Accessible Reasoning with Diagrams: from Cognition to Automation

Zohreh Shams[1], Yuri Sato[2], Mateja Jamnik[1], Gem Stapleton[2]

[1] Department of Computer Science and Technology, University of Cambridge, UK
{zohreh.shams, mateja.jamnik}@cl.cam.ac.uk
[2] Centre for Secure, Intelligent and Usable Systems, University of Brighton, UK
{y.sato, g.e.stapleton}@brighton.ac.uk

Abstract. High-tech systems are ubiquitous and often safety and security critical: reasoning about their correctness is paramount. Thus, precise modelling and formal reasoning are necessary in order to convey knowledge unambiguously and accurately. Whilst mathematical modelling adds great rigour, it is opaque to many stakeholders which leads to errors in data handling, delays in product release, for example. This is a major motivation for the development of diagrammatic approaches to formalisation and reasoning about models of knowledge. In this paper, we present an interactive theorem prover, called iCon, for a highly expressive diagrammatic logic that is capable of modelling OWL 2 ontologies and, thus, has practical relevance. Significantly, this work is the first to *design* diagrammatic inference rules using insights into what humans find accessible. Specifically, we conducted an experiment about relative cognitive benefits of *primitive* (small step) and *derived* (big step) inferences, and use the results to guide the implementation of inference rules in iCon.

1 Introduction

The long-held assumption that using diagrams makes modelling and reasoning accessible, goes back to ancient times (e.g., Euclid's Elements). Despite this, the development of automated diagrammatic reasoning tools (e.g., Hyperproof [2], Diamond [8] and Speedith [22]) has been rare in comparison to sentential theorem provers. One of the main areas yet to be explored in diagrammatic reasoning is the level of abstraction employed when constructing proofs, which relies on inference rule style. The 'right' level of abstraction can facilitate interaction between users and the system as well as increase the readability of the generated proofs [11, 3]. But what is the 'right' level of abstraction for a human user? In this paper we introduce a diagrammatic reasoning system, iCon, for concept diagrams [19], which is designed so that the level of abstraction for diagrammatic inference rules is based on empirical results of what humans find accessible.

In sentential theorem proving, *tactics*, tacticals, proof strategies, proof methods and 'derived rules' are all attempts to achieve higher level of abstraction in logical reasoning [5]: they enable applying a sequence of inference rules all in one go. Tools such as Isabelle [13] exploit tactical reasoning to provide a high level of abstraction and some level of automation. But the use of tactics in diagrammatic logics is largely unexplored. One attempt at controlling the level of abstraction in diagrammatic theorem proving is seen in Speedith [22] (a theorem prover for spider diagrams [4]) which uses tactics [11]. The choice of tactics in [11] is guided by metrics, which are informed by empirical studies [10] related to readability such as proof length and diagram clutter.

We focus on concept diagrams, which are based on spider diagrams but are more expressive. They were developed as a formal visualisation method for defining and reasoning about ontologies. Empirical evidence suggests [6] that concept diagrams are more accessible than the standard ontology language[1] OWL 2 and description logic [1]. In addition, cognitive theories support their effectiveness over common node-link ontology representation approaches [18].

Our goal is to develop a diagrammatic theorem prover for concept diagrams whose inference rules are designed to be accessible to users. We conducted an experiment to assess what level of abstraction users find accessible. We studied 10 inference tasks, based on practically relevant ontology inference problems formulated in [12]. The tasks were presented in two variations: using *primitive* inference steps and using *derived* inference steps, where the latter are coarser and more abstract than the former. User performance was measured in terms of their accuracy in identifying the validity of inference tasks. The level of abstraction (i.e., primitive vs. derived) that resulted in significantly higher accuracy rate was interpreted as the 'right' level of abstraction. We found that an appropriate level of abstraction was rule dependent. These results serve as the basis of design and implementation of inference rules in our interactive theorem prover iCon.

One of the unique selling points of iCon, and the main contribution of this paper, is the fact that the design of inference rules is guided by empirical studies of what inference rules people performed more accurately with, rather than being motivated by meta-level considerations such as completeness, which is often the case in formal systems. Although tactics incorporated in Speedith [22] are based on metrics that are found empirically, in this paper we have taken a step further by empirically testing the inferences themselves.

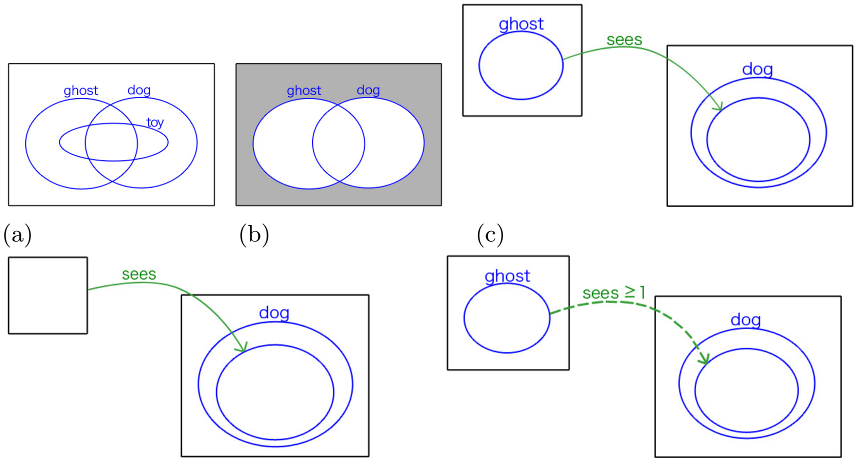
We give an overview of concept diagrams in Section 2, and introduce iCon in Section 3. In Section 4 we report on the empirical study that compares primitive vs. derived inference tasks. How the results of the empirical study inform the design of diagrammatic inference rules in iCon is discussed in Section 5. We compare our contributions to related work in Section 6 and finally, conclude in Section 7.

2 Concept diagrams: background and overview

Concept diagrams were introduced for the purpose of visualising and specifying ontologies, and they are expressive enough to handle binary predicates [19]. They consist of syntactic objects such as rectangles, closed curves, and shading (as seen in Euler and Venn diagrams) as well as other additional objects such as dots, solid arrows and dashed arrows.

