocorrelation disappearing in this range after more than 20 draws. This resulted in all effective sample sizes being larger than 2000. The p-values on the median of the posterior predictive distribution, which express the probability that a single draw from the posterior distribution is higher than the data median lie in the interval $(0.46, 0.54)$, which indicates that the posterior distribution resembles the data well. Moreover, the posterior distributions showed a visually close fit to kernel density estimations of the data; a typical example is shown in Figure 5.

The above indicates that the parameters were reliably estimated and that statistical inferences from the posterior predictive distribution can be reliably made.

To quantify the strength of the relationship between two variables the effect size is used (Sullivan & Feinn, 2012). It is expressed by Cohen's d:

$$d = \frac{\bar{x}}{s},$$

where $\bar{x}$ is the mean, and s is the standard deviation of the sample.

Figure 6 shows the distribution of the approximated effect sizes relating to the Element Name Independence hypothesis which were estimated by taking samples ($n = 70$) from the posterior predictive distribution and calculating the effect size for each sample. As a ROPE we chose the interval $(-0.2, 0.2)$, which is standard in

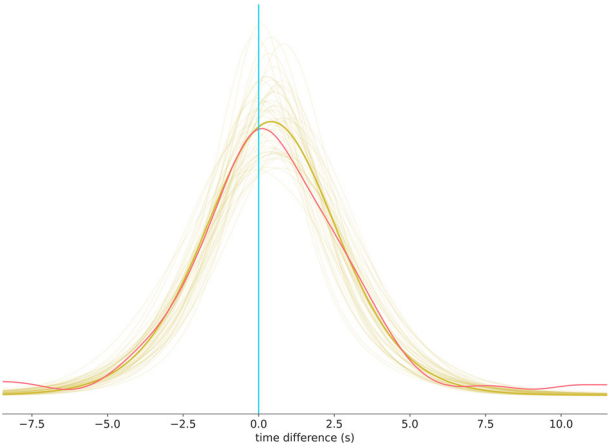

Fig. 5 Typical example of the posterior predictive distribution of time differences (olive) compared to a kernel density estimation of the empirical data (red). The light olive curves show the variation of the posterior predictive distribution. The p-value on the median is close to 0.5, thereby quantitatively corroborating the close similarity

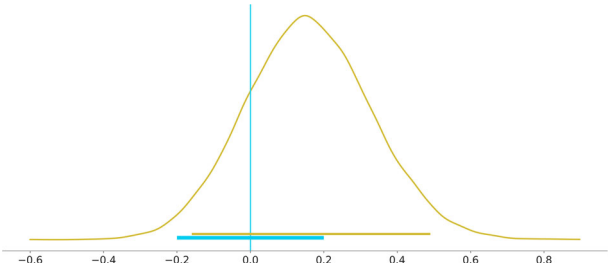

Fig. 6 Typical example of the sampling distribution of effect size as defined by Cohen's $d = \frac{\overline{x}}{s}$ with $\overline{x}$ the sample mean, s the sample standard deviation and $n = 70$ the sample size. The horizontal light blue line shows the ROPE, defined around the null value from -0.2 to 0.2. The 94% HDI (horizontal olive line segment) extends from 0.16 to 0.49

literature (Cohen, 1992, p.157) and Sawilowsky (2009) in the sense that effect sizes within this interval are generally considered too small to be interesting.

The Bayesian t-tests on the effect size on each of the $\frac{20\times(20-1)}{2} = 190$ pairs of ABoxes establishes relations among the ABoxes. For a given pair of different ABoxes (a, b), we allow four different sorts of relations:

1. a and b are indistinguishable, because the HDI lies completely inside the ROPE,
2. a and b are strictly ordered, because the HDI lies completely outside the ROPE,
3. a and b are non-strictly ordered, because the HDI partly overlaps with the ROPE and extends beyond it on one side only,
4. inconclusive, because the ROPE lies inside the HDI or because the HDI covers the ROPE and extends on either side.

Note that such ordering relations are based on the empirical data, but are numerically approximated as required by the Bayesian paradigm. It proves very difficult to cast

Table 1 The definitions of the complexity measures. In the tableau proof system of Fig. 2 of an ABox's inconsistency)

Complexity measure	Definition
$<_{mean}$	mean simulated inference time
$<_{least}$	mean simulated inference time of the fastest run
$<_{size}$	sum of the sizes of the formulas
$<_{quan}$	number of (universal or existential) restrictions
$<_{card}$	number of elements in the canonical model
$<_{edge}$	number of edges in the canonical model
$<_{pl}$	number of inference steps of the smallest proof
$<_{ps}$	total size of all expressions in a minimal proof
$<_{pd}$	depth of a minimal proof if it is viewed as a tree

these ordered pairs into one ordering on the ABoxes that is easy to interpret or visualise; more specifically, transitivity is difficult to guarantee. Hence, for allowing proper comparison a quantitative method is used that is described later.

4.2.1 Complexity Orderings

Regarding the simulated data from SHARP, we can define two different complexity orderings, based on:

- the mean simulated inference time ($<_{mean}$), and
- the mean simulated inference time of the fastest run ($<_{least}$).

The definitions of the complexity measures are summarised in Table 1. Table 2 shows how these two complexity orderings compare with the other complexity orderings found in the literature and mentioned in section 2.3. In more detail, the ordering $<_{size}$ is based on the sum of the sizes of the formulas in the ABox, where the size of a formula is defined as:

Definition 1 The *size* of an $\mathcal{ALE}$ formula is defined recursively as:

- $\mathsf{size}(C) = 1$ if C is a primitive concept symbol
- $\mathsf{size}(C_1 \sqcap C_2) = 1 + \mathsf{size}(C_1) + \mathsf{size}(C_2)$
- $\mathsf{size}(C) = 1 + \mathsf{size}(D)$ if $C = \neg D$ or $C = \exists r.D$ or $C = \forall r.D$ for some r.

The complexity ordering $<_{pl}$ is defined as the ordering entailed by the proof length, i.e. the number of inference steps of the smallest proof (in the tableau proof system of figure 2) of an ABox's inconsistency; note that in this proof system, the proof length of a consistent ABox equals the number of steps it takes to complete the ABox. The complexity ordering $<_{ps}$ denotes the ordering based on the total of the sizes of all formulas that appear in a (minimal) proof in the tableau proof system. The complexity ordering $<_{pd}$ is based on proof depth which is the depth of a minimal proof if it is viewed as a tree.

Table 2 The complexity orderings (shown by increasing complexity) entailed by various complexity measures. Horizontal lines denote borders between complexity ranks. The left two are based on SHARP's simulations and are the most fine-grained of the measures as no three ABoxes are placed in the same complexity category

