Evaluating Diagrammatic Patterns for Ontology Engineering

Eisa Alharbi, John Howse, Gem Stapleton, and Ali Hamie

University of Brighton, Brighton, UK
{e.alharbi2,John.Howse,g.e.stapleton,a.a.hamie}@brighton.ac.uk

Abstract. Diagrammatic logics have been widely studied since Shin's seminal work on Venn diagrams in the 1990s. There have been significant theoretical advances alongside empirical work investigating their efficacy with respect to symbolic notations. However, we have little understanding about how to choose between syntactically different diagrams when formulating logical axioms. This paper sets out to provide insight into such choices. By appealing to ontology engineering, we identify commonly required semantic properties that require axiomatization. We systematically identify three different ways of axiomatizing these properties using diagrammatic patterns. One way does not use explicit quantification. The other ways both use explicit quantification but employ different diagrammatic devices to capture the required semantics. We evaluated these competing patterns by conducting an empirical study, collecting performance data. We conclude that avoiding explicit quantification, and representing the information purely diagrammatically, best supports task performance. As a result, users and designers of diagrammatic logics are guided towards avoiding explicit quantification where possible.

Keywords: ontologies, axioms, diagrammatic patterns, visualization

1 Introduction

Understanding how to effectively represent information using diagrams is a major research goal of the Diagrams community. The focus of this paper is diagrams designed for making logical statements, in a precise and unambiguous way. Whilst a lot of research has been done into the design and theoretical development of diagrammatic logics, stemming from Shin's seminal work [12], little attention has been given to how to choose between semantically equivalent, yet syntactically different, diagrams. This paper begins to address this knowledge gap, by empirically evaluating competing choices of diagrams for axiomatizing semantic properties. To ensure practical relevance, and therefore wider significance of our research, we identified ontology engineering as a major endeavour where axioms are routinely defined.

Ontologies help us to structure and reason about information and data; formally, an ontology is a collection of axioms. With the abundance of data available in this information age, ontology engineering is becoming increasingly important.

Many different specialists are involved in the development of ontologies including domain experts, software engineers, data analysts and lawyers. Some of these stakeholders are not adept at using the existing approaches to ontology engineering, which involve the use of formal languages such as description logic [2] and OWL, the Web Ontology Language [1]. This implies that diagrammatic approaches to ontology engineering have the potential to appeal to ontology engineers without formal training in logic. With this in mind, concept diagrams [6] were designed to be used as an accessible ontology engineering language, usable by more stakeholders. Therefore, concept diagrams provide an ideal notation with which to provide an understanding into the relative effectiveness of different ways of axiomatizing semantic properties. Although specifically designed for ontology engineering, concept diagrams can be used in any logical context for which they are suitably expressive. As with any logic, it is frequently the case that any axiomatizable property can be represented by a variety of syntactically different concept diagrams but how to choose between the different representations is not obvious. This paper sets out to provide guidance on how to choose between such competing representations.

We briefly introduce the syntax and semantics of concept diagrams in section 2. We identify commonly occurring ontology axioms in section 3 where we also define different styles of diagrammatic patterns for representing them. The design and execution of an empirical study to determine which pattern styles are most accurately and most quickly interpreted by participants is described in section 4. The analysis and results are presented in section 5. We discuss the results in section 6 and conclude in section 7. Details of the questions and training material used in the study, along with the raw data collected, can be found at `https://sites.google.com/site/eisamalharbi/DiagramsPatternsStudy`.

2 Concept Diagrams

We present a brief overview of the syntax and semantics of concept diagrams, particularly with reference to the features occurring in this paper; a more detailed description of this fully formalized logic can be found in [14]. Closed curves represent sets which are called *concepts* in description logic and *classes* in OWL. Therefore concept diagrams are based on Euler diagrams. Binary relations, called *properties* or *roles* in ontology engineering, are represented by arrows. Individuals, or elements, are represented by dots or, more generally, trees.