Rectangles are used to represent all individuals in the world. By combining curves inside a rectangle, we can represent several cases. For example, in Fig-

[1] https://www.w3.org/TR/owl2-direct-semantics/



(a) (b) (c)

(d) (e)

Fig. 1. Concept diagrams: (a) toys are either ghosts, dogs, or both; (b) there is nothing which is not a ghost or a dog; (c) all ghosts see *only* dogs; (d) all individuals in the world see *only* dogs; (e) all ghosts see *at least* one dog.

ure 1(a), the circle toy is inside two circles, ghost and dog. Thus, toys are either ghosts, dogs, or both. Note that concept diagrams do not adopt the existential import assumption: the presence of a minimal region says nothing about whether there are some individuals in it (for details, see [15]). Shading is used to represent that there is nothing (i.e., to assert set emptiness). In Figure 1(b), the region outside of ghost and dog is shaded. This means that there is nothing which is not a ghost or a dog. That is, everything is a ghost or a dog. The syntax described so far should be familiar in that concept diagrams with only rectangles, closed curves and shading are Euler diagrams.

Concept diagrams add syntactic objects to Euler diagrams, including arrows which are used to express verb relations. There are two kinds of arrows: solid and dashed ones. Solid arrows mean that the source is related to *only* the target. For example, in Figure 1(c), the solid arrow labelled sees connects from the circle ghost to the unlabelled circle inside dog. This means that all ghosts see *only* dogs. Figure 1(d) is another example where the solid arrow sees connects from the rectangle to the unlabelled circle inside the circle dog. This means that all individuals in the world see *only* dogs. On the other hand, dashed arrows mean that the source is related to *at least* the target. That is, the source may be also related to other targets. For example, in Figure 1(e), the dashed arrow sees connects the circle ghost to the unlabelled circle inside the circle dog. Together with the arrow annotation ≥ 1, this means that all ghosts see *at least* one dog.

Here is an informal overview of the syntax and semantics of concept diagrams; for formalisation, see [20]. A *concept diagram* is a collection of *boundary rectangles* including the syntax contained by them, and all arrows *connecting* them. Each boundary rectangle properly contains some (possibly empty) set of closed curves, some of which (possibly none or all) are labelled. Labelled closed curves represent specific sets (e.g., the set of dogs in Figure 1(d)) or an anonymous set (e.g., some unnamed subset of dogs in Figure 1(d)). The closed curves within a rectangle partition the plane into *zones*: a zone is a region inside some (possibly no) curves and outside the rest of the curves. For example, in Figure 1(d), the concept diagram comprises two boundary rectangles, one of which contains no curves, the other contains two curves; these two curves give rise to three zones. In general, a set of zones inside a boundary rectangle is called a *region*.

Zones can be shaded and they may also contain dots. In addition, dots can be joined together by straight lines to form *spiders*. Each spider represents an individual. If the spider is labelled, it represents a specific individual. An unlabelled spider, just like an unlabelled curve, represents an anonymous individual. Distinct spiders represent distinct individuals, unless joined by $=$ to assert their equality, or by $\overset{?}{=}$ to indicate that it is unknown whether they represent the same individual. Also, the individual represented by a spider is an element of the sets represented by the region in which the spider is placed.

The last major component of concept diagrams is arrows, which are of two types, dashed and solid. Arrows are sourced and targeted on boundary rectangles, closed curves, or spiders. Each arrow has a label, p, which represents a binary relation. The source and target of any given arrow need not be inside the same boundary rectangle. In addition, the label can be annotated with $^-$ to indicate the inverse of the relation, or with cardinality constraints: $\leq n$, $\geq n$ or $= n$, where n is a natural number. Semantically, a solid arrow with source s, label p (resp. p^-) and target t expresses (blurring the distinction between syntax and semantics) that if the domain of p (resp. p^-) is restricted to the source s then the image is t. If, however, the arrow is instead dashed, then it expresses that if the domain of p (resp. p^-) is restricted to the source s then the image is a *superset* of t. In Figure 1(c), the solid arrow expresses that, under the relation sees with domain restricted to ghost, the image is an anonymous subset of dog. Intuitively, this means that ghosts see only dogs. In Figure 1(e), the dashed arrow expresses that, under the relation sees with domain restricted to ghosts, the image includes some anonymous subset of dog. Intuitively, this does not tell us anything. It is only through the use of the additional annotation, ≥ 1, that this arrow provides information: each ghost sees at least one dog.

Each rectangle, and its contents, in a concept diagram is called a *class and object property diagram*. This is because its curves represent classes (which are sets of individuals) and its arrows give information about properties (which are binary relations). Thus, a concept diagram is a set of class and object property diagrams, along with any connecting arrows.

3 iCon: A concept diagrams interactive theorem prover

We built an interactive theorem prover[2] for concept diagrams, iCon, that can be used to reason, for example, about ontologies. iCon consists of the *reason-*

[2] Available at: https://github.com/ZohrehShams/iCon.

ing engine and the *graphical user interface (GUI).* We based iCon's design on Speedith [22], a theorem prover for spider diagrams [4], since concept diagrams are based on spider diagrams. In Speedith, the design of inference rules was based on obtaining soundness and completeness, whereas the design of inference rules in iCon is guided by an experiment (Section 4) into what abstraction level of deduction steps do people perform more accurately with. This approach to designing iCon improves the readability of the resulting proofs.

