Designing Inference Rules for Spider Diagrams

Gem Stapleton
University of Brighton
Email: g.e.stapleton@brighton.ac.uk

Mateja Jamnik and Matej Urbas
University of Cambridge
Email: {mateja.jamnik,matej.urbas}@cl.cam.ac.uk

***Abstract*—Diagrammatic modes of communication have long been recognized for their accessible representations of information. One area in which they have been developed is that of logical reasoning, where symbolic notations are perceived by many as difficult to use. Significant progress has been made on formalizing diagrammatic logics and proving formal properties of their inference rules. To-date, most inference rules for diagrammatic logics have been designed from the perspective of a logician, aiming for the essential and, thus, desirable properties of soundness and completeness. However, this approach overlooks a fundamental goal of providing diagrammatic logics: to overcome barriers posed by symbolic logics to non-mathematicians. Even if the diagrams themselves are accessible, having inference rules that result in unwieldy proofs will fail to fulfil this fundamental goal. Thus, the time is ripe to fully address this goal and show how to design inference rules that give rise to more natural proofs. In this paper we take significant steps towards this ambitious target by devising new inference rules for spider diagrams. We demonstrate that they allow substantially shorter proofs to be written and, we argue, the resulting proofs are more natural.**

I. Introduction

The diagrams community has a longstanding aim to make modelling and inference easier for people, particularly for those who do not have mathematical training. Unlike alternative representations, such as symbolic or linguistic, the use of diagrams as external representations of information can give people a substantially increased ability to produce appropriate scientific ideas [1]. In the case of diagrammatic logics, it has been shown that Euler diagrams lead to better understanding and ability to carry out inference tasks than symbolic approaches [2]. This supports the ambition to make diagrams a suitable notation for modelling and inference, and addresses the problem that symbolic formalisms, such as predicate and description logics, are "difficult for most people" [3].

Diagrammatic logics are visual languages that have a set of inference rules. The last two decades have seen many of them successfully developed. In seminal work, Shin [4] produced the first sound and complete diagrammatic logic for a system based on Venn diagrams augmented with Peirce's x-sequences [5] with the expressiveness of monadic first-order logic. Around the same time, Hammer devised a sound and complete Euler diagram logic which had just three inference rules [6]. Other key examples include Peirce's existential graphs [5], further developed by both Shin [7] and Dau [8].

Our knowledge about how to formalize diagrammatic logics has, since those early days, considerably advanced. Typically, they are formally defined via an abstract syntax [9] and given a model theoretic semantics. Using an abstract syntax was found to overcome problems associated with attempting to reason about the logic at the concrete syntax level [10] (i.e., reasoning with the actual drawn diagrams) [11]. This approach can be seen in Kent's constraint diagrams [12], [13], Euler diagrams [14], Swoboda and Allwein's Euler/Venn logic [15] and, most relevant to this paper, Gil et al.'s spider diagrams [16].

These logics have had their inference rules developed by aiming for the existence of properties such as soundness and completeness. Whilst soundness is essential and completeness is highly desirable, little attention has been paid to the impact of inference rule choice on proof length or readability. One exception is the work on Euler diagrams in [17], where different sets of sound and complete inference rules were developed with the aim of finding rules that made automated proof search more efficient. A consequence was that some sets of inference rules produced shorter proofs than others.

Given that a primary motivation for developing diagrammatic modes of inference is to overcome the communication barriers presented by symbolic logics, the time is right for devising inference rules that address this aim. In this paper we do just that: we present new inference rules for spider diagrams that, we argue, allow for more natural proof steps to be made whilst ensuring soundness. Section II presents basic definitions needed throughout the paper. Motivation for developing new rules is given in section III. Section IV develops a theory of corresponding regions for spider diagrams that is necessary for the definitions of the new inference rules which are presented in section V. Section VI briefly describes their impact on proof length. Finally, we conclude in section VII.