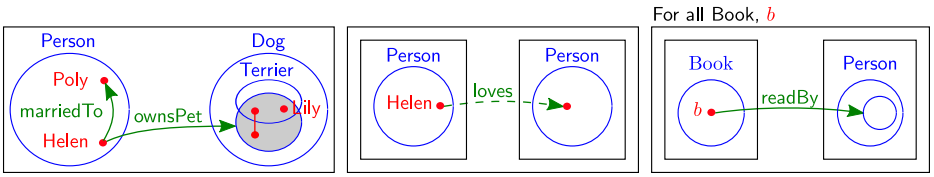


Fig. 1: Concept Diagrams

Suppose that the individual Helen is a Person who is married to only the Person Poly (identified by the binary relation marriedTo) and that Helen owns exactly two pets (identified by the binary relation ownsPet), both of which are Dogs. These two pets include a Terrier called Lily. The left-hand diagram in figure 1 expresses this information, requiring three closed curves to represent the concepts Person, Dog, and Terrier. Person and Dog are disjoint and Terrier is subsumed by Dog. The location of the dots identifies the concepts of which they are instances; for example, Lily is located inside the curve labelled Terrier. The fact that Helen owns a set of Pets is expressed by the arrow labelled ownsPet, which hits an unlabelled curve. This curve is drawn inside Dog, to assert that the image of the relation ownsPet, with its domain restricted to Helen, is subsumed by Dog. The two trees inside this unlabelled curve tell us that Helen owns two Dogs. Helen's dog that is not Lily *might* be a Terrier. This uncertainty is captured by the use of the unlabelled tree with two nodes, one inside both the Dog and Terrier curves and the other inside the Dog curve but outside the Terrier curve. Shading is used to express that the only dogs owned by Helen are represented by the trees.

Concept diagrams use dashed arrows to represent partial information, such as Helen loves some Person and that Person *could* be Helen herself. A concept diagram expressing this is in the middle diagram of figure 1. The arrow connects diagrammatic syntax placed in different boxes to ensure that we have not asserted that the Person Helen loves is different from Helen. The right-hand diagram of figure 1 expresses that every Book is readBy only a subset of Person. The quantification expression written outside of the rectangles tells us that the diagram is making an assertion about all books. Lastly, we note that concept diagrams can also make assertions involving inverse relations, by annotating arrow labels using the symbol $^-$, and negation by labelling a bounding box with 'Not'. These will be discussed in more detail in section 3.

3 Ontology Patterns

Concept diagrams are able to express commonly occurring ontology axioms in different ways. In this section we develop diagrammatic patterns for some types of axioms that commonly occur in ontology engineering: *subsumption*, *disjointness*, *All Values From*, *Some Values From*, *Domain* and *Range* [5].

3.1 Patterns Involving Only Classes

The *Subsumption* axiom type is one of the simplest and widely used. Class C_1 *subsumes* Class C_2 if all members of C_2 are also members of C_1. Diagrammatically there is a natural way of representing subsumption, shown in the left-hand diagram of figure 2.

The *Disjointness* axiom type is also widely used. Classes C_1 and C_2 are *disjoint* if no element of C_1 is also an element C_2. Again, there is a natural way of expressing disjointness shown in the second diagram in figure 2.

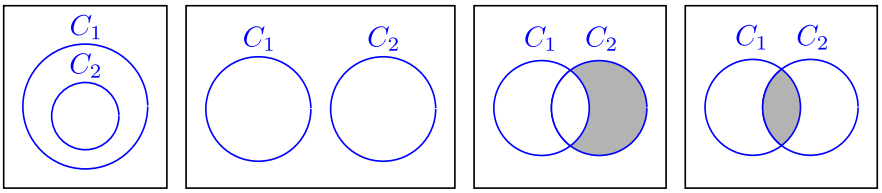


Fig. 2: *Subsumption* and *Disjointness* patterns

There are other ways of representing subsumption and disjointness using concept diagrams. For example, we could use shading to indicate that a region is empty; this is the way that Venn diagrams represent such properties. The two right-hand diagrams of figure 2 show alternative patterns for subsumption and disjointness involving the use of shading. It is well established that a salient feature of diagrams is *well-matchedness* [4]. A notation is *well-matched to meaning* when its syntactic relationships reflect the semantic relationships being represented. In the left-hand diagram of figure 2, the curve labelled C_2 is enclosed by the curve labelled C_1 matching the semantic interpretation that C_2 is a subset of C_1. Similarly, in the adjacent diagram, the curves labelled C_1 and C_2 are disjoint, reflecting the interpretation that C_1 and C_2 are disjoint sets. However, the right-hand diagrams are not well-matched. The closed curves intersect giving no indication of the relationship between the sets they represent. Moreover, the shading is purely symbolic [8, 13] and we have to learn that shaded regions represent the empty set. To confirm these theoretical insights, empirical studies have shown that users perform tasks more effectively when using well-matched Euler diagrams [3]. For these reasons, we recommend the well-matched subsumption and disjointness patterns for practical use by ontology engineers, and do not include them in our empirical study.

3.2 Patterns involving Classes and Properties

In ontology engineering, a property can be considered as a mathematical (binary) relation between two classes. When we consider axioms involving properties, it is not clear what is the best diagrammatic way to represent these constructs. We consider four constructs involving properties: *All Values From*, *Some Values From*, *Domain* and *Range*. For each of these constructs we have systematically identified three different styles of diagrammatic patterns:

1. *Unquantified*
2. *Quantified with Solid Arrow*
3. *Quantified with Dashed Arrow*

The *Unquantified* patterns were first developed in [15].