3.1 Reasoning engine

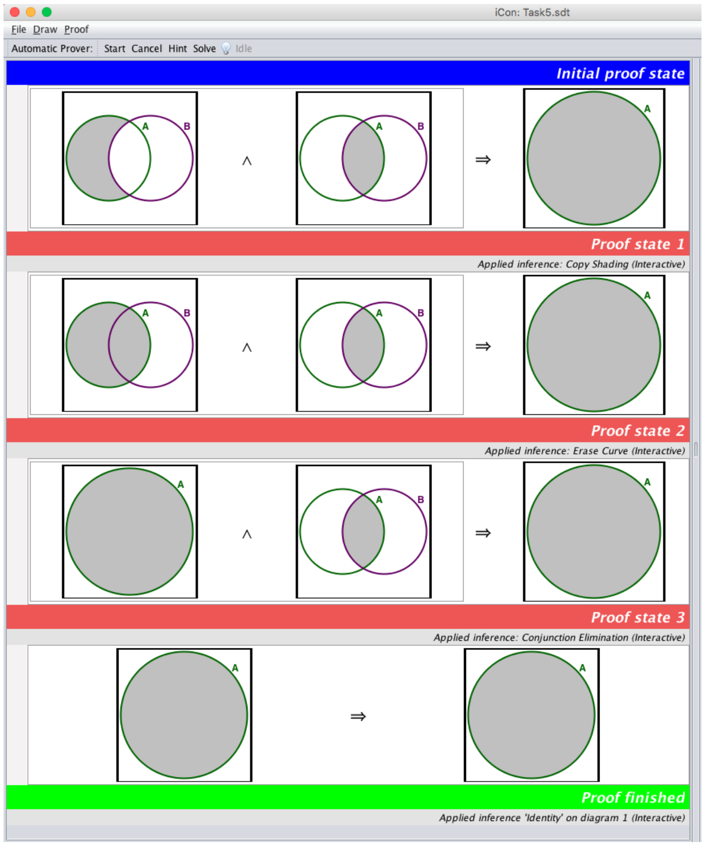
The iCon reasoning engine (i) contains a collection of *inference rules*; (ii) handles the application of inference rules to diagrams expressed in an abstract syntax; and (iii) manages *proofs.* iCon only applies valid inference rules, and, since these are sound, proofs generated in iCon are guaranteed to be correct.

Proofs A proof in iCon starts with the *initial proof state*, denoted as $\mathit{\Delta}_0$ which is of form $(d_1 \wedge \cdots \wedge d_m) \Rightarrow d_n$, where d_i are concept diagrams. This means that if $d_1, \cdots, d_m$ (referred to as *goals*, and denoted as set G) hold, then d_n holds. Proofs are linear and constructed by applying sequences of *inference rules* on goals. Starting from $\mathit{\Delta}_0$, the proof continues by applying inference rules to a goal $d \in G$. The result of applying an inference rule (with the exception of the inference rule *Identity* that will be explained in the next section) is a diagram d', such that d semantically entails d' ($d \vDash d'$). In a proof, $P = \mathit{\Delta}_0, \cdots, \mathit{\Delta}_k$ s.t. $0 \leq i < k$, there has to be a goal d in the set of goals in proof state $\mathit{\Delta}_i$ and d' in the set of goals in proof state $\mathit{\Delta}_{i+1}$ such that d' is the result of applying one of the inference rules to d. Proof P is finished if the final proof state $\mathit{\Delta}_k$ is of the form $d_n \Rightarrow d_n$, which means d_n implies d_n, and is trivially true.

Figure 2 shows a proof where proof states are separated by red bars. The inference rules are applied on the proof states above the lines and result in the proof states below the lines (stated as "Applied inference"). In the final proof state, as expected, the diagram on the left hand side is identical to the one on the right hand side. Finally, the proof is finished by applying *Identity.* The rest of inference rules used in this figure will be formally defined in Section 5 (page 11).

Inference rule design Inference rules are divided into logical and diagrammatic rules, where the former correspond to entailments and equivalences of propositional logic, while the latter rewrite the diagrams representing the goals. The fragment of concept diagrams used in this paper only uses $\wedge$ operator; therefore, the logical rules applicable are: *Conjunction Elimination* ($(d_1 \wedge d_2) \Rightarrow d_1$ and $(d_1 \wedge d_2) \Rightarrow d_2$), *Conjunction Idempotency* ($(d \wedge d) \Leftrightarrow d$), and *Identity* (applied to the final proof state to express that $d \Rightarrow d$ is trivially true, and thus concludes the proof). The diagrammatic inference rules are motivated by the domain in which the reasoner is intended to be used: we chose ontology reasoning and debugging. We focus on a study [12] that provides statistical evidence for the practical significance, commonality and coverage of inference rules that it introduces for ontology entailment reasoning.[3] There are 51 inferences (referred

[3] Any ontology reasoning task can be reduced to entailment reasoning.



Initial proof state

Proof state 1

Applied inference: Copy Shading (Interactive)

Proof state 2

Applied inference: Erase Curve (Interactive)

Proof state 3

Applied inference: Conjunction Elimination (Interactive)

Proof finished

Applied inference 'Identity' on diagram 1 (Interactive)

Fig. 2. An example of a proof.

to as ontology deduction patterns) identified in [12] and are ranked based on their accessibility measured through user studies. Each inference rule consists of up to four premises and a single conclusion and can often be broken down to more fine grained inference rules by introducing intermediate steps.

When designing the chosen diagrammatic inference rules, there are important choices with regards to the level of abstraction for these rules. To inform these choices, we conducted an empirical study (see Section 4) that compares two variations of diagrammatic inference rules, with different levels of abstraction and granularity in a number of deduction patterns. One variation takes the premises and conclusion of the deduction patterns in [12], and introduces an intermediate step; we call this variation *primitive*. The other variation is *identical* to the inference rules in [12], consisting of premises and conclusion only without an intermediate step; we call this variation *derived* – it is more abstract than the primitive version. The findings of the experiment comparing primitive and derived diagrammatic inference steps guides the process of implementing them, and it will be further discussed in Section 5.

3.2 Graphical user interface

iCon provides a graphical user interface that allows visualising diagrams and applying inference rules on them interactively. Visualisation of diagrams is based on iCircles [21] – a Java library for drawing Euler diagrams using circles. iCircles was extended in Speedith [22] to represent spiders (existential elements). Here we extended it further to visualise unlabelled and labelled spiders and curves, and also to visualise solid and dashed labelled arrows with possible cardinalities. Users can select via a graphical point-and-click mechanism any part of the diagram to apply a diagrammatic inference rule on. This changes the diagram's abstract representation, and the new abstract representation is visualised using the visualiser. For screenshot of iCon, see Figure 2.

4 Empirical experiment for the design of inference rules

To determine the right level of abstraction for diagrammatic rules, we compared participants' performance in inference tasks with concept diagrams in examples proved using primitive rules with those proved by derived inference rules.