II. Preliminary Definitions

We now present core definitions relating to spider diagrams, introduced in [16] and further developed in [18]. The spider diagram in Fig. 1 comprises two *unitary* diagrams, d_1 and d_2, joined with the $\wedge$ logical operator. The diagram d_1 expresses that the sets A and B are disjoint: the two *contours* (closed curves) labelled A and B have disjoint interiors. There are two elements in the set A; this is achieved through the use of *spiders* (the nodes which, in general, are trees) drawn inside the contour labelled A. There is exactly one element in B; this is achieved through the use of the spider, representing the existence of an element, together with the shading which tells us that no more elements exist. The diagram d_2 expresses $C \subseteq A$, since the contour labelled C is inside A. In these diagrams, the regions formed from the contours are called *zones*; d_2 has three zones, one of which is inside A but outside C and can, therefore, be identified by the pair $(\{A\}, \{C\})$.

We now define the abstract syntax of spider diagrams together with a model theoretic semantics. The labels used are drawn from a set $\mathcal{L}$. A **zone**, the set of which is denoted $\mathcal{Z}$, is

978-1-4799-0369-6/13/$31.00 ©2013 IEEE

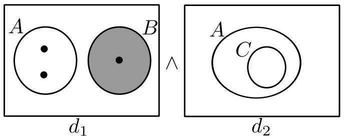

Fig. 1. A spider diagram formed of two unitary diagrams.

a pair of finite, disjoint sets, (in, out), such that $in \cup out \subseteq \mathcal{L}$. A **region** is a set of zones. Spiders are drawn from a countably infinite set, $\mathcal{S}$. In the following, $\mathbb{P}$ denotes power set.

Definition 1: A **unitary spider diagram** is a tuple, $d = (L, Z, ShZ, S, \eta)$, where

1) $L = L(d)$ is a subset of $\mathcal{L}$;
2) $Z = Z(d)$ is a set of zones such that each zone, (in, out), in $Z(d)$ forms a partition of $L(d)$;
3) $ShZ = ShZ(d)$ is a set of shaded zones in d such that $ShZ(d) \subseteq Z(d)$;
4) $S = S(d)$ is a subset of $\mathcal{S}$;
5) $\eta = \eta_d$ is a function, $\eta_d \colon S(d) \to \mathbb{P}Z(d) - \{\emptyset\}$, that returns the **habitat** of each spider.

A **spider diagram** is defined as follows:

1) Every unitary spider diagram is a spider diagram.
2) The symbol $\perp$ is a spider diagram.
3) If d_1 and d_2 are spider diagrams then so are $d_1 \wedge d_2$ and $d_1 \vee d_2$.

The definition that we have given of a spider diagram is different from that in [18]. Our approach for unitary diagrams uses a set of spiders together with a function mapping them to their habitats. By contrast, [18] just recorded how many spiders are placed in each region. For our purposes, it is helpful to have access to actual spiders, since we need to compare them when defining inference rules. Having defined the syntax, we can now define the semantics of spider diagrams.

Definition 2: An **interpretation** is a pair, $I = (U, \Psi)$, where U is a set and $\Psi \colon \mathcal{L} \to \mathbb{P}U$ assigns a subset of U to each label. An extension of Ψ to zones and regions is as follows. For each zone, (in, out),

$$\Psi(in, out) = \bigcap_{l \in in} \Psi(l) \cap \bigcap_{l \in out} (U - \Psi(l)).$$

For each region, r, $\Psi(r) = \bigcup_{z \in r} \Psi(z)$.

Whenever we have an interpretation, we assume that we have access to the extension to zones and regions. We now define when an interpretation is a model for a diagram. Our approach again differs in presentation from that in [18], in part because of the way we handle spiders at the syntactic level, but both approaches are equivalent: the same interpretations are identified as models. In the following definition the image of the function ψ is denoted $im(\psi)$.

Definition 3: Let $I = (U, \Psi)$ be an interpretation. Let d be a spider diagram. We say that I is a **model** for d under the following circumstances. If d is a unitary diagram then I is a model whenever the zones, between them, represent the universal set, that is $\Psi(Z(d)) = U$, and there exists an injection, $\psi \colon S(d) \to U$, such that each spider, s, represents an element in the set represented by its habitat, that is, $\psi(s) \in \Psi(\eta_d(s))$, and each shaded zone, z, represents a set containing only elements represented by spiders, that is, $\Psi(z) \subseteq im(\psi)$. If d is not unitary then the definition of a model extends in the obvious inductive manner, where $\perp$ has no models.