3.3 All Values From Patterns

The *All Values From* axiom type represents a constraint involving two classes and a property: if each element of class C_1 is related, under property p, only to

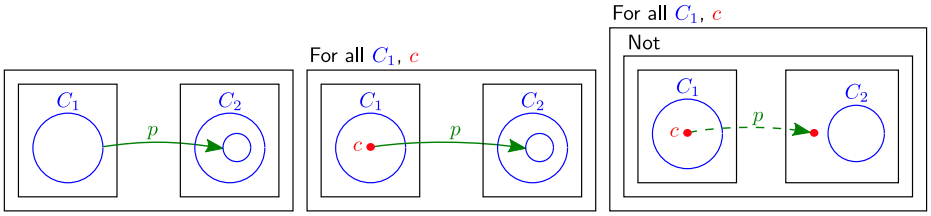


Fig. 3: *All Values From* patterns

elements of class C_2 (if it is related to anything), then C_1 is said to have *All Values From* C_2 under p.

Unquantified Pattern The left-hand diagram of figure 3 expresses that the image of property p, when its domain is restricted to C_1, is a subset of C_2. This axiomatizes the *All Values From* constraint. The closed curves representing classes C_1 and C_2 are each presented within a bounding rectangle because we do not want to express any relationship between C_1 and C_2.

Quantified with Solid Arrow Pattern The middle diagram of figure 3 expresses that for each c in C_1, the image of property p, when its domain is restricted to c, is a subset of C_2. Thus C_1 has *All Values From* C_2 under p.

Quantified with Dashed Arrow Pattern The right-hand diagram of figure 3 expresses that for each c in C_1, it is not the case that c is related, under p, to an element not in C_2. Thus no element of C_1 is related, under p, to an element not in C_2. Hence, each element of class C_1 is related, under property p, only to elements of class C_2.

3.4 Some Values From Patterns

The *Some Values From* axiom type also represents a constraint involving two classes and a property: if each element of class C_1 is related, under property p, to some element of class C_2, then C_1 has *Some Values From* C_2 under p.

Unquantified Pattern The left-hand diagram of figure 4 expresses that the image of property p^-, when its domain is restricted to C_2, includes C_1. Therefore, for each a in C_1, there exists b in C_2 such that b is related to a under p^-. Hence, for each a in C_1, there is some b in C_2 such that a is related to b under p.

Quantified with Solid Arrow Pattern The middle diagram of figure 4 expresses that for each c in C_1, the image of property p, when its domain is restricted to c, includes some element in C_2.

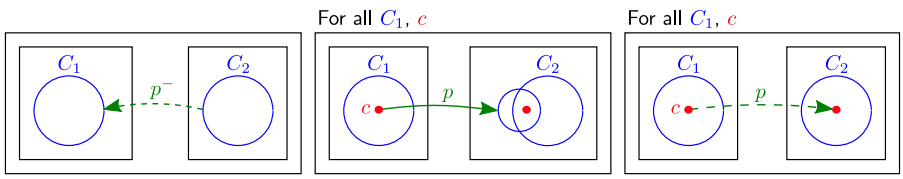


Fig. 4: *Some Values From* patterns

Quantified with Dashed Arrow Pattern The right-hand diagram of figure 4 expresses that each c in C_1 is related, under p, to some element in C_2.

3.5 Domain Patterns

The *Domain* axiom type represents a constraint involving a class and a property: Class C is the *Domain* of property p if only elements from C are related to *something* under p. Each pattern for *Domain* will use the inverse of property p.

Unquantified Pattern Noting that innermost rectangles represent the universal set, the left-hand diagram of figure 5 expresses that the image of property p^- is a subset of C. Hence, only elements in C are related to *something* by p.

Quantified with Solid Arrow Pattern The middle diagram of figure 5 expresses that for each *Thing* t, the image of property p^-, when its domain is restricted to t, is a subset of C. Hence, only elements in C are related to *something* under p.

Quantified with Dashed Arrow Pattern The right-hand diagram of figure 5 expresses that for each *Thing* t, it is not the case that t is related, by p^-, to an element not in C. Hence, only elements in C are related to *something* under p.

3.6 Range Patterns

The *Range* axiom type also represents a constraint involving a class and a property: Class C is the *Range* of property p if *things* are related, under p, only to elements in C.