$<_{mean}$	$<_{least}$	$<_{size}$	$<_{quan}$	$<_{card}$	$<_{edge}$	$<_{pl}$	$<_{ps}$	$<_{pd}$
$\mathcal{A}_2$	$\mathcal{A}_2$	$\mathcal{A}_{nin0}$	$\mathcal{A}_{end0}$	$\mathcal{A}_{nse0}$	$\mathcal{A}_2$	$\mathcal{A}_2$	$\mathcal{A}_{nin0}$	$\mathcal{A}_2$
$\mathcal{A}_{end0}$	$\mathcal{A}_{end0}$	$\mathcal{A}_{ts0}$	$\mathcal{A}_{end1}$	$\mathcal{A}_{nse1}$	$\mathcal{A}_{end0}$	$\mathcal{A}_{end0}$	$\mathcal{A}_{nin1}$	$\mathcal{A}_{end0}$
$\mathcal{A}_{end1}$	$\mathcal{A}_{end1}$	$\mathcal{A}_2$	$\mathcal{A}_{oin0}$	$\mathcal{A}_{spr0}$	$\mathcal{A}_{end1}$	$\mathcal{A}_{end1}$	$\mathcal{A}_{rnd0}$	$\mathcal{A}_{end1}$
$\mathcal{A}_{oin0}$	$\mathcal{A}_{oin0}$	$\mathcal{A}_{end0}$	$\mathcal{A}_{oin1}$	$\mathcal{A}_{oin0}$	$\mathcal{A}_{oin0}$	$\mathcal{A}_{nin0}$	$\mathcal{A}_2$	$\mathcal{A}_{nin0}$
$\mathcal{A}_{oin1}$	$\mathcal{A}_{oin1}$	$\mathcal{A}_{end1}$	$\mathcal{A}_{nin0}$	$\mathcal{A}_{oin1}$	$\mathcal{A}_{oin1}$	$\mathcal{A}_{nin1}$	$\mathcal{A}_{end0}$	$\mathcal{A}_{nin1}$
$\mathcal{A}_{nin0}$	$\mathcal{A}_{nin0}$	$\mathcal{A}_{nin1}$	$\mathcal{A}_{nin1}$	$\mathcal{A}_{nin0}$	$\mathcal{A}_{nin0}$	$\mathcal{A}_{oin0}$	$\mathcal{A}_{end1}$	$\mathcal{A}_{oin0}$
$\mathcal{A}_{nin1}$	$\mathcal{A}_{nin1}$	$\mathcal{A}_{rnd0}$	$\mathcal{A}_{nse0}$	$\mathcal{A}_{nin1}$	$\mathcal{A}_{nin1}$	$\mathcal{A}_{oin1}$	$\mathcal{A}_{ts0}$	$\mathcal{A}_{oin1}$
$\mathcal{A}_{nse0}$	$\mathcal{A}_{nse0}$	$\mathcal{A}_{rnd1}$	$\mathcal{A}_{nse1}$	$\mathcal{A}_2$	$\mathcal{A}_{nse0}$	$\mathcal{A}_{rnd0}$	$\mathcal{A}_{rnd1}$	$\mathcal{A}_{rnd0}$
$\mathcal{A}_{nse1}$	$\mathcal{A}_{spr1}$	$\mathcal{A}_{oin0}$	$\mathcal{A}_{spr1}$	$\mathcal{A}_{end0}$	$\mathcal{A}_{nse1}$	$\mathcal{A}_{spr1}$	$\mathcal{A}_8$	$\mathcal{A}_{spr1}$
$\mathcal{A}_{ts0}$	$\mathcal{A}_{nse1}$	$\mathcal{A}_{oin1}$	$\mathcal{A}_{spr0}$	$\mathcal{A}_{end1}$	$\mathcal{A}_{spr0}$	$\mathcal{A}_{rnd1}$	$\mathcal{A}_{oin0}$	$\mathcal{A}_{ts0}$
$\mathcal{A}_{rnd0}$	$\mathcal{A}_{rnd0}$	$\mathcal{A}_8$	$\mathcal{A}_{17}$	$\mathcal{A}_{ts0}$	$\mathcal{A}_{spr1}$	$\mathcal{A}_{nse0}$	$\mathcal{A}_{oin1}$	$\mathcal{A}_{rnd1}$
$\mathcal{A}_{spr1}$	$\mathcal{A}_{ts0}$	$\mathcal{A}_{ts1}$	$\mathcal{A}_2$	$\mathcal{A}_{spr1}$	$\mathcal{A}_{17}$	$\mathcal{A}_{nse1}$	$\mathcal{A}_{17}$	$\mathcal{A}_{17}$
$\mathcal{A}_{spr0}$	$\mathcal{A}_8$	$\mathcal{A}_{ts2}$	$\mathcal{A}_{ts0}$	$\mathcal{A}_{17}$	$\mathcal{A}_{rnd0}$	$\mathcal{A}_{ts0}$	$\mathcal{A}_{ts1}$	$\mathcal{A}_{nse0}$
$\mathcal{A}_{rnd1}$	$\mathcal{A}_{rnd1}$	$\mathcal{A}_{nse0}$	$\mathcal{A}_{rnd0}$	$\mathcal{A}_{rnd0}$	$\mathcal{A}_{rnd1}$	$\mathcal{A}_{spr0}$	$\mathcal{A}_{spr1}$	$\mathcal{A}_{nse1}$
$\mathcal{A}_8$	$\mathcal{A}_{spr0}$	$\mathcal{A}_{nse1}$	$\mathcal{A}_{rnd1}$	$\mathcal{A}_{rnd1}$	$\mathcal{A}_{ts0}$	$\mathcal{A}_8$	$\mathcal{A}_{nse0}$	$\mathcal{A}_8$
$\mathcal{A}_{ts1}$	$\mathcal{A}_{ts1}$	$\mathcal{A}_{ts3}$	$\mathcal{A}_{ts1}$	$\mathcal{A}_{ts1}$	$\mathcal{A}_8$	$\mathcal{A}_{17}$	$\mathcal{A}_{nse1}$	$\mathcal{A}_{spr0}$
$\mathcal{A}_{17}$	$\mathcal{A}_{17}$	$\mathcal{A}_{spr0}$	$\mathcal{A}_8$	$\mathcal{A}_8$	$\mathcal{A}_{ts1}$	$\mathcal{A}_{ts1}$	$\mathcal{A}_{spr0}$	$\mathcal{A}_{ts1}$
$\mathcal{A}_{ts2}$	$\mathcal{A}_{ts2}$	$\mathcal{A}_{ts4}$	$\mathcal{A}_{ts2}$	$\mathcal{A}_{ts2}$	$\mathcal{A}_{ts2}$	$\mathcal{A}_{ts2}$	$\mathcal{A}_{ts2}$	$\mathcal{A}_{ts2}$
$\mathcal{A}_{ts3}$	$\mathcal{A}_{ts3}$	$\mathcal{A}_{spr1}$	$\mathcal{A}_{ts3}$	$\mathcal{A}_{ts3}$	$\mathcal{A}_{ts3}$	$\mathcal{A}_{ts3}$	$\mathcal{A}_{ts3}$	$\mathcal{A}_{ts3}$
$\mathcal{A}_{ts4}$	$\mathcal{A}_{ts4}$	$\mathcal{A}_{17}$	$\mathcal{A}_{ts4}$	$\mathcal{A}_{ts4}$	$\mathcal{A}_{ts4}$	$\mathcal{A}_{ts4}$	$\mathcal{A}_{ts4}$	$\mathcal{A}_{ts4}$

The table shows that some ABoxes have the same associated inference time according to SHARP's simulations, based on frequentist independent t-tests. These ABoxes are considered to have indistinguishable levels of complexity and are therefore assigned the same rank. The first two orderings were established after 50 simulations on each ABox and are very similar, with a Spearman rank correlation of 99.3%. In fact, only the ABoxes $\mathcal{A}_{spr1}$, $\mathcal{A}_8$ and $\mathcal{A}_{rnd0}$ are higher up in $<_{least}$ compared to $<_{mean}$, because the first two allow for faster runs by skipping formulas that are not necessary for deriving the clash and the last allows for inspecting the universal restriction multiple times. Changing the sub-symbolic parameters of ACT-R changes the exact values of the simulated inference times, but it does not seem to change the orderings.

The other complexity orderings are less fine-grained than the first two, taking more ABoxes together in the same rank of the ordering. We expect the SHARP-based complexity orderings to correlate strongly to the order of complexity induced by the empirically established inference times.

4.2.2 Analysis

We use the following definition to compare the predicted ordering with the ordering found in the empirical data.