For example, the interpretation $I = (U, \Psi)$ partially defined by $U = \{1, 2, 3, 4\}$, $\Psi(A) = \{1, 2, 3\}$, $\Psi(B) = \{4\}$ and $\Psi(C) = \{2, 4\}$ is a model for d_1 in Fig. 1 but not for d_2. Therefore I is not a model for $d_1 \wedge d_2$ as depicted in the figure but instead, I is a model for $d_1 \vee d_2$.

III. Limitations of Current Inference Rules

Here we highlight inadequacies with the existing inference rules for spider diagrams and thus other logics based on Euler diagrams. We present a proof written using the spider diagram inference rules given in [18]. We argue that these inference rules result in unwieldy proofs.

In Fig. 2, $d_1 \wedge d_2$ is semantically equivalent to $d_1' \wedge d_2$. From d_1 we can see that $A \cap B = \emptyset$ and that $B - C = \emptyset$. From d_2, we can see $D \subseteq B$. We can deduce that $D \subseteq (B \cap C) - A$, shown in d_1'. In d_1, the spider inside both B and C can represent an element, e, where $e \in D$ or $e \notin D$, reflected by its new habitat in d_1'. We argue that a natural proof step would allow us to 'copy' D into d_1 to give d_1', since we can 'see' information about D in d_2. The question arises as to how we can use inference rules to prove that $d_1 \wedge d_2$ is semantically equivalent to $d_1' \wedge d_2$. We now demonstrate how to use the sound and complete inference rules for spider diagrams given in [18] to prove this equivalence. We start by showing that $d_1 \wedge d_2$ logically entails $d_1' \wedge d_2$, that is, $d_1 \wedge d_2 \vdash d_1' \wedge d_2$. The proof that $d_1' \wedge d_2 \vdash d_1 \wedge d_2$ is much more straightforward.

There is only a small difference between d_1 and d_1' and the proof task requires us to transform d_1 into d_1'. In particular, d_1' is a copy of d_1 but with D added to reflect the information we have about D. It is surprisingly complicated to establish logical entailment. We start by observing that the *only* inference rule in [18] that allows the addition of a contour is the following:

Rule 1: (Add a contour) Let d_1 be a unitary diagram and let l be a contour label not used in d_1. Let d_2 be a copy of d_1 except that there is a new contour labelled l in d_2, added so that all zones are split by the new contour and that each zone in any spider's habitat becomes two zones in the new habitat in the obvious way (so each spider has twice as many feet as in the original diagram). Then d_1 is logically equivalent to d_2.

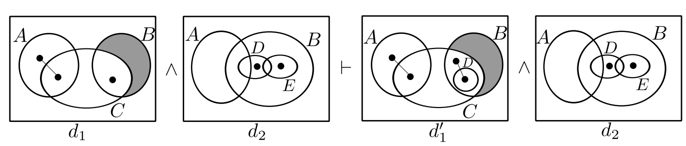

Fig. 2. A proof task.

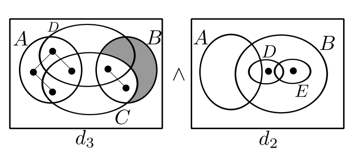

Fig. 3. Adding the contour D to d_1.

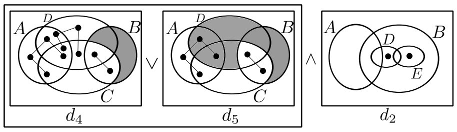

Fig. 4. Applying the excluded middle rule to d_3.

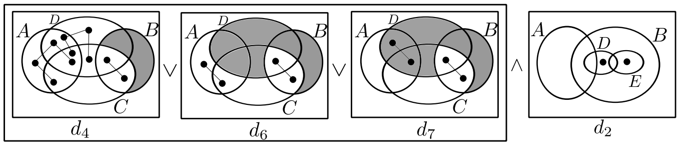

Fig. 5. Applying the splitting spiders rule to d_5.

Adding contour D to d_1 results in d_3, Fig. 3, giving the first step in a proof that $d_1 \wedge d_2 \vdash d_1' \wedge d_2$. Our task now is to show that we can move D so it is inside both B and C. This process begins by applying the *excluded middle* rule to $d_3 \wedge d_2$ giving $(d_4 \vee d_5) \wedge d_2$, Fig. 4, giving our second proof step. This rule adds a spider to a region, r, shown in d_4, and shades the same region, shown in d_5: either there is an element in the set represented by r that is not represented by a spider (so we can add a spider to r) or all elements are represented by spiders (so we can shade r). Informally, the rule is:

Rule 2: (Excluded Middle) Let d be a unitary diagram and let r be a region in d that is completely non-shaded. Let d_1 and d_2 be copies of d except that a new spider, s, has habitat r in d_1 and r is shaded in d_2. Then d is logically equivalent to $d_1 \vee d_2$.