4.1 Method

Fifty-one undergraduate students from seven classes on elementary computer science at the University of Brighton were recruited. The mean age was 24.12 ($SD = 5.89$) with a range of 19–48 years. All participants gave informed consent and were paid for their participation. The experiment method was approved by the CEM School Research Ethics Panel of University of Brighton. In order to provide an inference system that is accessible to a broad range of people, not just ontology experts, none of the participants had any prior knowledge of ontology engineering. One participant withdrew, so their data was excluded. Participants were randomly divided into two groups: the primitive group (N = 25) and the derived group (N = 25).

The participants in the primitive group were given tasks with intermediate step diagrams as well as premise and conclusion diagrams (i.e., primitive rules are applied in the proof; see Figure 3 on page 9). The participants in the derived group were given tasks with only premise and conclusion diagrams without intermediate steps (i.e., derived rules are applied). Participants were asked to determine if the diagram transformations were valid or not. We presented 20 items in total: 10 consisted only of valid transformations of diagrams (#01–10) and 10 items included invalid transformations of diagrams (#11–20). The valid 10 items were selected from the *medium* difficulty amongst the 51 patterns given in [12]. This is because in any study, tasks that are too easy (leading to ceiling effect) or too hard (floor effect) reveal no insights. Minimal changes were made to the valid items (e.g., a set name, relation, cardinality) to create invalid ones. The tasks were further divided into the so-called Euler-Venn diagram level (#01–05; #11–15; they include only labelled curves and shading), and the concept diagram level (#06–10; #16–20; they include arrows and other syntax); the tasks

Table 1. List of inference tasks (#01–10 are valid; #11–20 are invalid. #01–05/#11–15 are Euler-Venn diagram level; #06–10/#16–20 are concept diagram level) and their accuracy rates ($*$ refers to a significant difference between the two groups at $p < 0.05$; $+$ refers to $p < 0.10$).

Number	Premises $\Rightarrow$ Conclusion	Primitive%		Derived%
01.	$(A \sqsubseteq B) \wedge (Dis(A,B)) \Rightarrow (A \sqsubseteq \bot)$	95.0		80.1
02.	$(Dis(A,B)) \wedge (C \sqsubseteq A) \wedge (D \sqsubseteq B) \Rightarrow (Dis(C,D))$	80.0		85.7
03.	$(A \sqsubseteq (B \sqsubseteq C)) \wedge (B \sqsubseteq C) \Rightarrow (A \sqsubseteq C)$	60.0	$+$	85.7
04.	$(\top \sqsubseteq B) \wedge (Dis(A,B)) \Rightarrow (A \sqsubseteq \bot)$	60.0		66.7
05.	$(A \sqsubseteq B) \wedge (A \sqsubseteq \neg\, B) \Rightarrow (A \sqsubseteq \bot)$	70.0		76.2
06.	$(A \sqsubseteq \exists R.B) \wedge (Rang(R,C)) \Rightarrow (C \sqsubseteq \exists R.(B \sqcap C))$	75.0		80.1
07.	$(A \sqsubseteq \exists R.(B \sqcap C)) \wedge (Dis(B,C)) \Rightarrow (A \sqsubseteq \bot)$	70.0		85.7
08.	$(A \sqsubseteq \geq 3R.B) \wedge (A \sqsubseteq \leq 1R.B) \Rightarrow (A \sqsubseteq \bot)$	50.0		52.4
09.	$(A \sqsubseteq \exists R.B) \wedge (B \sqsubseteq \bot) \Rightarrow (A \sqsubseteq \bot)$	65.0		57.1
10.	$(A \sqsubseteq \geq 4R.B) \wedge (Fun(R)) \Rightarrow (A \sqsubseteq \bot)$	20.0	$+$	47.6
11.	$(B \sqsubseteq A) \wedge (Dis(A,B)) \Rightarrow (A \sqsubseteq \bot)$	75.0		71.4
12.	$(Dis(A,B)) \wedge (C \sqsubseteq A) \wedge (B \sqsubseteq D) \Rightarrow (Dis(C,D))$	70.0		52.4
13.	$(A \sqsubseteq (B \sqsubseteq C)) \wedge (B \sqsubseteq C) \Rightarrow (A \sqsubseteq B)$	55.0		47.6
14.	$(\top \sqsubseteq B) \wedge (A \sqsubseteq B) \Rightarrow (A \sqsubseteq \bot)$	60.0		66.7
15.	$(A \sqsubseteq B) \wedge (\neg\, B \sqsubseteq A) \Rightarrow (A \sqsubseteq \bot)$	50.0	$*$	85.7
16.	$(A \sqsubseteq \exists R.B) \wedge (Rang(R,C)) \Rightarrow (Rang(R, C \sqcap \neg B))$	40.0		38.1
17.	$(A \sqsubseteq \exists R.(B \sqcup C)) \wedge (Dis(B,C)) \Rightarrow (A \sqsubseteq \bot)$	85.0		85.7
18.	$(A \sqsubseteq \geq 1R.B) \wedge (A \sqsubseteq \leq 3R.B) \Rightarrow (A \sqsubseteq \bot)$	70.0		66.7
19.	$(A \sqsubseteq \exists R.B) \wedge (A \sqsubseteq \bot) \Rightarrow (B \sqsubseteq \bot)$	60.0		66.7
20.	$(A \sqsubseteq \geq 1R.B) \wedge (Fun(R)) \Rightarrow (A \sqsubseteq \bot)$	90.0		81.0

are summarised using stylised description logic syntax in Table 1.[4] Tasks #01 and #06 are semantically equivalent and can be expressed by the same diagrammatic representation. Thus, #01 was expressed by Venn diagrams and #06 was expressed by Euler diagrams. The tasks were presented in one of three random orders as a paper-and-pencil test. There was no time limit for completing the tasks, although the approximate time (30 minutes) for taking the experiment was instructed.

All participants were gathered in a room. First, the participants were provided with three pages of instructions on the basic meaning of concept diagrams, but not on particular rules to solve inference tasks. Second, a pretest was conducted to check whether they understood the instructions correctly; they were asked to choose, from a list of three possibilities, the sentence corresponding to the meaning of a given diagram (for the importance of pretest settings, see [15]). Third, the participants were provided with one page of instruction on the meaning of a valid transformation (entailment), with two examples of diagrams: one

[4] For unfamiliar readers, informally the DL syntax has the following interpretation: $A \sqsubseteq B$: A is a subset of B; $\bot$: the empty set; $\exists R.A$: the set of things related to something in set A under binary relation R; $Rang(R.A)$: the range of R is A; $Fun(R)$: R is functional; $\geq nR.A$: the set of things related to at least n things in A under R; $\leq nR.A$: the set of things related to at most n things in A under R.