Definition 2 An *ordered-pair dataset* is a triple $\mathcal{D} = (D_i, D_s, D_n)$ of mutually exclusive sets of ordered pairs: D_i denotes a set of indistinguishable ordered pairs, D_s

denotes a set of strictly ordered pairs and D_n denotes a set of non-strictly ordered pairs. In case $D_n = \emptyset$ the ordered-pair dataset is *strict*, otherwise it is non-strict.

The sets D_i, D_s and D_n respectively correspond to the judgements 1, 2 and 3 on the empirical data, as discussed on page 19.

To assess the effectiveness of the linear preorderings (rankings) based on SHARP's simulated data in predicting the empirically established ordered-pair dataset we use the *pairwise correlation*. This correlation is an extension of the *hindsight accuracy*, defined in (Cameron et al., 2021).

Definition 3 Let $\mathcal{L} = (L, \leq_L)$ be a linear preorder and let $\mathcal{D} = (D_i, D_s, D_n)$ be an ordered-pair dataset on L. The *pairwise correlation*, $\mathrm{C}_p(\mathcal{L}, \mathcal{D})$ is the percentage $\frac{2n-N}{N}$, where n is the number of pairs (a, b) for which:

- $a \approx_L b$ and $(a, b) \in D_i$, or
- $a <_L b$ and $(a, b) \in D_s$, or
- $a <_L b$ and $(a, b) \in D_n$, or
- $a \approx_L b$ and $(a, b) \in D_n$, or
- $a \approx_L b$ and $(b, a) \in D_n$,

and $N = |\cup \{D_i, D_s, D_n\}|$.

In this definition N is the number of ordered pairs for which the empirical data determines an ordering, i.e. excluding the inconclusive pairs, while n is the number of such pairs that agree with the linear preorder $\mathcal{L}$. The pairwise correlation can be straightforwardly understood as the percentage of pairs for which the linear preorder $\mathcal{L}$ agrees with the ordered-pair dataset $\mathcal{D}$; it assumes values between -100% and 100%, in which case no or, respectively, all ordered pairs from $\mathcal{D}$ agree with $\mathcal{L}$.

4.2.3 Results

Regarding the Bayesian analysis on the empirical data, from all 190 ordered pairs of ABoxes there are 88 for which the HDI lies completely outside the ROPE (strict) and 82 for which the HDI partly overlaps with the ROPE and extends to only one side of it (non-strict). The remaining 20 cases were inconclusive.

Table 3 shows how the orderings based on SHARP's simulations compare to orderings based on the other complexity measures from table 2. Both measures $<_{mean}$ and $<_{least}$, based on SHARP, outperform all other measures on the data for which the ABoxes' complexities were most easily distinguishable, i.e. the strict data.

For reference, a comparison is shown that includes both the strict and non-strict empirical data; this is called the 'lenient' case. On the latter, the ordering $<_{pl}$ has the highest pairwise correlation with the SHARP-based orderings on second and third place.

The worst correlating ordering in both cases is $<_{ps}$ with only 29.5% and 43.5% pairwise correlation for the strict and lenient empirical ordering respectively.

Table 3 Pairwise correlations of the different complexity measures with the empirical data (both the consistent and inconsistent ABoxes together)

Data	$<_{mean}$	$<_{least}$	$<_{size}$	$<_{quan}$	$<_{card}$	$<_{edge}$	$<_{pl}$	$<_{ps}$	$<_{pd}$
Strict	97.7%	93.2%	34.1%	52.3%	93.2%	52.3%	84.1%	29.5%	59.1%
Lenient (Strict and Non-strict)	78.8%	74.1%	48.2%	67.1%	71.8%	68.2%	81.2%	43.5%	65.9%

4.2.4 Discussion

SHARP's simulations result in more accurate complexity orderings in comparison to competing methods of establishing complexity orderings.

On a critical note, only the ordering $<_{pl}$ outperforms the SHARP-based orderings regarding the empirical data that includes both the strict and non-strict cases – which we call the lenient case –, but its value is inflated as a consequence of the following. The empirical data happens to have many instances of ABoxes for which the complexities could only be distinguished non-strictly. By the definition of the pairwise correlation this situation 'rewards' complexity measures that show many instances of ABoxes having the same complexity rank – which $<_{pl}$ indeed does – because ABox pairs with the same rank i.e. $a \approx_L b$ will count for either $(a, b) \in D_i$ or $(b, a) \in D_i$ and can thus not be shown to be wrong. On the other hand, complexity measures that differentiate between many complexity levels, such as $<_{mean}$ and $<_{least}$, can more easily be proven to be wrong. This means that, given the data, the pairwise correlation of $<_{pl}$ is inflated relative to $<_{mean}$ and $<_{least}$. It is worth pointing out that the strict data most clearly empirically distinguishes different complexity levels and is hence the most important kind of data to compare complexity measures on.

For the orderings $<_{mean}$ and $<_{least}$, comparing the predicted orderings with the empirical data, shows that the ABoxes $\mathcal{A}_{nin0}$ and $\mathcal{A}_{nin1}$, which are both consistent, are predicted to be more complex than they empirically turn out to be. This is related to the simulated inference time being relatively small for these ABoxes, which causes the time overhead by component five in SHARP to have a relatively large impact on the total inference time. As mentioned in Section 2.5.2, this is an implementational issue that was necessitated by ACT-R's design and not an intended feature.

To assess the orderings in more detail, we create six different ordered-pair datasets by varying two characteristics:

- by either including the 82 non-strictly ordered pairs (creating a lenient ordered-pair dataset) or by excluding them (creating a strict ordered-pair dataset);
- by observing the data of the 10 consistent and 10 inconsistent ABoxes separately, or by observing the data for all 20 together.

The ensuing ordered-pair datasets do not describe unique linear preorders. Table 4 shows the pairwise correlation of the orderings based on SHARP. The table shows that $<_{mean}$ has a higher pairwise correlation than $<_{least}$ in all cases. Also, both orderings show greater pairwise correlation to the strict dataset than to the lenient one, which indicates that both predicted orderings best capture those pairs that are empirically most easily distinguished. Furthermore, $<_{mean}$ performs better in the cases where it is compared to the consistent and inconsistent ABoxes separately than when it is compared to both simultaneously. The ordering $<_{least}$, however, has a lower pairwise correlation to the inconsistent ABoxes than to the consistent ABoxes, while its pairwise correlation to both together lies between the two.

The fact that the $<_{mean}$-ordering performs better than the $<_{least}$-ordering might indicate that, in a simulation, skipping formulas that are not necessary for deriving a clash gives a less accurate prediction than treating these formulas and the other ones equally. This indicates that the subjects, on average, appear to have no special heuristic

Table 4 Pairwise correlations of $<_{mean}$ and $<_{least}$ for various sections of the empirical data

	Strict			Lenient (Strict and Non-strict)		
Ordering	Consistent	Inconsistent	Both	Consistent	Inconsistent	Both
$<_{mean}$	100%	100%	97,7%	89,5%	82,9%	78,8%
$<_{least}$	100%	84.6%	93.2%	89,5%	71,4%	74,1%

for selecting only those formulas that give rise to a clash; similar to SHARP they take all formulas into account in deciding the consistency of the ABox. This supports the idea that SHARP can be used as an estimate for the relative cognitive complexity for a task even when there are multiple solving strategies available.

The order of consistent ABoxes is generally predicted more accurately than the order of inconsistent ABoxes. This indicates that the thinking processes of the subjects searching for an inconsistency are likely based on something different than merely applying the syntax expansion rules to derive a clash: some inconsistencies are more difficult compared to this method, while others are easier.

4.3 Experienced Difficulty

We are interested in how humans experience the difficulty of the ABoxes and whether this correlates with their inference time. We expect high correlation between the two, as we consider them to be two operational aspects of the same concept. In case the two do not correlate at all, both are likely bad operational definitions of the concept of cognitive complexity of the $\mathcal{ALE}$ ABox consistency task.