Examining Fig. 4, we can intuitively see that d_4 and d_2 are in contradiction: d_4 asserts that $D - B \neq \emptyset$ whereas d_2 asserts $D \subseteq B$ (so $D - B = \emptyset$); we return to the contradiction later. Moreover, d_5 can be transformed into d_1' by removing the two nodes of the spider that are inside D but outside B and removing some of the shaded zones. The removal of these spider nodes first requires an application of the so-called *splitting spiders* inference rule from [18]:

Rule 3: (Splitting Spiders) Let d be a unitary diagram and let s be a spider whose habitat, $\eta_d(s)$ can be partitioned into two disjoint, non-empty regions, r_1 and r_2 say. Let d_1 and d_2 by copies of d except that s has habitat r_1 in d_1 and r_2 in d_2. Then d is logically equivalent to $d_1 \vee d_2$.

The result of applying the splitting spiders rule can be seen in Fig. 5, and this is the third step in our proof. The diagram d_6 can readily be transformed into d_1' using an inference rule that allows shaded zones which do not contain spiders to be removed: five applications of this rule give d_1'. The result of applying this inference rule 5 times can be seen in Fig. 6, so the proof we are constructing is now 8 steps long.

The question remains as to how to remove d_4 and d_7 from the diagram $(d_4 \vee d_1' \vee d_7) \wedge d_2$. Intuitively, d_4 and d_7 are both in contradiction with d_2, since they assert that there is an element in D that is not in B. There is only one inference rule in [18] that enables us to identify contradictions between two unitary diagrams. It requires (a) the zone sets to be made identical and (b) *all* of the spiders to have single zone habitats. To make the zone sets equal, the contour sets must first be equalized. In this example, we can remove contours, without

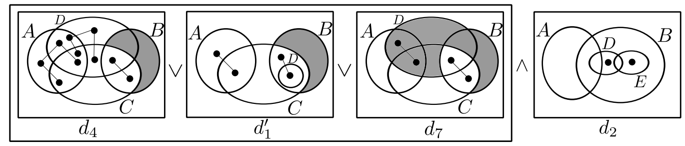

Fig. 6. Removing five zones from d_6.

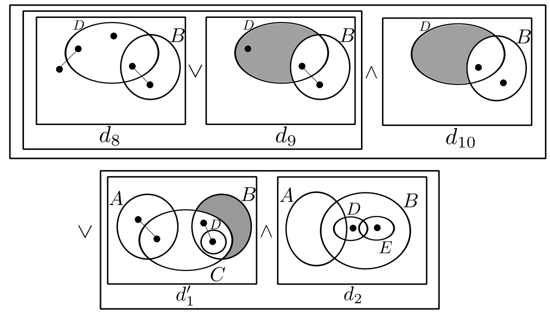

Fig. 7. Removing contours and equalizing zones.

losing the contradictory information. This contour removal has the added advantage that it reduces the size of spider habitats, necessary for (b). Informally, the rule to remove contours is:

Rule 4: (Erasure of a Contour) Let d_1 be a unitary diagram and let c be a contour in d_1. Let d_2 be a copy of d except that c has been erased. If two zones combine to form a single zone then (a) if at most one of these zones is shaded then the resulting zone is not shaded, otherwise the resulting zone is shaded, and (b) any spider whose habitat included at least one of these zones has a new habitat that includes the resulting zone. Then d_1 logically entails d_2.