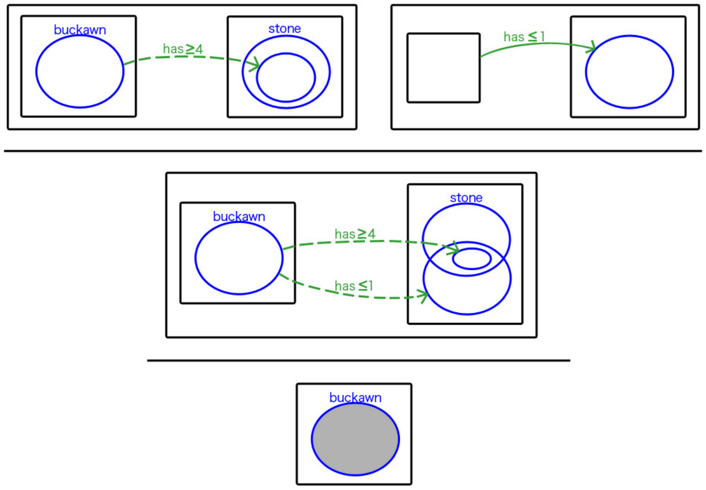


Fig. 3. A task (#10) in the primitive group. In the derived group, the intermediate diagram was removed.

was valid and one was not valid.[5] After the instruction phase, the participants were asked to solve the main reasoning tasks of the experiment.

4.2 Results

The data for participants who made mistakes in more than two items (out of five) in the pretest was removed. In our analysis, 5 out of 25 in the primitive group, and 4 out of 25 participants in the derived group were removed.

The accuracy data for each task was analysed using a χ^2 test. In task #15, accuracy rates in the derived group were significantly higher than those in the primitive group (47.1% vs. 88.2%, $p = 0.014$). In task #03, accuracy rates in the derived group were significantly higher than those in the primitive group, at a reduced threshold of $p < 0.10$ (64.7% vs. 88.2%, $p = 0.063$). In task #10, accuracy rates in the derived group were significantly higher than those in the primitive group, at a reduced threshold of $p < 0.10$ (23.5% vs. 47.1%, $p = 0.062$). In other tasks, there were no significant differences between both groups.

In the comparison between #01 (expressed with Venn diagrams) and #05 (expressed with Euler diagrams), a significant difference was found in the primitive group (95.0% vs. 70.0%, $p = 0.037$), but not in the derived group (80.1% vs. 76.2%, $p = 0.432$). Overall, the comparison of accuracy data between the primitive group and the derived group revealed no significant difference.[6]

[5] See `https://sites.google.com/site/myardproject/exp/MateInst2.zip?aredirects=0&d=1` for full instructions.

[6] 65.0% vs. 69.1% (Mann-Whitney $U = 179$, $p = 0.416$) for overall (#01--20), 64.5% vs. 71.9% ($U = 179$, $p = 0.159$) for valid transformations (#01--10), 65.5% vs. 66.2% ($U = 208$, $p = 0.958$) for invalid ones (#11--20), 73.0% vs. 79.1% ($U = 166$, $p = 0.232$)

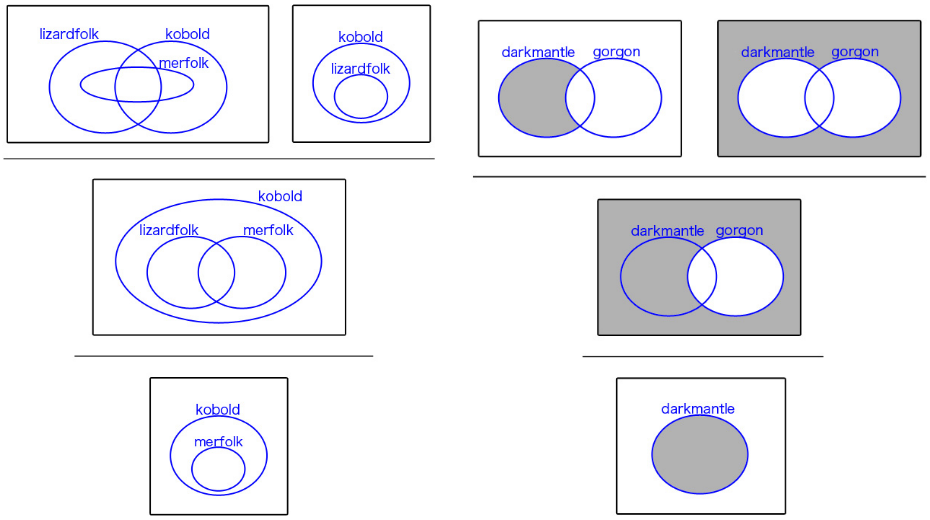


Fig. 4. Task #03 (left) and task #15 (right) in the primitive group. In the derived group, the intermediate diagrams were removed.

4.3 Discussion

In task #10, shown in Figure 3 (described as $(A \sqsubseteq \geq 4R.B) \land (Fun(R)) \Rightarrow (A \sqsubseteq \bot)$), 80% of participants *incorrectly* judged the validity of diagram transformation using primitive rules. In comparison to the (random) chance level of 50%, there was a significant difference in the primitive group ($p = 0.047$), but not in the derived group ($p = 0.877$). In the diagram transformation using primitive rules, the solid arrow from the rectangle is explicitly rewritten into the dashed arrow from the curve inside the rectangle. On the other hand, there is no explicit rewriting of solid and dashed arrows in the diagram transformation using derived rules. Since this difference is found in task #10, it is a candidate to explain the result of task #10 where better accuracy was achieved with the derived rule. Therefore, a resulting heuristic suggests that we should not design diagram transformations where solid arrows are replaced by dashed arrows.