4.3.1 Analysis

To gain insight into how well humans' own judgements about the difficulty of an ABox correlate with their inference time, we compute the Pearson correlation between the rated difficulty scores of a subject for an ABox and that subject's inference time for the ABox. It is also interesting to see how the judgement of a group of subjects correlates to their inference time. To get insight in this, we compute the Spearman rank correlation between the orderings induced by the mean difficulty score and the mean inference time. Moreover, we compute the pairwise correlation for the ordering based on mean difficulty score and the orderings established by Bayesian t-tests on effect size.

4.3.2 Results

Figure 7 shows a scatter plot of the subjects' difficulty rating score and their inference time. The Pearson correlation coefficient is 0.59. This means that the participants are moderately good at rating the difficulty of ABoxes. Note the fact that the five-point rating scale is coarse in comparison to the continuous scale inference time.

The Spearman rank correlation between the ordering based on mean difficulty score and the ordering based on mean inference time is: 78%. The pairwise correlation of

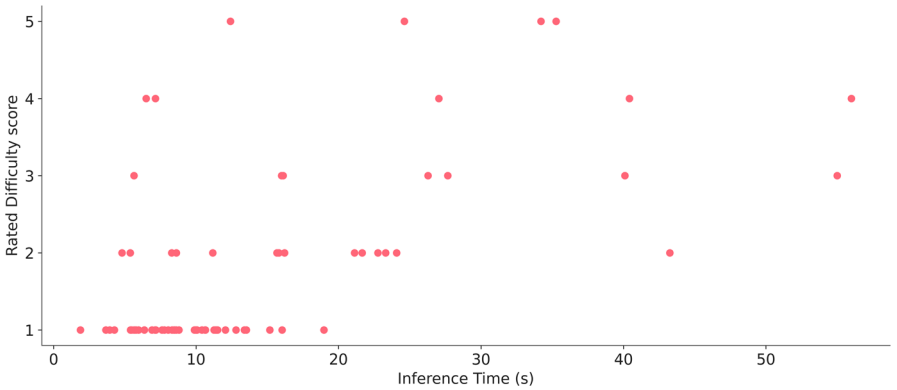

Fig. 7 Difficulty rating score plotted against empirically measured inference time

the mean rated difficulty (which is a linear preorder) is 95.5% and 80.0% for the strict and non-strict ordering respectively.

4.3.3 Discussion

The five-point scale is a rather coarse measure with which to rate the difficulty of ABoxes. On an individual level, only moderate correlation exists between one's difficulty score and one's inference time, but this might be just because of the coarseness of the rating scale. The correlation becomes stronger if the mean rating score of all subjects is used. The pairwise correlation is almost perfect with the strict ordering, which means that those ABoxes which are easiest to differentiate empirically by inference times are also the easiest to rate the difficulty of. The above considerations indicate that both the inference time and the difficulty rating score are likely converging operational definitions of the same concept.

5 Conclusion

The debugging of ontologies is a cognitively straining and error-prone task for humans. There exists, therefore, a high demand for facilitating this task and various approaches to achieve this can be found in the literature. Among other things, there is a demand for a complexity measure on proofs and justifications that corresponds to the cognitive complexity with which humans process them. Although cognitive complexity does not have an established conceptual definition, operational aspects are the error rate, the inference time, the rated difficulty, to name a few. The latter two are used in this paper to operationally express the cognitive complexity of the $\mathcal{ALE}$ ABox consistency task. We model the task with a cognitive architecture by implementing the abstract ABox consistent() algorithm for the description logic $\mathcal{ALE}$ into ACT-R. This is, to the authors' knowledge, the first attempt to apply a cognitive architecture to model deductive symbolic reasoning of a description logic. As such, it is an improvement over other attempts in defining complexity measures that either do not involve cognitive

theory at all, or make only selective use of cognitive theory. Written in ACT-R, SHARP models a plausible reasoning process that, by design, includes known cognitive effects related to learning, forgetting and retrieving information from memory in an integrated manner. The approach encapsulates the straightforward idea of simulating the ABox consistent() algorithm as if it were run by a human mind. By simulation, SHARP allows the prediction of the inference time of a human performing the $\mathcal{ALE}$ ABox consistency task. The simulated inference times show patterns that give rise to concrete hypotheses on human performance on the task. These hypotheses were empirically tested in this paper.

The empirical inference times generally show much larger variance than the simulated inference times, making it difficult to either refute or accept the model. The hypotheses about Role Name Dependence and Nesting Depth Dependence were rejected, showing that the way humans perform the ABox consistency task is different from executing the syntax expansion rules to derive a clash. Most hypotheses, however, allowed for both accepting and rejecting the model, except in the case of the hypothesis about the predicted ordering of the cognitive complexity of the ABoxes. The latter hypothesis was strongly supported by the empirical data with the ordering of ABoxes that are empirically strongly different being most accurately simulated by SHARP. Moreover, the hypothesis that the rated difficulty score correlates strongly with the inference time showed moderate support, the conclusion being that both inference time and rated difficulty are useful operational aspects of the concept of cognitive complexity. The above indicates that cognitive complexity of the ABox consistency task can be predicted and can be so by a relatively straightforward but novel idea, namely by implementing a known abstract algorithm in a cognitive architecture. The same approach can be taken to model human performance on legal reasoning, medical diagnosis, programming, or indeed any task that can be solved by an algorithm.

This paper supports the idea that defining a complexity measure for the ABox consistency task that corresponds to cognitive performance is possible. The complexity measures defined in this paper are adequate for the relative cognitive complexity of the ABoxes, despite the absolute inference times output by SHARP lie far away from the empirical inference times. From a modelling perspective, it is interesting and desirable to improve SHARP's performance and this can be done in a number of ways. Firstly, an additional visual component can be added that simulates the processing of the $\mathcal{ALE}$ ABox formulas. Secondly, the sub-symbolic parameters can be tweaked to make the simulated output fit the empirical data better. Thirdly, it is desirable to remove the overhead inference time created by component five of SHARP, as it is responsible for unwanted simulation results. Fourthly, ACT-R's partial matching can be invoked to more realistically model errors; this is important, because the error rate is another operational aspect of cognitive complexity. Fifthly, SHARP can be extended to incorporate reasoning about TBox axioms. Finally, and very relevantly, extensions to more complex logics can be made.

The latter point is an important one, because even though the complexity measures defined in this paper allow the selection of a justification with the least cognitive complexity, thereby facilitating debugging, the scope of application is rather limited. The reason is that the complexity measures only work for justifications of inconsistencies. Justifications of consistent conclusions generally lie beyond the scope, as the stan-

dard method of checking logical consequence by adding the negated conclusion to the premises and inspecting the consistency of the resulting set is not guaranteed to work here, because the description logic $\mathcal{ALE}$ does not have full negation. Extension to a logic with full negation seems, therefore, necessary for applying the complexity measures to justifications in general and thereby also to logical consequences and proofs. The low cognitive effort needed to understand such optimised justifications and proofs should help the ontology engineer who is working with the software find the root cause of the incorrect conclusion in the least amount of time, thereby facilitating the debugging process. Note that the absolute values of the complexity measures are not important here, rather the adequacy of the complexity ordering entailed by the measures is relevant, which is indeed high for our data. Cases where the complexity measures disagree with the empirical data usually involve ABoxes that have similar cognitive complexity, in which case selecting one or the other should arguably not result in meaningfully different levels of performance.

One practical limitation to applying these complexity measures is that they are computationally expensive. To mitigate this problem surrogate modelling seems promising. A surrogate model is a model with roughly the same input-output function as SHARP which requires less computational power. Different approaches exist; in particular random forests, symbolic regression and neural networks seem useful. These lines of research will be pursued in future.