Notice that this inference rule is not, in general, a logical equivalence. In our running example, we can remove A and C from both d_4 and d_7 and remove A and E from a copy of d_2 without losing the contradictory information. It takes 8 proof steps to transform the diagram in Fig. 6 to the diagram in Fig. 7: transform $(d_4 \vee d_1' \vee d_7) \wedge d_2$ first into $((d_4 \vee d_7) \wedge d_2) \vee (d_1' \wedge d_2)$, then remove contours (6 proof steps) and equalize the zone sets (1 proof step, using a rule that allows shaded zones to be added) to give the diagram $((d_8 \vee d_9) \wedge d_{10}) \vee (d_1' \wedge d_2)$. We have now used 16 proof steps in total. We must now finish addressing (b), in that two of the diagrams involved in the contradiction still have spiders with multiple zone habitats. The fewest proof steps needed to resolve this, without losing the contradictory information, use the erasure of a spider rule:

Rule 5: (Erasure of a Spider) Let d_1 be a unitary diagram and let s be a spider whose habitat, $\eta_{d_1}(s)$, does not contain any shaded zones. Let d_2 be a copy of d_1 except that s has been erased. Then d_1 logically entails d_2.

We can apply this rule three times to Fig. 7, followed by distributivity, to give $(d_{11} \wedge d_{10}) \vee (d_{12} \wedge d_{10}) \vee (d_1' \wedge d_2)$ in Fig. 8; so far, we have used a total of 20 proof steps. The so-called combining rule (omitted for space reasons) identifies that $d_{11} \wedge d_{10}$ and $d_{12} \wedge d_{10}$ are both inconsistent and replace them with the symbol $\perp$ (the 'false' spider diagram), thus adding a further two steps to our proof. Lastly, we can use two applications of the logic rule '$\perp \vee d$ entails d' to deduce $d_1' \wedge d_2$. This proof, which is the shortest we have been able to

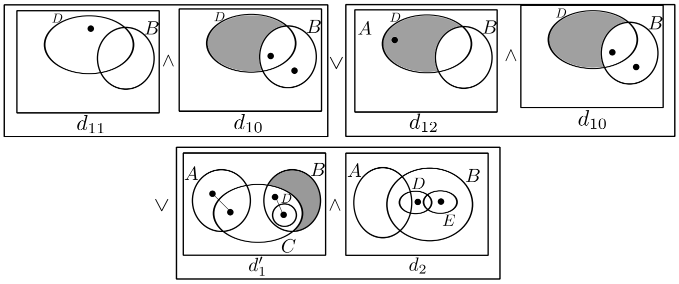

Fig. 8. Removing spiders and applying distributivity.

find, establishes that $d_1 \wedge d_2 \vdash d_1' \wedge d_2$, and has taken a total of 24 steps. Showing that $d_1' \wedge d_2 \vdash d_1 \wedge d_2$ is much more straightforward: simply use the erasure of a contour label rule to remove D from d_1' to give d_1. Thus, to establish logical equivalence we can write two proofs, one requiring 24 steps and the other requiring just a single step.

In conclusion, we argue that requiring 24 proof steps to make an obvious deduction is unwieldy, and does not best reflect the primary motivation for using diagrams which is to overcome communication barriers presented by symbolic logics and to make inference tasks easier for people to undertake. Thus, we propose a new set of inference rules that allow obvious proof steps to be made.

IV. Corresponding Regions

Considering the example in Fig. 2, we could see how to add D to d_1 to give d_1'. Intuitively, we could observe that D is a subset of B and, thus, draw D inside B in d_1'. However, at a formal level, we only have access to the sets of zones in the two diagrams and it is not immediately obvious that the zone in d_1 into which D is drawn, namely $(\{B, C\}, \{A\})$, represents a superset of D. We need some way, at the abstract syntax level, of identifying when regions represent the same sets, or when one represents a subset of another.

We define three syntactic correspondence relations, identifying when two regions represent the same set, a subset or a superset of each other. To achieve this, it is helpful to define various syntactic concepts. For instance, the zones that are not present in a unitary diagram, d, but which could be given the label set $L(d)$, are *missing*. Such zones represent the empty set in models for d. Moreover, shaded zones that do not form part of any spider's habitat in d also represent the empty set.

Definition 4: Let d be a unitary diagram. The **missing** zones of d are those in the set

$$MZ(d) = \{(in, out) \in \mathcal{Z} : in \cup out = L(d)\} - Z(d).$$

The **empty** zones of d are elements of the set

$$EZ(d) = MZ(d) \cup \{z \in ShZ(d) : \forall s \in S(d)\ z \notin \eta_d(s)\}.$$

Lemma 1: Let d be a unitary diagram and let $I = (U, \Psi)$ be a model for d. Then the empty zones represent the empty set, that is, if $z \in EZ(d)$ then $\Psi(z) = \emptyset$.