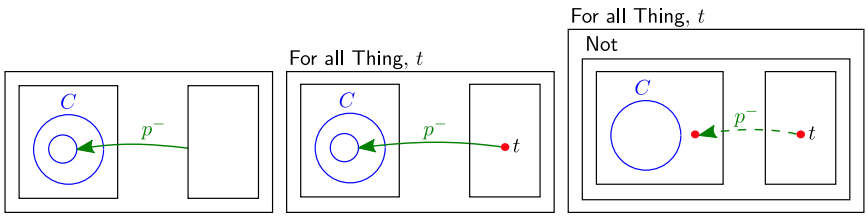


Fig. 5: *Domain* patterns

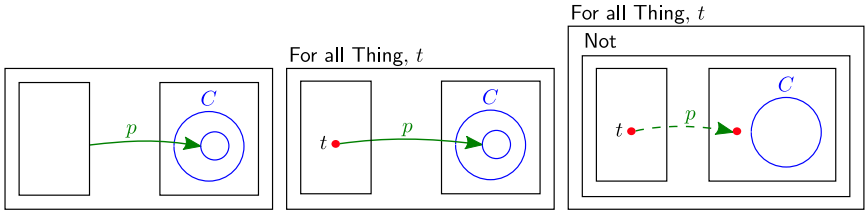


Fig. 6: *Range* patterns

Unquantified Pattern The left-hand diagram of figure 6 expresses that the image of property p is a subset of C. Hence, C is the *Range* of p.

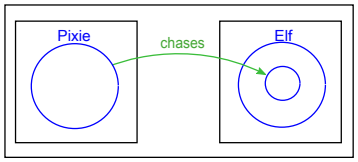
Quantified with Solid Arrow Pattern The middle diagram of figure 6 expresses that for each *Thing* t, the image of property p, when its domain is restricted to t, is a subset of C. Hence, C is the *Range* of p.

Quantified with Dashed Arrow Pattern The right-hand diagram of figure 6 expresses that for each *Thing* t, it is not the case that t is related, under p, to an element not in C. Hence, *things* are related, under p, only to elements in C.

4 Empirical Study

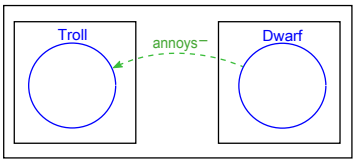
An empirical study was designed to determine which pattern style was more effective overall as well as for each of the four constructs, *All Values From*, *Some Values From*, *Domain* and *Range*. A pattern style was considered more effective than another if, on average, participants interpreted it with significantly fewer errors. If the pattern styles could not be distinguished on error rate then the pattern style that could be interpreted, on average, significantly more quickly was considered the most effective. In order to give some context to the questions used in the empirical study, a case study based on mythical creatures was developed. This context was chosen so that participants would be unable to guess the answers based on prior domain knowledge.

As described above, three different patterns for each of the four constructs were developed, giving a total of twelve different diagram patterns. Participants



	Options
1	Pixies chase only Elves.
2	Pixies chase at least one Elf.
3	At least one Pixie chases only Elves.
4	At least one Pixie chases Elves.

Fig. 7: Unquantified *All Values From* pattern with multiple-choice answers



	Options
1	At least one Troll annoys Dwarves.
2	Trolls annoy only Dwarves.
3	At least one Troll annoys only Dwarves.
4	Trolls annoy at least one Dwarf.

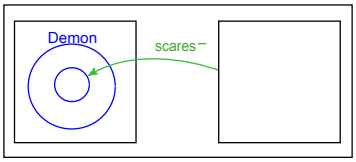
Fig. 8: Unquantified *Some Values From* pattern with multiple-choice answers

were shown 24 diagrams in total, with each different diagram pattern being shown twice, representing different information, in order to generate sufficient data points for statistical analysis. Each diagram was associated with a single question: "What does the diagram tell you?". The participants were provided with four multiple-choice options, presented in random order, exactly one of which was correct; the random order was the same for each participant. Figures 7, 8, 9 and 10 show example questions for each construct from the study, in each case the *Unquantified* pattern is used. The same multiple-choice options were used for each of the questions for each particular construct, but with the names changed to those given in the diagram (each diagram represented different information). The questions were presented in random order, generated uniquely for each participant. We set a time limit of two minutes to answer each question; attempts at the questions in the design phase by members of authors' research group indicated that the time taken to answer each question was usually much less than this. A time limit was deemed important so that the study did not continue indefinitely. We adopted a within-group design because there was unlikely to be any learning effect which could bias the results; each of the patterns has a different appearance and each diagram represented different information.

4.1 Experiment Execution

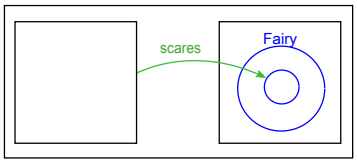
The experiment was performed within the university's usability laboratory, providing a quiet environment without interruption. Each participant was treated equally with the same environment, furniture, equipment, materials and procedures. Participants performed the experiment individually, and were provided with full details about the purpose of their role by an experiment facilitator who was present throughout.

At the beginning of the experiment, the facilitator introduced the participants to concept diagrams using paper-based training material. Participants were then



	Options
1	Only Demons scare things.
2	Not only Demons scare things.
3	Demons scare at least one thing.
4	At least one Demon scare things.