On the other hand, what causes the significant differences in tasks #03 and #15 between the primitive and the derived groups? In the diagram transformation in task #03, as shown in Figure 4 (left), it is important that the curves crossing between lizardfolk and kobold in the first premise diagram mean that the semantic relationship between them is indeterminate (see the existence-free assumption for minimal regions, mentioned in Section 2). Therefore, the unification between the premise diagrams results in the diagram where (i) lizardfolk is inside kobold, (ii) merfolk is inside kobold, (iii) the relation between lizardfolk

for valid Euler ones (#01--05), 56.0% vs. 64.8% ($U = 166.5$, $p = 0.242$) for valid concept ones (#06--10), 52.0% vs. 64.8% ($U = 199$, $p = 0.769$) for invalid Euler ones (#11--15), and 69.0% vs. 67.6% ($U = 194.5$, $p = 0.673$) for invalid concept ones (#15--20).

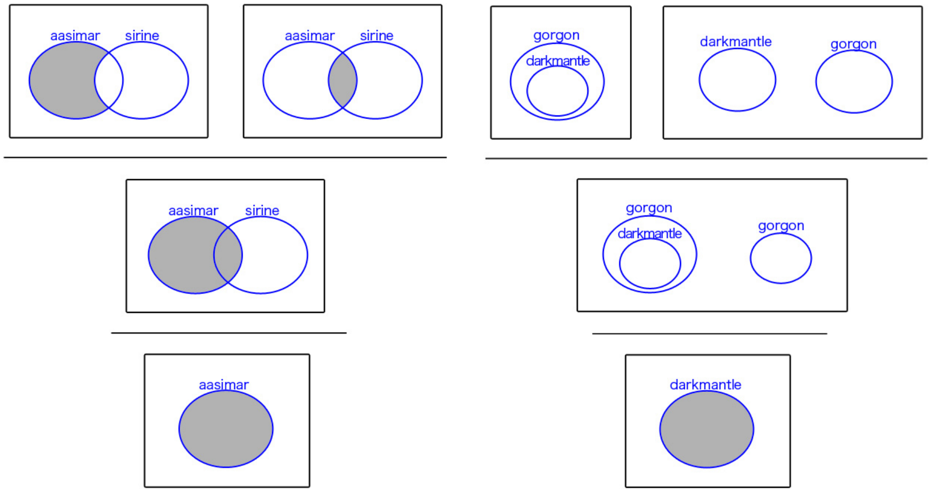


Fig. 5. Task #01 (left) and task #05 (right) in the primitive group. In the derived group, the intermediate diagrams were removed.

and merfolk is unknown. Here, understanding crossing curves plays an essential role in both deducing (i–iii) using a primitive rule *and* deducing (ii) using a derived rule. Thus, the meaning of crossing curves cannot explain the performance difference between both groups. In the case of task #15, as shown in Figure 4 (right), shading plays an important role in solving the task. However, the same can be said for tasks #01, 04, 11, and 14, where significant differences in accuracy performances were not found. Thus, it is not clear why the difference between both groups was found only in task #15.

As stated before, tasks #01 and #05, which are semantically equivalent, were expressed by Venn and Euler diagrams, respectively, as shown in Figure 5. The result that the accuracy rate for #01 was higher than for #05 in the primitive group suggests that Venn diagrams are more suitable than Euler for reasoning about the emptiness of a set.[7] In #05 (using Euler diagrams), the derivation of the shaded curve labelled darkmantle, meaning darkmantle $\sqsubseteq \bot$, requires not only spatial operation on diagrammatic objects, but also meta-level information concerning semantic values (cf. the discussion in [16]). Whether for primitive rules or derived rules, reasoning with Euler diagrams in this case can require more cognitive effort than with Venn diagrams. Note that the effectiveness of Venn over Euler diagrams is distinct from previous empirical findings [15].

5 Inference rules: design and implementation guidelines

We chose ontology reasoning and debugging for the first application domain of iCon. In order to develop a theorem prover with practical relevance in this domain, we focused on the 51 inferences found in [12]. However, there are a lot

[7] Note that in ontology engineering, sets that are necessarily empty are called *incoherent*.

of choices when designing the diagrammatic version of these inferences in iCon. For example, the inference patterns are large and can often be broken down into smaller steps, but what is the right level of granularity for the diagrammatic inference rules? To ensure accessibility, this granularity level was informed by our experiment where we translate the results into design and implementation guidelines. Note that in the user study we tested the accessibility of 10 inferences out of 51 from [12], but the design guidelines are general and can be used in the implementation of any concept diagram inference rules.

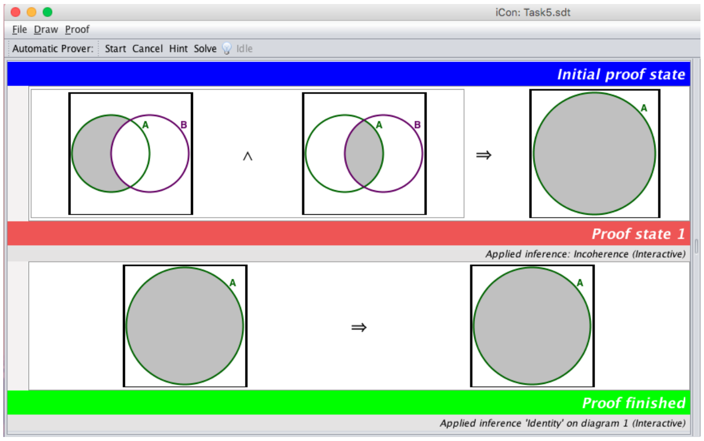
Overall, no significant difference was observed between primitive and derived rules, which suggests that the level of abstraction in the implementation is a choice that the users ought to have and use as they see appropriate. Thus, the first guideline we extract from the experiment is to implement both primitive and derived versions of the inference rules.

The second guideline is related to the heuristic (Section 4.3, first paragraph) that suggests that inference rules should not transform solid arrows into dashed arrows. In the user study, tasks #6-10 involve arrows, with only task #10 transforming arrows in its primitive version (Figure 3). Thus, we adopted the primitive version of task #10, such that in the intermediate step diagram, the arrow with cardinality ≤ 1 stays solid (right hand premise in Figure 3). For the remaining 41 inferences, we employ this heuristic and ensure that their diagrammatic versions retain the arrow type.