Appendix: A ABoxes used

The following ABoxes were used in the experiment, an asterisk indicates that the ABox is inconsistent.

- $\mathcal{A}_{practice0} = \{a : A,\ b : \neg A,\ a : \neg B\}$
- $\mathcal{A}^*_{practice1} = \{a : \forall s.(\neg A \sqcap B),\ (b, a) : r,\ c : (\neg B \sqcap C),\ b : \forall r.\exists s.(A \sqcap B),\ a : (B \sqcap \neg C),\ (b, c) : s\}$
- $\mathcal{A}^*_{end0} = \{a : A,\ a : B,\ a : \neg A\}$
- $\mathcal{A}^*_{end1} = \{a : A,\ b : B,\ a : \neg A\}$
- $\mathcal{A}_{rnd0} = \{a : \exists r.A,\ a : \forall s.\neg A,\ a : \exists r.B\}$
- $\mathcal{A}_{rnd1} = \{a : \exists r.A,\ a : \forall s.\neg A,\ a : \exists s.B\}$
- $\mathcal{A}^*_{oin0} = \{a : A \sqcap B,\ a : \neg B\}$
- $\mathcal{A}^*_{oin1} = \{a : \neg B,\ a : A \sqcap B\}$
- $\mathcal{A}_{nin0} = \{a : A,\ a : B\}$
- $\mathcal{A}_{nin1} = \{a : \neg A,\ a : \neg B\}$
- $\mathcal{A}^*_{nse0} = \{a : (A \sqcap (\neg A \sqcap (B \sqcap C)))\}$
- $\mathcal{A}^*_{nse1} = \{a : (B \sqcap (C \sqcap (\neg A \sqcap A)))\}$
- $\mathcal{A}_{ts0} = \{a : \exists r.A \sqcap \exists r.B\}$
- $\mathcal{A}_{ts1} = \{a : \exists r.(\exists r.A \sqcap \exists r.B) \sqcap \exists r.B\}$
- $\mathcal{A}_{ts2} = \{a : (\exists r.A \sqcap \exists r.B) \sqcap \forall r.(\exists r.A \sqcap \exists r.B)\}$
- $\mathcal{A}_{ts3} = \{a : (\exists r.A \sqcap \exists r.B) \sqcap \forall r.(\exists r.(\exists r.A \sqcap \exists r.B) \sqcap \exists r.B)\}$
- $\mathcal{A}_{ts4} = \{a : ((\exists r.A \sqcap \exists r.B) \sqcap \forall r.((\exists r.A \sqcap \exists r.B) \sqcap \forall r.(\exists r.A \sqcap \exists r.B)))\}$
- $\mathcal{A}^*_{spr0} = \{a : (A \sqcap (B \sqcap (C \sqcap (D \sqcap \neg A))))\}$

- $\mathcal{A}^*_{spr1} = \{a : (A \sqcap B),\ a : (B \sqcap C),\ a : (C \sqcap D),\ a : (D \sqcap \neg A)\}$
- $\mathcal{A}_{17} = \{a : (\neg A \sqcap (B \sqcap \neg C)),\ b : (\neg B \sqcap (C \sqcap A)),\ b : (A \sqcap C),\ a : B\}$
- $\mathcal{A}^*_2 = \{a : A,\ a : \neg A,\ b : \forall r.A\}$
- $\mathcal{A}^*_8 = \{a : \exists r.\exists s.\neg A,\ a : \forall r.\forall s.A,\ b : \exists s.\neg B\}$

Appendix: B Some peculiar effects of SHARP

Although the complexity measures derived from SHARP capture the overall relative cognitive complexity well, SHARP displays some peculiar effects. This section discusses these effects by first describing them, analysing their associated data and finally discussing the found results, leading to the conclusion that, on a fine-grained scale, SHARP leaves some room for improvement.

Element Name Independence
The simulated inference times for deciding the consistency of

$\mathcal{A}_{end0} = \{a : A,\ a : B,\ a : \neg A\}$ and
$\mathcal{A}_{end1} = \{a : A,\ b : B,\ a : \neg A\}$

are identical. These ABoxes are designed to be the same except for the element name of the second formula. This causes the two ABoxes to describe different models, but SHARP shows the *same* simulated inference times for both cases. The reason is that the number of unique atomic formulas to be inspected by SHARP's first component is the same in both cases. The fact that the exact formulas are different in both ABoxes is irrelevant, as they are treated in the same way formally.

Role Name Dependence
The simulated inference time of deciding the consistency of

$\mathcal{A}_{rnd0} = \{a : \exists r.A,\ a : \forall s.\neg A,\ a : \exists r.B\}$ is shorter than
$\mathcal{A}_{rnd1} = \{a : \exists r.A,\ a : \forall s.\neg A,\ a : \exists s.B\}$.

The ABoxes are similar, except that the role name of their last formula is different. This causes the decision procedure to make an inference on a universal restriction in the second case, but not in the first, thereby requiring *more* time.

Formula Order Independence
The simulated inference times for deciding consistency of

$\mathcal{A}_{oin0} = \{a : A \sqcap B,\ a : \neg B\}$ and
$\mathcal{A}_{oin1} = \{a : \neg B,\ a : A \sqcap B\}$

are identical. These ABoxes are the same, except that the order of their formulas is different. SHARP makes no distinction between the two cases, as all formulas of the input ABox are loaded to declarative memory simultaneously. The order in which the formulas are presented, therefore, does not make a difference and the simulated inference times are the *same*.

Negation Independence
The simulated inference times for deciding consistency of

$\mathcal{A}_{nin0} = \{a : A,\ a : B\}$ and
$\mathcal{A}_{nin1} = \{a : \neg A,\ a : \neg B\}$

are identical. The second ABox differs from the first in that its concepts are negated. SHARP lacks simulations of visual processes and makes no distinction between negated and non-negated concepts, so the simulated inference times of these ABoxes the *same*.

Nesting Depth Dependence

To decide the consistency of

$\mathcal{A}_{nse0} = \{a : (A \sqcap (\neg A \sqcap (B \sqcap C)))\}$ requires less simulated inference time than
$\mathcal{A}_{nse1} = \{a : (B \sqcap (C \sqcap (\neg A \sqcap A)))\}$.

By applying the syntax expansion rules, SHARP unpacks nested conjunctions from the outside inwards. Clashing concepts that are more deeply nested are therefore found later than clashing concepts that are less deeply nested. The ABoxes are designed to differ only in the nesting depth of the clashing concepts, so that SHARP shows a *longer* simulated inference times for the second ABox.

B.1 Results

Denoting the empirical inference time associated to ABox $\mathcal{A}$ by $t(\mathcal{A})$ and using a 94% HDI[4], we can conclude that:

1. $t(\mathcal{A}_{end0}) \leq t(\mathcal{A}_{end1})$, thereby *neither rejecting nor accepting* the hypothesis of Element Name Independence.
2. $t(\mathcal{A}_{rnd1}) \leq t(\mathcal{A}_{rnd0})$, thereby *rejecting* the hypothesis of Role Name Dependence.
3. $t(\mathcal{A}_{oin0}) \leq t(\mathcal{A}_{oin1})$, thereby *neither rejecting nor accepting* the hypothesis of Formula Order Independence.
4. $t(\mathcal{A}_{nin0}) \leq t(\mathcal{A}_{nin1})$, thereby *neither rejecting nor accepting* the hypothesis of Negation Independence.
5. $t(\mathcal{A}_{nse1}) \leq t(\mathcal{A}_{nse0})$, thereby *rejecting* the hypothesis of Nesting Depth Dependence.