Fig. 9: Unquantified *Domain* pattern with multiple-choice answers



	Options
1	Things scare at least one Fairy.
2	Things scare only Fairies.
3	At least one thing scares Fairies.
4	At least one thing scares only fairy.

Fig. 10: Unquantified *Range* pattern with multiple-choice answers

given software training. They were shown three questions with a similar design to those in the main study in order to help familiarize them with the software's user interface. Finally, the facilitator allowed the participants to work on the study questions. Participants were able to refer to a hard copy single side of A4 paper detailing the elements of concept diagrams used in the study, which formed part of the training material. Upon completion of the experiment, each participant was provided with a debrief summary. Participants were offered a £6 canteen voucher for their time spent in the study, which was approximately 30 minutes.

A pilot study was conducted to test the experiment design, research software used to display the diagrams and questions, and the data collection process. Five participants (1F, 4M, ages 18-29) took part in the pilot study. As a result of the pilot, a minor change was made to the training material. Forty participants (12F, 28M, ages 18-38) participated in the main experiment, all students from the University of Brighton studying computing, mathematics or engineering. They reported no previous knowledge of concept diagrams, OWL or DL, but were familiar with Venn/Euler diagrams, first order logic and set theory.

5 Results

To determine whether there are differences between the interpretability of the three pattern styles, we analysed both errors and the time taken to answer each question. We performed this analysis on the pattern styles overall and separately for each of the four axiomatized constructs, *All Values From*, *Some Values From*, *Domain* and *Range*. For the errors, we performed chi-square tests. For the time analysis, we performed ANOVAs. However, as the time data were not normally distributed, we performed a log transformation to reduce the skewness to within tolerable levels for conducting robust ANOVAs. When the ANOVAs revealed significant differences, we proceeded to conduct Tukey tests to rank the pattern styles. The results are based on the data collected from 40 participants, with each participant answering 24 questions providing a total of 960 observations, 240 for each of the four axiom types and 80 for each diagrammatic pattern.

5.1 Overall Analysis

To determine which pattern style was most effective overall, we considered how the three pattern styles, *Unquantified*, *Quantified with Solid Arrow* and *Quanti-*

Table 1: Overall summary

	Error Analysis			Time Analysis		
Pattern	*N*	Errors	Rate %	*N*	Mean	StDev
Un	320	75	23.44	245	18.40	14.66
QwSA	320	89	27.81	231	20.89	14.85
QwDA	314	250	79.62	64	29.05	21.62

fied with Dashed Arrow, compared for the entire 24 questions. Firstly, we compared the error rates for each pattern style, which are summarised in Table 1; these data exclude six timeouts, and, thus, only include data from questions for which an answer was provided within the 2 minutes allowed. Conducting a chi-square test established that there was no significant difference in error rate between the *Unquantified* (Un) and *Quantified with Solid Arrow* (QwSA) pattern styles ($p = 0.205$). However, both of these pattern styles yielded significantly fewer errors than *Quantified with Dashed Arrow* (QwDA); in each case, $p = 0.000$. We can see that *Quantified with Dashed Arrow* yielded approximately 56 more errors for every 100 answers than *Unquantified*, which falls to 52 more errors as compared to *Quantified with Solid Arrow*.

To further distinguish the pattern styles, we analysed the time data. Consistent with Meulemans et al. [7], we only analyze the correct answers; it can be argued that it does not matter how long it takes to provide a wrong answer. The mean times and standard deviations are summarised in Table 1; these data are from questions for which a *correct* answer was provided within the 2 minutes allowed. Conducting an ANOVA test established that there were significant differences in the times taken between the three pattern styles ($p = 0.000$). To expose the nature of these differences, we proceeded to conduct a Tukey test in order to rank the pattern styles. This revealed that the *Unquantified* pattern style allowed participants to perform significantly faster than *Quantified with Solid Arrow* which, in turn, was significantly faster than *Quantified with Dashed Arrow*. In terms of time taken, we see that *Unquantified* is approximately 13.5% faster than *Quantified with Solid Arrow* and approximately 57.9% faster than *Quantified with Dashed Arrow*, on average.

Combining both our error analysis and time analysis, we conclude that using the *Unquantified* pattern style significantly improves overall task performance, as compared to the other two pattern styles.