We use the concept of empty zones when defining inference rules: if we have two unitary diagrams taken in conjunction and a zone, z, is empty in one of them then we can use that information to determine, in part, how we apply inference rules

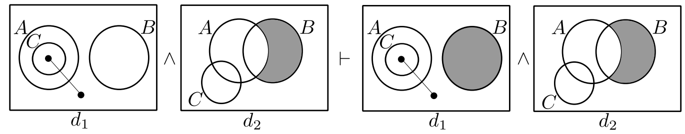

Fig. 9. Using empty zones to make deductions.

on the other diagram, for example. To illustrate, in Fig. 9, in d_2 the zone $(\{B\}, \{A, C\})$ is empty so we can add shading to this zone in d_1, as shown in d_1'.

The notion of corresponding regions was introduced in [19] for Euler diagrams, where a syntactic definition was provided that established when two regions represented the same set. Here, we give a definition of corresponding regions that is effective for unitary spider diagrams taken in conjunction: we prove that our definition captures when two regions, one from d_1 and the other from d_2, necessarily represent the same set in models for $d_1 \wedge d_2$. We also define the notion of a corresponding sub-region and a corresponding super-region, relating to subset and superset respectively. To illustrate, the two regions $r_1 = \{(\{A, D\}, \{B\}), (\{A\}, \{B, D\})\}$ and $r_2 = \{(\{A, C\}, \{B\}), (\{A\}, \{B, C\})\}$ both represent the same set and are corresponding; informally, they both represent the set $A - B$. In this example, we can be confident that r_1 and r_2 represent the same set in any interpretation:

$$\begin{aligned}
\Psi(r_1) &= \Psi(\{A, D\}, \{B\}) \cup \Psi(\{A\}, \{B, D\}) \\
&= \Psi(\{A, D, C\}, \{B\}) \cup \Psi(\{A, C\}, \{B, D\}) \cup \\
&\quad\ \Psi(\{A, D\}, \{B, C\}) \cup \Psi(\{A\}, \{B, D, C\}) \\
&= \Psi(\{A, C\}, \{B\}) \cup \Psi(\{A\}, \{B, C\}) \\
&= \Psi(r_2).
\end{aligned}$$

Given d_1 and d_2 as in Fig. 10, the region

$$r_3 = \{(\{A, D\}, \{B\}), (\{A\}, \{B, D\}), (\{B\}, \{A, D\})\}$$

also represents the same set as r_2 (and r_1) in any model for $d_1 \wedge d_2$, since the zone $(\{B\}, \{A, D\})$ is empty:

$$\begin{aligned}
\Psi(r_3) &= \Psi(\{A, D\}, \{B\}) \cup \Psi(\{A\}, \{B, D\}) \cup \Psi(\{B\}, \{A, D\}) \\
&= \Psi(\{A, D\}, \{B\}) \cup \Psi(\{A\}, \{B, D\}) \\
&= \Psi(r_2).
\end{aligned}$$

The region r_3 corresponds to r_2. In order to syntactically identify whether two regions, r and r' are corresponding we need to transform them, altering their zones by adding labels. The transformation is based on the observation that given any zone, (in, out), and a label, l, not used in the zone, we have

$$\Psi(in, out) = \Psi(in \cup \{l\}, out) \cup \Psi(in, out \cup \{l\}).$$

The zone (in, out) can, thus, be transformed into the two zones $(in \cup \{l\}, out)$ and $(in, out \cup \{l\})$. We use this insight to define the notion of an *expansion* of a region, which iteratively 'splits'

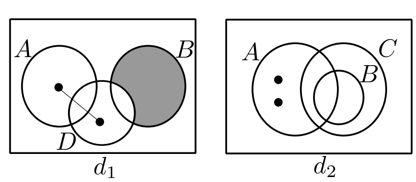

Fig. 10. Corresponding regions.

zones in this manner, given some set of labels. In what follows, we denote the set of labels used in a region, r, by $L(r)$, so

$$L(r) = \bigcup_{(in,out)\in r} (in \cup out).$$

If r is a region in a unitary diagram then $L(r) = L(d)$.