B.2 Discussion

Two hypotheses are rejected, while the others are neither accepted nor rejected because of high variance of the empirical data. The hypothesis about Element Name Independence is partially supported by the empirical data, although there is also room for supporting the alternative hypothesis that the inference time of $\mathcal{A}_{end1}$ is greater than the inference time of $\mathcal{A}_{end0}$. In case of the latter, the subjects do something different than applying the syntax expansion rules like in the ABox consistency algorithm. They might be creating models in their minds, and because models that consist of more elements are harder, they require more inference time. Alternatively, the longer inference

[4] The 94% HDI is the default in Arviz (Devs, 2024), which is the package that was used for making most plots in this paper. It is introduced as a 'friendly remainder [sic] of the arbitrary nature of the 95% value.' (Martin, 2018).

time might be explained by it being visually more straining to process different symbols, rather than the same symbol multiple times. Of course many more strategies can explain the effects. While the data does not distinguish between the null hypothesis and the hypothesis that the inference time of $\mathcal{A}_{end1}$ is greater than the inference time of $\mathcal{A}_{end0}$, it does exclude the hypothesis that the inference time of $\mathcal{A}_{end1}$ is smaller than the inference time of $\mathcal{A}_{end0}$. This means that replacing an element name in a given ABox with an element name that is new for that ABox, most likely does not reduce the cognitive complexity of the ABox consistency task for that ABox.

The empirical data rejects the predicted hypothesis that the inference time of $\mathcal{A}_{rnd0}$ is smaller than the inference time of $\mathcal{A}_{rnd1}$. This is evidence that the subjects do not strictly follow the syntax expansion rules in performing the ABox consistency task. The empirical data supports two alternative scenarios, one in which the inference time for both ABoxes is the same, and one in which the inference time for $\mathcal{A}_{rnd1}$ is smaller. The second scenario is difficult to explain with mental models theory, as the mental model corresponding to $\mathcal{A}_{rnd1}$ is arguably the more complex one. Perhaps some subjects were spending extra time on the ABox $\mathcal{A}_{rnd0}$: the formula $a : \forall s.\neg A$ is in a sense irrelevant, because no inference rule can be applied to it and this might have confused the subjects.

The predicted hypothesis that the inference time of $\mathcal{A}_{oin0}$ is the same as the inference time of $\mathcal{A}_{oin1}$ does not contradict the empirical data. The empirical data, however, also supports the alternative hypothesis that the inference time of $\mathcal{A}_{oin1}$ is greater than the inference time of $\mathcal{A}_{oin0}$. If this were the case, the order in which the formulas are presented, even in relatively simple cases such as this, matters in the cognitive complexity of the ABox consistency task. The subjects might have had a preference for complex formulas coming first, or perhaps for having as little symbols as possible between clashing concept symbols.

Based on SHARP's simulation results, the inference times of $\mathcal{A}_{nin0}$ and $\mathcal{A}_{nin1}$ are predicted to be the same. The empirical data supports this scenario, but also supports the scenario in which the inference time of $\mathcal{A}_{nin1}$ is greater than the inference time of $\mathcal{A}_{nin0}$. If this were true, it would mean that negations add to the cognitive complexity of the ABox consistency task. Whether this increase in complexity is explained by the difficulty to imagine concept complements in mental models, or just because the extra negation symbols add to visual processing time is unclear.

The hypothesis that the inference time of $\mathcal{A}_{nse1}$ is greater than the inference time of $\mathcal{A}_{nse0}$ is rejected by the empirical data. Either the corresponding inference times are the same, or the inference time corresponding to $\mathcal{A}_{nse1}$ is smaller than the inference time corresponding to $\mathcal{A}_{nse0}$. The latter case can be explained by the fact that the clashing concepts visually appear closer together in $\{a : (B \sqcap (C \sqcap (\neg A \sqcap A)))\}$ than in $\{a : (A \sqcap (\neg A \sqcap (B \sqcap C)))\}$. Although the difference is very small, the subjects might recognise the visual pattern $\neg A \sqcap A$ very easily. An alternative explanation is that the subjects are using *deep inference rules* (Tubella & Strassburger, 2019) to draw conclusions; instead of working outside inwards, the more deeply nested connectives are made inferences on first. Mental model theory cannot be used to explain the difference, as the mental constructions are arguably the same: the element satisfies the exact same concepts in both cases. In any case, the data shows that the straightforward

idea of subjects applying the syntax expansion rules is too simplistic when differences of cognitive complexity are small.

Appendix: C Exponential Scaling of Inference Times

The size of an ABox is recursively defined as:

Definition 4 (Baader et al., 2017, p.58)**.** Given an $\mathcal{ALE}$ concept C, its *size* $\mathsf{size}(C)$ and set of *subconcepts* $\mathsf{sub}(C)$ are defined by recursion on the structure of C:

- If C is a concept name, then $\mathsf{size}(C) = 1$ and $\mathsf{sub}(C) = \{C\}$,
- If $C = C_1 \sqcap C_2$, then $\mathsf{size}(C) = 1 + \mathsf{size}(C_1) + \mathsf{size}(C_2)$ and $\mathsf{sub}(C) = \{C\} \cup \mathsf{sub}(C_1) \cup \mathsf{sub}(C_2)$,
- If $C = \neg D$ or $C = \exists r.D$ or $C = \forall r.D$, then $\mathsf{size}(C) = 1 + \mathsf{size}(D)$ and $\mathsf{sub}(C) = \{C\} \cup \mathsf{sub}(D)$.

These definitions extend to an ABox $\mathcal{A}$ as follows: $\mathsf{sub}(\mathcal{A}) = \bigcup_{a:C \in \mathcal{A}} \mathsf{sub}(C)$ and $\mathsf{size}(\mathcal{A}) = \sum_{a:C \in \mathcal{A}} \mathsf{size}(C) + R_{\mathcal{A}}$, where $R_{\mathcal{A}}$ is the number of role formulas (i.e. formulas of the form $(a, b) : r$) in $\mathcal{A}$.

The ABoxes:

- $\mathcal{A}_{ts0} = \{a : \exists r.A \sqcap \exists r.B\}$,
- $\mathcal{A}_{ts1} = \{a : \exists r.(\exists r.A \sqcap \exists r.B) \sqcap \exists r.B\}$,
- $\mathcal{A}_{ts2} = \{a : (\exists r.A \sqcap \exists r.B) \sqcap \forall r.(\exists r.A \sqcap \exists r.B)\}$,
- $\mathcal{A}_{ts3} = \{a : (\exists r.A \sqcap \exists r.B) \sqcap \forall r.(\exists r.(\exists r.A \sqcap \exists r.B) \sqcap \exists r.B)\}$,
- $\mathcal{A}_{ts4} = \{a : ((\exists r.A \sqcap \exists r.B) \sqcap \forall r.((\exists r.A \sqcap \exists r.B) \sqcap \forall r.(\exists r.A \sqcap \exists r.B)))\}$

grow linearly in size, but the corresponding expanded ABoxes grow exponentially in size due to AND-branching: the ABoxes encode binary trees and the size of those trees increases exponentially with their depth. The corresponding simulated inference times also scale exponentially with the size of the ABoxes.

The simulations of the ABox $\mathcal{A}_{ts4}$ are problematic: the long inference time for these simulations causes certain chunks to decrease their activation below the retrieval threshold; this problem was discussed in Section 2.5.2. SHARP fails to draw a conclusion in about 85% of the cases for this ABox; simulations for the other ABoxes do not suffer from this problem.

Learning effects in SHARP such as proceduralisation and activation base-level learning seem insufficient to make the simulated inference times of the ABox consistency problem scale better than exponentially. Due to the relatively simple structure of these ABoxes – they contain no negations – one may have reasons to doubt that this hypothesis holds.