5.2 All Values From Analysis

Table 2 summarizes the error rates for each pattern; these data exclude a single timeout which was for *Quantified with Dashed Arrow*. Conducting a chi-square test showed that there was no significant difference between *Unquantified* and *Quantified with Solid Arrow*, with $p = 0.548$. However, *Unquantified* and *Quantified with Solid Arrow* both yielded significantly fewer errors than *Quantified with Dashed Arrow*, with $p = 0.000$ in each case. *Quantified with Dashed Arrow*

Table 2: *All Values From* summary

	Error Analysis			Time Analysis		
Pattern	*N*	Errors	Rate %	*N*	Mean	StDev
Un	80	5	6.25	75	15.34	11.99
QwSA	80	7	8.75	73	18.57	11.53
QwDA	79	67	84.81	12	28.75	23.38

Table 3: *Some Values From* summary

	Error Analysis			Time Analysis		
Pattern	*N*	Errors	Rate %	*N*	Mean	StDev
Un	80	53	66.25	27	24.20	15.14
QwSA	80	67	83.75	13	38.75	25.83
QwDA	80	52	65.00	28	28.94	23.33

yielded approximately 79 more errors for every 100 answers than *Unquantified*, which fell to 76 more errors as compared to *Quantified with Solid Arrow*.

The mean times and standard deviations for each pattern style are given in Table 2. An ANOVA test revealed that there were significant differences ($p = 0.005$) between the pattern styles. A Tukey test indicated that *Unquantified* was significantly faster than *Quantified with Solid Arrow* which, in turn, was significantly faster than *Quantified with Dashed Arrow*. We can see that *Unquantified* was approximately 87.4% faster than *Quantified with Dashed Arrow* and approximately 21.1% faster than *Quantified with Solid Arrow*, on average.

Combining the error and time analysis, we again conclude that the *Unquantified* pattern style significantly improves task performance, as compared to the other two pattern styles, in the case of *All Values From*.

5.3 Some Values From Analysis

Table 3 summarizes the error rates for each pattern; there were no timeouts. A chi-square test found no significant difference between *Unquantified* and *Quantified with Dashed Arrow*, with $p = 0.868$. However, *Unquantified* and *Quantified with Dashed Arrow* both yielded significantly fewer errors than *Quantified with Solid Arrow*, with $p = 0.011$ and $p = 0.007$ respectively. *Quantified with Solid Arrow* yielded approximately 18 or 19 more errors for every 100 answers than both *Unquantified* and *Quantified with Dashed Arrow*.

The mean times and standard deviations for each pattern style are given in Table 3. An ANOVA test revealed that there were no significant differences ($p = 0.167$) between the pattern styles. Therefore, we did not proceed to conduct a Tukey test. We conclude, on the basis of the error analysis, that using either the *Unquantified* or *Quantified with Dashed Arrow* best supports task performance for *Some Values From* axioms.

5.4 Domain Analysis

Table 4 summarizes the error rates for each pattern; there were three timeouts, all for *Quantified with Dashed Arrow*. A chi-square test found no significant difference between *Unquantified* and *Quantified with Solid Arrow*, with $p = 0.442$. However, *Unquantified* and *Quantified with Solid Arrow* both yielded significantly fewer errors than *Quantified with Dashed Arrow*, with $p = 0.000$ in each case. *Quantified with Dashed Arrow* yield approximately 82 more errors for every

Table 4: *Domain* summary

	Error Analysis			Time Analysis		
Pattern	*N*	Errors	Rate %	*N*	Mean	StDev
Un	80	10	12.50	70	24.52	18.10
QwSA	80	7	8.75	73	24.42	16.41
QwDA	77	70	90.91	7	44.48	23.76

Table 5: *Range* summary

	Error Analysis			Time Analysis		
Pattern	*N*	Errors	Rate %	*N*	Mean	StDev
Un	80	7	8.75	73	13.54	10.04
QwSA	80	8	10.00	72	16.44	9.84
QwDA	78	61	78.21	17	23.08	13.88

100 answers than *Quantified with Solid Arrow* which slightly reduces to 78 more errors as compared to *Unquantified*.

The mean times and standard deviations for each pattern style are given in Table 4. An ANOVA test revealed that there were significant differences ($p = 0.041$) between the pattern styles. Therefore, we conducted a Tukey test, which ranked the pattern styles as follows: *Unquantified* and *Quantified with Solid Arrow* were not significantly different, but both were significantly faster than *Quantified with Dashed Arrow*. We can see that *Unquantified* and *Quantified with Solid Arrow* were approximately 81.4% and 82.1%, respectively, faster than *Quantified with Dashed Arrow*, on average.

Our error and time analysis consistently support the use of either *Unquantified* and *Quantified with Solid Arrow* over *Quantified with Dashed Arrow* for *Domain* axioms.

5.5 Range Analysis

Table 5 summarizes the error rates for each pattern; there were two timeouts, both for *Quantified with Dashed Arrow*. A chi-square test found no significant difference between *Unquantified* and *Quantified with Solid Arrow*, with $p = 0.786$. Again, we found that both *Unquantified* and *Quantified with Solid Arrow* yielded significantly fewer errors than *Quantified with Dashed Arrow*, with $p = 0.000$ in each case. *Quantified with Dashed Arrow* yield approximately 68 or 69 more errors for every 100 answers than *Unquantified* and *Quantified with Solid Arrow*.