The third guideline is based on the heuristic (Section 4.3, last paragraph) that Venn seems to be a more effective representation than Euler when proving incoherence of a set. More than 20% of 51 inferences from [12] prove incoherence, and thus should employ this heuristic. Concretely, in our study, tasks #1,#4, #5, and #7-10 deduce that a set is incoherent. Out of these, tasks #1,#4, #5, and #7 can be alternatively represented in Euler or Venn form. In the experiment, apart from task #5, which used Euler form (see Figure 5 (right)), the rest of them were presented in Venn form. We revised task #5 accordingly, such that it is presented in Venn form.

We now exemplify how these guidelines are put into practice in iCon. We focus on the design and implementation of inference rules for task #5 in Figure 5 (right). Following the first design guideline, inference rules for both, primitive and derived version of this task are implemented. In fact, the proof in Figure 2 (page 6) shows the implementation of the primitive version, while Figure 6 shows the implemented derived version, where A is darkmantle and B is gorgon. The second design guideline does not apply here, as there are no arrows. As seen in Figures 2 and 6, following the third guideline presents the premises in Venn instead of the Euler form that was originally used in our study. We now formally define of the diagrammatic inference rules used in Figures 2 and 6. These rules can be proved sound as in [17]. In addition to diagrammatic rules, the two logical rules used are *Conjunction Elimination* and *Identity* (see Section 3.1).

The first diagrammatic rule we define copies shading from one region to another. It relies on syntactically identifying when two regions represent the same set. This has been extensively studied for Euler diagrams [7] and spider dia-



Applied inference: Incoherence (Interactive)

Applied inference 'Identity' on diagram 1 (Interactive)

Fig. 6. Proof of task # 5 using derived rule.

grams [22]. Thus, the syntactic identification of regions that represent the same sets can be identified using the underlying Euler diagram (i.e., the boundary rectangle containing only the labelled curves). Given a set of labelled curves, C, a *fixed zonal region* is the set of zones that are inside all of the curves in C and possibly other curves; note the name 'fixed' since such regions represent particular, that is, 'fixed' sets and are not anonymous. Given a region, r, in a concept diagram, d, we say that r is *fixed* if it is formed from a union of fixed zonal regions. Fixed regions are said to be *corresponding* if, informally, they represent the same set (the details of how this can be identified syntactically can be found in [22]). We can now define the *Copy Shading* inference rule.

Definition 1 (Copy Shading). *Let d_1 and d_2 be two concept diagrams containing corresponding fixed regions r_1 and r_2, respectively, where:*

1. *r_1 comprises only shaded zones and r_2 has at least one non-shaded zone,*
2. *any spider with a dot in r_1 (resp. r_2) is completely contained by zones in r_1 (resp. r_2), and*
3. *the spiders in r_1 match the spiders in r_2.*

Let d_2' be a copy of d_2 except that r_2 is entirely shaded. From $d_1 \wedge d_2$ we can infer $d_1 \wedge d_2'$ and vice versa.

The Next rule needed for task #5 deletes a curve from a concept diagram.

Definition 2 (Erase Curve). *Let c be a curve in a concept diagram d. Then c can be removed from d, resulting in a new diagram, d', with modified shaded zones, spider habitats and arrows. In particular, if upon erasure of c, a shaded zone merges with a non-shaded zone then the shading is removed, otherwise the shading is preserved. Also, if a spider s has a foot in two zones that collapse into one, then the spider will have a foot in the collapsed zone in d'. In addition, arrows that have c as target or source are deleted when forming d'. From d we can infer d'.*

Unlike the *Copy Shading* rule, which preserves semantics and is an equivalence, *Erase Curve* weakens information and can be applied only in one direction.

Our next rule, *Incoherence* is used in the derived version of task #5. It allows deducing that a curve, say c, is entirely shaded, by copying shading from a conjunct diagram in which the corresponding non-shaded region of c are shaded.

Definition 3 (Incoherence). *Let d_1 and d_2 be two concept diagrams containing curves c_1 and c_2, respectively, such that: c_1 and c_2 have the same label as each other; c_1 and c_2 do not contain any spiders; and the non-shaded zones inside c_1 in d_1 form a fixed region that corresponds to some entirely shaded, fixed region contained by c_2 in d_2. Let d_3 be a concept diagram comprising a single boundary rectangle containing a curve, c_3, with the same label as c_1 and c_2 whose interior is entirely shaded. From $d_1 \wedge d_2$ we can infer d_3.*

Currently, iCon implements 7 out of 10 inferences from [12] that were used in the user study, plus additional rules that enable the user to vary the granularity of the proof. To provide the same coverage as in [12], we are currently building both primitive and derived variations for the remaining inferences in [12].

6 Related work and evaluation

Like iCon, DIAMOND [8] and Cinderella [9] are diagrammatic theorem provers, however they operate in different domains of inductive theorems of natural numbers and geometry, respectively. In contrast, iCon deals with theorems about sets. Unlike DIAMOND and iCon, proof steps in Cinderella may not be sound and have to be verified externally by an automatic symbolic theorem prover.

iCon's concept diagrams are a more expressive extension of Speedith's spider diagrams. Similarly to Speedith, the inference rules in iCon are purely logical or diagrammatic. However, in Speedith the choice of inference rules is motivated by the completeness property. In contrast, in iCon the focus is on the commonality of the inference rules in the ontology domain. In addition, the design and implementation of inference rules in iCon is informed by empirical studies of what people find intuitive, and in particular, what level of granularity of rules enables human users to reason most accurately. This is in line with one of the most challenging areas of theorem proving, which is reducing the gap between user's model of the proof and the actual proof constructed by mechanised theorem provers [3]. The proof steps in the user's model are often coarser and have intuitive semantics, whereas the prover's steps tend to be much more fine grained. Our user study presented inference rules at different levels of granularity. The derived ones, which are coarser than the primitive ones, can be seen as tactics, as is common in sentential theorem proving. However, the role of tactics in sentential theorem proving is typically to provide some level of automation, and rarely to reduce the gap between user's reasoning and that of a prover. Our work addresses both, automation as well as human approach to constructing proofs. For instance, in Figure 6 the tactic *Incoherence* reduces the length of proof in comparison with Figure 2, while it still remains accessible by allowing the user to choose a curve and establish its incoherence.