Definition 5: Let r be a region such that all of the zones in r partition $L(r)$. Let L' be a set of labels such that $L(r) \subseteq L'$. An **expansion** of r given L', denoted $exp(r, L')$, is the region defined as follows:

1) If $|L' - L(r)| = 1$ then
$$exp(r, L') = \{(in \cup (L' - L(r)), out) : (in, out) \in r\} \cup \{(in, out \cup (L' - L(r))) : (in, out) \in r\}.$$
2) If $|L' - L(r)| > 1$ then $exp(r, L') = exp(r', L')$ where $r' = exp(r, L(r) \cup \{\lambda\})$ for some label $\lambda \in L' - L(r)$.

For example, given $r = \{(\{B\}, \{A\})\}$ and $L' = \{A, B, C, D\}$, we have

$$\begin{aligned} exp(r, L') &= exp(exp(r, \{A, B, C\}), L') \\ &= (\{B, C, D\}, \{A\}), (\{B, C\}, \{A, D\}), \\ &\quad (\{B, D\}, \{A, C\}), (\{B\}, \{A, C, D\})\}. \end{aligned}$$

It is obvious that the order in which the labels are introduced during the expansion does not matter. Moreover, we do not change the represented set:

Lemma 2: Let r be a region such that all of the zones in r partition $L(r)$. Let L' be a set of labels such that $L(r) \subseteq L'$. In any interpretation, $I = (U, \Psi)$, $\Psi(r) = \Psi(exp(r, L'))$.

Definition 6: Let d_1 and d_2 be unitary diagrams. Let r_1 and r_2 be regions in $Z(d_1) \cup MZ(d_1)$ and $Z(d_2) \cup MZ(d_2)$ respectively. Then r_1 and r_2 are **corresponding**, denoted $r_1 \equiv_c r_2$, provided that

$$\begin{aligned} &exp(r_1, L) \cup exp(EZ(d_1), L) \cup exp(EZ(d_2), L) = \\ &\qquad exp(r_2, L) \cup exp(EZ(d_1), L) \cup exp(EZ(d_2), L). \end{aligned}$$

where $L = L(d_1) \cup L(d_2)$. Furthermore, r_1 is a **corresponding sub-region** of r_2, denoted $r_1 \subseteq_c r_2$, provided that

$$\begin{aligned} &exp(r_1, L) \cup exp(EZ(d_1), L) \cup exp(EZ(d_2), L) \subseteq \\ &\qquad exp(r_2, L) \cup exp(EZ(d_1), L) \cup exp(EZ(d_2), L). \end{aligned}$$

If $r_1 \subseteq_c r_2$ then r_2 is a **corresponding super-region** of r_1, denoted $r_2 \supseteq_c r_1$.

In Fig. 10, we have $r_4 \subseteq_c r_5$ where $r_4 = \{(\{A\}, \{B, D\})\}$ and $r_5 = \{(\{A\}, \{B, C\}), (\{A, C\}, \{B\}), (\{A, B, C\}, \emptyset)\}$. Intuitively, r_4 represents the set $A - (B \cup D)$ and r_5 represent A and we see that in any model, $I = (U, \Psi)$, for $d_1 \wedge d_2$ that $\Psi(r_4) \subseteq \Psi(r_5)$. The following theorem establishes that our syntactic correspondence relations respect the semantics as intended:

Theorem 1: Let d_1 and d_2 be unitary diagrams and let r_1 and r_2 be regions in $Z(d_1) \cup MZ(d_1)$ and $Z(d_2) \cup MZ(d_2)$ respectively. Let $I = (U, \Psi)$ be a model for $d_1 \wedge d_2$.

1) If $r_1 \equiv_c r_2$ then $\Psi(r_1) = \Psi(r_2)$.
2) If $r_1 \subseteq_c r_2$ then $\Psi(r_1) \subseteq \Psi(r_2)$.
3) If $r_1 \supseteq_c r_2$ then $\Psi(r_1) \supseteq \Psi(r_2)$.

V. Inference rules

We now present new inference rules for spider diagrams, extending the sound and complete set in [18] conservatively (so we still have completeness), further developed in [17], [21]. Proving that all of our new rules are sound is relatively straightforward; soundness proofs are omitted for space reasons.