The mean times and standard deviations for each pattern style are given in Table 5. An ANOVA test revealed that there were significant differences ($p = 0.001$) between the pattern styles. A Tukey test ranked the patterns as follows: *Unquantified* was significantly faster than *Quantified with Solid Arrow* which, in turn, was significantly faster than *Quantified with Dashed Arrow*. Participants' performance using the *Unquantified* pattern was approximately 70% faster than *Quantified with Dashed Arrow* and approximately 21.4% faster than *Quantified with Solid Arrow*, on average.

Drawing on the error and time analysis, for *Range* using the *Unquantified* pattern most effectively supports user task performance and *Quantified with Solid Arrow* is placed second.

5.6 Summary of Analysis

As well as being ranked as the most effective overall, the *Unquantified* pattern style allows participants to perform at least as well, if not significantly better,

than both the other pattern styles for each individual axiom type. Interestingly, in all but one case – namely *Some Values From – Quantified with Dashed Arrow* was ranked last by both errors and time taken. This indicates that using quantification with dashed arrows is particularly poor for cognition and an overall error rate of 79.62% is not dissimilar to what is expected when randomly choosing one out of four options. Given this and that the overall error rates for the other two patterns styles are much lower, being 23.44% for *Unquantified* and 27.81% for *Quantified with Solid Arrow*, it is surprising that *lowest* error rate for *Quantified with Dashed Arrow* is for the axiom type *Some Values From* at 65%. The other two styles have, by far, their *highest* error rates for this axiom type, namely 66.25% and 83.75%. These high error rates are, however, consistent with findings for symbolic approaches to ontology engineering, where it has been established that users have particular difficulty understanding *Some Values From* axioms [5, 9–11, 16]. We will further discuss this observation in section 6.

6 Discussion

There are some interesting observations to be made from the results of the empirical study. In particular, the results for *Some Values From* are striking, with an overall error rate of 72.67%. As just stated, it is well established that users have difficulties with this construct, so it may be the inherent conceptual difficulty of this axiom type that causes the high error rate. For example, Rector et al. [9, 10] showed that new OWL students do not understand the exact meaning of *Some Values From*, and "are unsure if it means *all*, *any* or *nothing else*".

Delving deeper into the results of our study, we analysed the incorrect responses for the *Some Values From* construct. For the *Quantified with Solid Arrow* pattern 60 out of 80 (75%) participants confused *Some Values From* with *All Values From*, for example, choosing "Trolls recruit only Goblins" rather than "Trolls recruit at least one Goblin". This is in agreement with other studies that report that users confuse *Some Values From* with *All Values From* [9, 10]. In particular, one of the common logical errors made by ontology users is that C_1 has *All Values From* C_2 implies C_1 has *Some Values From* C_2. Furthermore, Schwitter and Tilbrook [11] showed that one of the common errors new OWL users make is to use the universal restriction *All Values From* as a default, when the existential restriction *Some Values From* actually applies. Interestingly, in our study, confusing *Some Values From* with *All Values From* was not the case for the *Unquantified* and *Quantified with Dashed Arrow* patterns: only three out of 80 (3.7%) in both cases chose the *All Values From* option.

Other studies have also shown description logic users may interpret *Some Values From* incorrectly; for example, considering the pizza ontology, many users initially read, 'Pizza hasTopping MozzarellaTopping' to mean "some pizzas have toppings that are mozzarella topping", rather than the correct reading, "all pizzas have toppings that are some mozzarella topping" [5]. In our study, 41 out of 80 (51.25%) for the *Unquantified* pattern and 36 out of 80 (45%) for the *Quantified with Dashed Arrow* pattern made the same mistake as reported in the pizza

example, choosing, for example, 'at least one Troll recruits Goblins' (equivalent to 'some Trolls recruit Goblins'). The reasons for this kind of misunderstanding are not clear, although it may have been that participants associated 'at least one' with the wrong class.

Moving on to consider the other constructs, it is surprising that there were differences in the results for *Domain* and *Range* in that they are diagrammatically 'mirror images' of each other. The difference, that there was a statistically significant best pattern for *Range* but not for *Domain*, could be explained by the use of the conceptually more difficult inverse property in *Domain*.

The results for the *Quantified with Dashed Arrow* pattern style were also striking, with an overall error rate of 79.62%. This seems to imply that using a dashed arrow may be difficult to interpret. However, it may not be the dashed arrow but the other features of these patterns that cause problems. Three of the *Quantified with Dashed Arrow* patterns used explicit negation; no other pattern style used negation. All of the *Quantified with Dashed Arrow* patterns that involved negation performed badly, with error rates of 83.75% for *All Values From*, 90.91% for *Domain* and 78.21% for *Range*. All six of the timeouts were for patterns involving negation, and each pattern involving negation had at least one timeout. By contrast, the *Quantified with Dashed Arrow* pattern that did not involve negation, for *Some Values From*, had the lowest absolute error rate at 65.00% for this construct. In cognitive psychology, it is well-known that human reasoning with negation is harder than reasoning without [17]. We therefore conjecture that negation is a major contributor to the poor task performance observed for the *Quantified with Dashed Arrow* pattern styles.