C.1 Analysis

To test for this exponential scaling, two linear fits are performed on the empirical data: one on the raw data and one on the log-transformed data. The R^2 coefficients

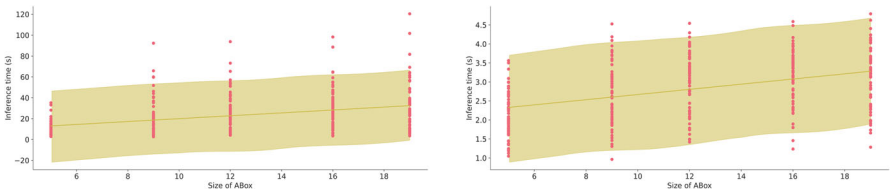

Fig. 8 The red points are the empirically measured inference times and the yellow lines linear regression lines. The left figure shows a linear regression on the size and inference time, resulting in $R^2 = 0.36$. The figure on the right uses log-transformed inference times, resulting in $R^2 = 0.38$. Both figures show the 94% predictive confidence interval

are determined for both regressions and are compared after performing Fisher's z-transformation to the R^2 coefficients. Fisher's z-transformation is defined by $z = \operatorname{arctanh}(r)$ and is usually used for mapping correlation coefficients from the interval $[-1, 1]$ to the real line. The skewness that distributions of such coefficients have because of the interval's boundaries, is removed by the transformation, making such distributions approximately normal. We therefore apply Fisher's z-transformation to the R^2 coefficients and assume they are normally distributed so that their difference can be tested for significance. If the z-value of the second regression is significantly higher (at alpha level 0.06 [5]) than the z-value of the first, it can be concluded that the data shows exponential scaling of inference time with ABox size.

C.2 Results

Figure 8 shows the two regressions and the 95% predictive interval. The linear regression on the raw data has an R^2-value of 0.36 ± 0.02. The regression on the log-transformed data has the slightly higher R^2-value of 0.38 ± 0.02. Performing Fisher's Z-transformation on the data gives a p-value of 0.83, which means that the null hypothesis is very likely and that the two regressions are not statistically significantly different.

C.3 Discussion

The exponential scaling of inference time with size cannot be demonstrated in the empirical data. There is a slight indication of exponential behaviour, but this is not statistically significant. This may indicate that the subjects find and use certain shortcuts in the reasoning process which speeds up the performance. Private and unprompted communication with some participants indicated that the absence of negations in the ABoxes led some subjects to decide their consistency very quickly; it is, however, not clear how many subjects used this shortcut.

[5] A similar derfault in Arviz Devs (2024) showing the arbitrary nature of the 0.05 value. Martin (2018).

Appendix: D Spread Dependence on Nesting Structure

- $\mathcal{A}_{spr0} = \{a : (A \sqcap (B \sqcap (C \sqcap (D \sqcap \neg A))))\}$ shows less spread in simulated inference times than
- $\mathcal{A}_{spr1} = \{a : (A \sqcap B),\ a : (B \sqcap C),\ a : (C \sqcap D),\ a : (D \sqcap \neg A)\}$.

These ABoxes are similar in that their single element satisfies the same concepts in both; they are different in the way these concepts are distributed over the conjunctions. The clashing concepts are A and $\neg A$, and the first ABox allows only one way to derive this clash. The second ABox allows multiple ways to derive the clash, because some formulas in it are not necessary for deriving the clash and can hence be skipped. Humans, focussing on finding a clash, might skip irrelevant formulas automatically and their inference times would therefore not show a difference in spread. Yet from SHARP, designed to apply the syntax expansion rules only, it follows that the inference times of $\mathcal{A}_{spr1}$ show a larger spread than the inference times of $\mathcal{A}_{spr0}$.

D.1 Analysis

To test this hypothesis, samples are drawn from the posterior distributions which are then compared with each other using the Brown-Forsythe test (Brown & Forsythe, 1974). The Brown-Forsythe test is designed for comparing variances of two samples and is robust to the underlying distributions possibly being asymmetric. It is defined by:

$$F = \frac{N-p}{p-1} \frac{\sum_{j=1}^{p} n_j (\bar{z}_j - \bar{\bar{z}})^2}{\sum_{j=1}^{p} \sum_{i=1}^{n_j} (z_{ij} - \bar{z}_j)^2},$$

where $p = 2$ denotes the number of groups, n_j the size of group j, $N = n_1 + n_2$ and $z_{ij} = |y_{ij} - \bar{y}_j|$ with $\bar{y}_j$ the median of group j. Furthermore, $\bar{z}_j$ is the mean value of z_{ij} for group j and $\bar{\bar{z}}$ is the overall mean of z_{ij}.

To perform Bayesian hypothesis testing on the resulting distribution of Brown-Forsythe values, we defined our ROPE to be the 95% confidence interval of the F-distribution with $p-1 = 1, N-p = 68$ degrees of freedom, which is the theoretically predicted distribution of Brown-Forsythe values under the hypothesis that the two samples have the same variance.

D.2 Results

Figure 9 shows the distribution of Brown-Forsythe values estimated by taking samples from the posterior predictive distribution. The ROPE lies completely within the HDI, so no meaningful conclusion can be drawn.

D.3 Discussion

The fact that the difference in variance of inference times for the two ABoxes cannot be empirically demonstrated might be an indication that the distributions of inference

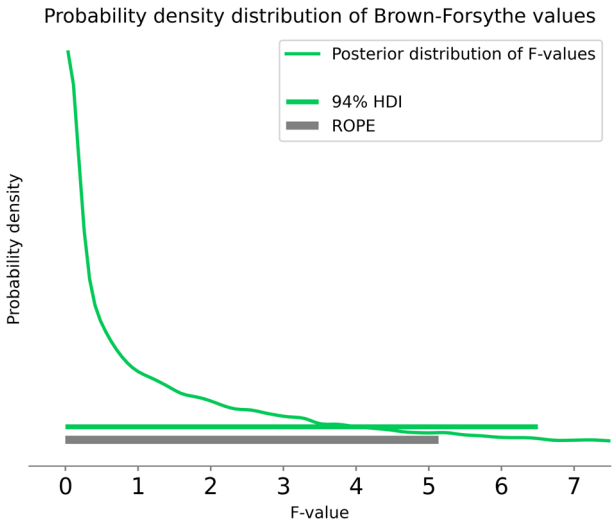

Fig. 9 The posterior predictive distribution of Brown-Forsythe values. The ROPE is shown as the horizontal grey line and is defined (based on theoretical considerations) to extend from 0.0 to 5.10. The horizontal green line is the 94% HDI from 0.0 to 6.5

times simulated by SHARP are too narrow to be accurately representing human performance. Indeed, SHARP uses fixed parameter settings, which means that the cognitive processes it simulates, correspond to cognitive processes of humans that are exactly identical in their cognitive performance. Group performance on the ABox consistency task therefore shows larger variance in the inference time. It is interesting for future research to investigate the simulated inference times of SHARP where each simulation uses a different parameter setting such that it approximates a group performance better.

Acknowledgements Both authors are indebted to Graham E. Leigh and the reviewer for their very helpful comments during the writing of this paper.

Author Contributions Both authors contributed to the study design and data collection. Data Analysis was performed by Jelle Tjeerd Fokkens, who also wrote the first draft. Both authors commented on previous versions of the manuscript. Both authors read and approved the final manuscript.

Funding Open access funding provided by University of Gothenburg. This research was carried out while both authors were employed by the University of Gothenburg. The second author was partially supported by the Swedish Research Council (Vetenskapsrådet), Grant No.2022-01685. Additionally, the authors received financial support through the Stiftelsen Karl Langenskiölds minnesfond, reference number KL2023-0010.

Data Availability The empirical data gathered for this study is freely accessible at: https://doi.org/10.5878/5739-da47

Declarations

Conflicts of Interest Not applicable as no conflict was mentioned.