Speedith deploys diagrammatic tactics to facilitate user interactions and devise a higher abstraction level in proofs which renders them more self-explanatory [11]. The choice of tactics is guided by metrics related to the readability of proofs (e.g. length of proof, the amount of clutter) and are informed by empirical studies. Our work presents a step change in that we directly test the accessibility of inferences themselves.

7 Conclusion and future work

Usability and accessibility of reasoning systems is paramount to harness their utility in diverse domains. By developing an interactive diagrammatic theorem prover iCon, we demonstrated that it is possible to build a formal reasoning system that is based on empirical studies of what humans find accessible.

iCon implements the logic of concept diagrams and can be applied in various domains (e.g., for reasoning in ontology engineering as presented here). We focused on deduction patterns found in [12], which also discusses their significance in terms of commonality and coverage. In order to gain an insight into how to implement the concept diagrams version of these patterns, we conducted an empirical study that identified that the level of abstraction and granularity of inference rules did not affect what human reasoners find accessible in general, but was rule specific. We used this result and others explained in Section 4.3 to guide the design and implementation of iCon's inference rules.

Displaying the application of inferences via the GUI presents many challenges and avenues for future work. Laying out the drawn diagrams after each inference requires analysing the invariant parts of the diagrammatic statement, because these are the syntactic elements that must remain unchanged before and after the application of the inference rule. But there are choices and trade-offs between what elements could or should be preserved. We are planning to conduct a user study that will investigate where this trade-off lies with the human users. Furthermore, the layout algorithms of iCon should preserve certain wellformedness properties of the diagrams [14]. For example, ideally a curve should not be split into two disjoint curves with the same label. Improving layout algorithms for iCon's GUI remains future work.

Acknowledgements This research was funded by a Leverhulme Trust Research Project Grant (RPG-2016-082) for the project entitled Accessible Reasoning with Diagrams. The authors would like to thank Prof. John Howse, Dr Andrew Blake and Dr Ryo Takemura for their cooperation in the experiments.

References

1. Baader, F., Horrocks, I., Sattler:, U.: Description logics. In: Handbook on Ontologies, pp. 21–43. Springer (2009)
2. Barwise, J., Etchemendy, J.: Hyperproof. CSLI Publications (1994)
3. Beckert, B., Grebing, S., Böhl, F.: A usability evaluation of interactive theorem provers using focus groups. In: Software Engineering and Formal Methods. LNCS, vol. 8938, pp. 3–19. Springer (2015)

4. Gil, J., Howse, J., Kent, S.: Formalizing spider diagrams. In: IEEE Symposium on Visual Languages. pp. 130–137. IEEE (1999)
5. Harrison, J., Urban, J., Wiedijk, F.: History of interactive theorem proving. In: Computational Logic, vol. 9, pp. 135–214. Elsevier (2014)
6. Hou, T., Chapman, P., Blake, A.: Antipattern comprehension: An empirical evaluation. In: Formal Ontology in Information Systems. Frontiers in Artificial Intelligence, vol. 283, pp. 211–224. IOS Press (2016)
7. Howse, J., Stapleton, G., Flower, J., Taylor, J.: Corresponding regions in Euler diagrams. In: Diagrammatic Representation and Inference. LNCS, vol. 2317, pp. 76–90. Springer (2002)
8. Jamnik, M.: Mathematical Reasoning with Diagrams. CSLI Publications (2001)
9. Kortenkamp, U., Richter-Gebert, J.: Using automatic theorem proving to improve the usability of geometry software. In: Mathematical User-Interfaces Workshop (2004)
10. Linker, S., Burton, J., Blake, A.: Measuring user comprehension of inference rules in Euler diagrams. In: Diagrammatic Representation and Inference. LNCS, vol. 9781, pp. 32–39. Springer (2016)
11. Linker, S., Burton, J., Jamnik., M.: Tactical diagrammatic reasoning. In: International Workshop on User Interfaces for Theorem Provers. Electronic Proceedings in Theoretical Computer Science, vol. 239, pp. 29–42 (2016)
12. Nguyen, T.A.T., Power, R., Piwek, P., Williams, S.: Measuring the understandability of deduction rules for OWL. In: International Workshop on Debugging Ontologies and Ontology Mappings. pp. 1–12. Linköping Electronic Conference Proceedings (2012)
13. Paulson, L.C.: Isabelle - A Generic Theorem Prover (with a contribution by T. Nipkow), LNCS, vol. 828. Springer (1994)
14. Rodgers, P., Zhang, L., Purchase, H.: Wellformedness properties in Euler diagrams: Which should be used? IEEE Transactions on Visualization and Computer Graphics 18(7), 1089–1100 (2012)
15. Sato, Y., Mineshima, K.: How diagrams can support syllogistic reasoning: An experimental study. Journal of Logic, Language and Information 24(4), 409–455 (2015)
16. Sato, Y., Ueda, K., Wajima, Y.: Strategy analysis of non-consequence inference with Euler diagrams. Journal of Logic, Language and Information. (2017)
17. Shams, Z., Jamnik, M., Stapleton, G., Sato, Y.: Reasoning with concept diagrams about antipatterns in ontologies. In: Intelligent Computer Mathematics. LNCS, vol. 10383, pp. 255–271. Springer (2017)
18. Shimojima, A.: Semantic Properties of Diagrams and Their Cognitive Potentials. CSLI Publications (2015)
19. Stapleton, G., Compton, M., Howse, J.: Visualizing OWL 2 using diagrams. In: IEEE Symposium on Visual Languages and Human-Centric Computing. pp. 245–253. IEEE (2017)
20. Stapleton, G., Howse, J., Chapman, P., Delaney, A., Burton, J., Oliver, I.: Formalizing concept diagrams. In: Visual Languages and Computing. pp. 182–187. Knowledge Systems Institute (2013)
21. Stapleton, G., Zhang, L., Howse, J., Rodgers, P.: Drawing Euler diagrams with circles: The theory of piercings. IEEE Transactions on Visualization and Computer Graphics 17(7), 1020–1032 (2011)
22. Urbas, M., Jamnik, M., Stapleton, G.: Speedith: A reasoner for spider diagrams. Journal of Logic, Language and Information 24(4), 487–540 (2015)