Other factors may have influenced the results. The relative complexity of the diagrams could have an effect on performance. The two quantified styles are diagrammatically more complex than the unquantified style. They could also be considered as heterogeneous, in that they contain textual notation, rather than purely diagrammatic. This may be why the unquantified patterns are easier to deal with cognitively. Similarly, the unquantified styles may be better matched to meaning than the quantified styles. Little work has been carried out on well-matchedness in diagrammatic notations more expressive than Euler diagrams.

7 Conclusion

The aim of this paper was to provide insight into how to choose between syntactically different diagrams when formulating logical axioms, particularly from the perspective of ontology engineering. In the context of this empirical study, we conclude that avoiding explicit quantification and representing the information purely diagrammatically best supports task performance. As a result, we recommend that users and designers of diagrammatic logics, and in particular ontology engineers, avoid using explicit quantification where possible.

Having made this recommendation, it is important to determine whether there really is an advantage in using diagrammatic patterns over standard notations in ontology engineering. Further work is needed to empirically evaluate

the recommended patterns from this paper, that is the *Unquantified* patterns for *Subsumption*, *Disjointness*, *All Values From*, *Some Values From*, *Domain* and *Range*, with equivalent axioms expressed in OWL and description logic. This will allow us to determine whether there is an advantage in performance when using these diagrammatic patterns over equivalent textual or symbolic representations. Further work is also required to determine whether it is negation that is causing poor task performance.

References

1. The OWL 2 Web Ontology Language. http://www.w3.org/TR/owl2-overview/ (2016), accessed April 2016
2. Baader, F., Calvanese, D., McGuinness, D., Nadi, D., (eds), P.P.S.: The Description Logic Handbook. CUP (2003)
3. Chapman, P., Stapleton, G., Rodgers, P., Micallef, L., Blake, A.: Visualizing sets: An empirical comparison of diagram types. In: Diagrams 2014. pp. 146–160. Springer (2014)
4. Gurr, C.: Effective diagrammatic communication: Syntactic, semantic and pragmatic issues. Journal of Visual Languages and Computing 10(4), 317–342 (1999)
5. Horridge, M., Drummond, N., Goodwin, J., Rector, A.L., Stevens, R., Wang, H.: The Manchester OWL syntax. In: OWLed. vol. 216 (2006)
6. Howse, J., Stapleton, G., Taylor, K., Chapman, P.: Visualizing ontologies: A case study. In: International Semantic Web Conference. pp. 257–272. Springer (2011)
7. Meulemans, W., Henry Riche, N., Speckmann, B., Alper, B., Dwyer, T.: Kelpfusion: A hybrid set visualization technique. IEEE Transactions on Visualization and Computer Graphics 19(11), 1846–1858 (2013)
8. Peirce., C.: Collected Papers, vol. 4. Harvard University Press (1933)
9. Rector, A., Drummond, N., Horridge, M., Rogers, J., Knublauch, H., Stevens, R., Wang, H., Wroe, C.: Owl pizzas: Practical experience of teaching owl-dl: Common errors & common patterns. In: Engineering Knowledge in the Age of the Semantic Web, pp. 63–81. Springer (2004)
10. Rector, A.L., Drummond, N., Horridge, M., Rogers, J., Knublauch, H., Stevens, R., Wang, H., Wroe, C.: Designing user interfaces to minimise common errors in ontology development: The co-ode and hyontuse projects. In: Proceedings of the UK e-Science All Hands Meeting. vol. 2004, pp. 493–499 (2004)
11. Schwitter, R., Tilbrook, M.: Controlled natural language meets the semantic web. In: Proc. Australasian Language Technology Workshop. vol. 2004, pp. 55–62 (2004)
12. Shin, S.J.: The Logical Status of Diagrams. Cambridge University Press (1994)
13. Shin, S.J.: The Iconic Logic of Peirce's Graphs. Bradford Book (2002)
14. 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)
15. Stapleton, G., Howse, J., Taylor, K., Delaney, A., Burton, J., Chapman, P.: Towards Diagrammatic Ontology Patterns. In: 4th Workshop on Ontology and Semantic Web Patterns. CEUR, Sydney, Australia (Oct 2013)
16. Warren, P., Mulholland, P., Collins, T., Motta, E.: The Usability of Description Logics. In: The Semantic Web: Trends and Challenges. LNCS, vol. 8465, pp. 550–564. Springer (2014)
17. Wason, P., Johnson-Laird, P.: Psychology of Reasoning: Structure and Content. Harvard University Press (1972)