Ethical Approval Not applicable.

References

Alrabbaa, C., Baader, F., Borgwardt, S., Koopmann, P., & Kovtunova, A. (2020). On the complexity of finding good proofs for description logic entailments. In: Proceedings of the 33rd International Workshop on Description Logics, vol. 33, pp. 1–19. CEUR

Alrabbaa, C., Baader, F., Borgwardt, S., Koopmann, P., & Kovtunova, A. (2021). *Finding good proofs for description logic entailments using recursive quality measures,12699*. https://doi.org/10.1007/978-3-030-79876-5_17. International Conference on Automated Deduction

Alrabbaa, C., Baader, F., Dachselt, S. B. R., & Méndez, P. K. J. (2022). Evonne: Interactive proof visualization for description logics (system description). In: Automated Reasoning. IJCAR 2022. Lecture Notes in Computer Science, vol. 13385. https://doi.org/10.1007/978-3-031-10769-6_16 . IJCAR

Alrabbaa, C., Baader, F., Dachselt, R., Flemisch, T., & Koopmann, P. (2020). Visualizing proofs and the modular structure of ontologies to support ontology repair. In: DL 2020: International Workshop on Description Logics. http://ceur-ws.org/Vol-2663/paper-2.pdf . CEUR Workshop Proceedings, vol. 2663. CEUR-WS.org

Anderson, J. R. (2005). Human symbol manipulation within an integrated cognitive architecture. *Cognitive Science, 29*(3), 313–341.

Anderson, J. R. (2007). *How Can the Human Mind Occur in the Physical Universe?* Oxford: Oxford University Press.

Anderson, J. R., & Byrne, M. D. (2004). An integrated theory of the mind. *Psychological Review, 111*(4), 1036–1060.

Anderson, J. R., & Lebiere, C. (1998). *The Atomic Components of Thought*. New York: Psychology Press.

Andrade, J. A. A. (2022). On the robustness to outliers of the student-t process. *Scandinavian Journal of Statistics*. https://doi.org/10.1111/sjos.12611

Baader, F., & Suntisrivaraporn, B. (2008). Debugging SNOMED CT using axiom pinpointing in the description logic $\mathcal{EL}^+$. In: Proceedings of the 3rd Knowledge Representation in Medicine (KR-MED'08): Representing and Sharing Knowledge Using SNOMED. CEUR-WS, vol. 410

Baader, F., Horrocks, I., Lutz, C., & Sattler, U. (2017). *An Introduction to Description Logic*. Cambridge: Cambridge University Press.

Borst, J. P., & Anderson, J. R. (2013). Using model-based functional MRI to locate working memory updates and declarative memory retrievals in the fronto-parietal network. *Proceedings of the National Academy of Sciences, 110*(5), 1628–1633. https://doi.org/10.1073/pnas.1221572110

Borst, J. P., & Anderson, J. R. (2015). Using the ACT-R Cognitive Architecture in combination with fMRI data. In: An introduction to model-based cognitive neuroscience (pp. 339-352). Springer, NewYork.

Brown, M. B., & Forsythe, A. B. (1974). Robust tests for the equality of variances. *Journal of the American Statistical Association,69*(346), 364–367. https://doi.org/10.1080/01621459.1974.10482955

Cameron, T. R., Charmot, S., & Pulaj, J. (2021). On the linear ordering problem and the rankability of data. *Foundations of Data Science, 3*(2), 133–149. https://doi.org/10.3934/fods.2021010

Clinical terminology SNOMED CT: experiences from the implementation of a terminology standard in the Swedish region of Västra Götaland. https://www.clearbyte.org/?p=7021&lang=en#more-7021. Accessed: 2024-02-22

Cohen, J. (1992). A power primer. *Psychological Bulletin, 112*(1), 1.

devs, A. (2024). Arviz. https://python.arviz.org/en/latest/index.html

Dotlacil, J. (2024). pyactr. https://github.com/jakdot/pyactr

Fokkens, J. T. (2023). Modelling the logical mind. Licentiate thesis, University of Gothenburg. Available at: https://gupea.ub.gu.se/handle/2077/74797

Fokkens, J. T., & Engström, F. (2024). SHARP-ABox-Consistency.https://doi.org/10.5281/zenodo.13944514. Source Code.
Fokkens, J. T., Engström, F. (2023). Human performance on a deductive reasoning task in the description logic ALE. https://doi.org/10.5878/5739-da47
Halford, G. S., Wilson, W. H., & Phillips, S. (2010). Relational knowledge: the foundation of higher cognition. *Trends in Cognitive Sciences, 14*(11), 497–505.
Hoffman, M. D., & Gelman, A. (2014). The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo. Journal of Machine Learning Research.
Introduction to ALE ABox Inconsistency. https://www.youtube.com/watch?v=4M65TjeHJSE&t=10s (2023)
Johnson-Laird, P. (2001). Mental models and deduction. *TRENDS in Cognitive Science, 5*(10), 434–442. https://doi.org/10.1016/S1364-6613(00)01751-4
Kruschke, J. K. (2014). *Doing Bayesian Data Analysis* (2nd ed.). Amsterdam: Elsevier.
Martin, O. (2018). *Bayesian Analysis with Python* (2nd ed.). Birmingham: Packt Publishing.
Neisser, U. (1979). The concept of intelligence. *Intelligence, 3*(3), 217–227. https://doi.org/10.1016/0160-2896(79)90018-7
Peñaloza, R. (2019). *Explaining axiom pinpointing*. Description Logic: Theory Combination, and All That.
PyMC. https://www.pymc.io/welcome.html (2022)
Qin, Y., Carter, C. S., Stenger, E. M. S. V. A., Fissell, K., Goode, A., & Anderson, J. R. (2004). The change of the brain activation patterns as children learn algebra equation solving. *Biological Sciences,101*(15), 5686–5691. https://doi.org/10.1073/pnas.0401227101
Rips, L. (1994). *Psychology of Proof: Deductive Reasoning in Human Thinking*. Cambridge, Massachusetts: MIT Press.
Sawilowsky, S. (2009). New effect size rules of thumb. *Journal of Modern Applied Statistical Methods, 8*(2), 26. https://doi.org/10.22237/jmasm/1257035100
Simonton, D. K. (2012). Quantifying creativity: can measures span the spectrum? *Dialogues in Clinical Neuroscience, 14*(1), 100–104. https://doi.org/10.31887/DCNS.2012.14.1/dsimonton
Stenning, K., & Lambalgen, M. (2012). *Human Reasoning and Cognitive Science*. Cambridge, Massachusetts: MIT Press.
Strannegård, C., Engström, F., Nizamani, A. R., & Rips, L. (2013). Reasoning about truth in first-order logic. *Journal of Logic, Language and Information, 22*, 115–137.
Sullivan, G. M., & Feinn, R. (2012). Using effect size-or why the p value is not enough. *Journal of Graduate Medical Education*. https://doi.org/10.4300/JGME-D-12-00156.1
Tubella, A. A., Strassburger, L. (2019) Introduction to Deep Inference. https://inria.hal.science/hal-02390267/document.
Warren, P., Mulholland, P., Collins, T., & Motta, E. (2015). Making sense of description logics. In: Semantics '15: Proceedings of the 11th International Conference on Semantic Systems. https://doi.org/10.1145/2814864.2814866 . International Conference on Semantic Systems
Whitehill, J. (2013). Understanding act-r - an outsider's perspective
Whittaker, M., Crawford, K., Dobbe, R., Fried, G., Kaziunas, E., Mathur, V., West, S.M., Richardson, R., Schultz, J., & Schwartz, O. (2018). AI Now 2018 Report. https://ec.europa.eu/futurium/en/system/files/ged/ai_now_2018_report.pdf