Our new rules are designed to allow intuitive proof steps to be made and to substantially reduce proof length and are all *logical equivalences*. We focus on defining rules for two unitary diagrams taken in conjunction, say $d_1 \wedge d_2$, and demonstrate how to alter the information in d_1 without changing the informational content of the conjunction. We can alter the syntax of d_1 in five ways, impacting on the contours, regions, shading, spiders, and spider habitats, giving rise to five new rules; to derive our new inference rules, for each of these syntactic elements, we asked 'under what conditions can we alter this element in d_1, to yield $d_1' \wedge d_2$, without altering the information. We adopt the standard approach of defining the inference rules at the abstract syntax level and, as is normal, the intention is that they are applied at the concrete syntax level, for which tool support, described below, is important.

The new rules make similar alterations to the syntax of d_1 as some of the information weakening rules that apply to unitary diagrams in [18]. For example, a previously existing rule that applies to a unitary diagram allows zones to be added to a spider's habitat, losing information, whereas our new 'add zones to a spider's habitat' rule below applies to a compound diagram and preserves information.

Our first new rule allows us to copy a contour from one diagram into another diagram; we argue that this is a natural inference step. To illustrate, we can copy D from d_2 in the diagram $d_1 \wedge d_2$ in Fig. 2 to d_1, as shown in $d_1' \wedge d_2$. To define the copy contour rule, we start by observing that each zone in the diagram, d_1, is either (a) completely outside the copied contour, (b) completely inside the copied contour, or (c) split into two zones by the copied contour. The effect of copying the contour with label l on each zone is determined by the information available. In particular, for each non-empty zone, z, in d_1 that is *not* a corresponding sub-region of the empty zones in d_2:

1) z is completely inside the copied contour if it represents a subset of l (captured by a corresponding sub-region relation); in Def. 7 below these zones are in Z_i;
2) z is completely outside the copied contour if it represents a subset of $U - l$ (captured by a corresponding sub-region relation); these zones are in Z_o; and
3) z is split into two zones otherwise; these zones are in Z_s.

Zones which are empty in d_1 or are corresponding sub-regions of empty zones of d_2, can either be inside, outside or split. This is reflected in the definition of the copy contour rule, which extends the much more restricted copy contour inference rule given in [21] that only considered pairwise relationships between contours and used no notion of corresponding regions.

Definition 7: Let d_1 and d_2 be unitary diagrams and let l be in $L(d_2) - L(d_1)$. We define three subsets of $Z(d_1) - EZ(d_1)$, namely $Z_i(l, d_2)$, $Z_o(l, d_2)$, and $Z_s(l, d_2)$, according to the following rules. Let $z \in Z(d_1) - EZ(d_1)$ such that $\{z\} \not\subseteq_c EZ(d_2)$.

1) If $\{z\} \subseteq_c \{(in_2, out_2) \in Z(d_2) : l \in in_2\}$ then $z \in Z_i(l, d_2)$.
2) If $\{z\} \subseteq_c \{(in_2, out_2) \in Z(d_2) : l \in out_2\}$ then $z \in Z_o(l, d_2)$.
3) If $z \notin Z_i(l, d_2) \cup Z_o(l, d_2)$ then $z \in Z_s(l, d_2)$.

Rule 6 (Copy a Contour): Let d_1 and d_2 be unitary diagrams and let l_2 be in $L(d_2) - L(d_1)$. Let Z_{IN}, Z_{OUT} and Z_{SPLIT} be a 3-way partition of $Z(d_1)$ such that

1) $Z_i(l_2, d_2) \subseteq Z_{IN}$,
2) $Z_o(l_2, d_2) \subseteq Z_{OUT}$, and
3) $Z_s(l_2, d_2) \subseteq Z_{SPLIT}$.

Let d_1' be the diagram defined as follows:

1) the contour labels are $L(d_1') = L(d_1) \cup \{l_2\}$,
2) the zones are

$$Z(d_1') = \{(in \cup \{l_2\}, out) : (in, out) \in Z_{IN} \cup Z_{SPLIT}\} \cup \{(in, out \cup \{l_2\}) : (in, out) \in Z_{OUT} \cup Z_{SPLIT}\},$$

3) the shaded zones are

$$ShZ(d_1') = \{(in \cup \{l_2\}, out) : (in, out) \in (Z_{IN} \cup Z_{SPLIT}) \cap ShZ(d_1)\} \cup \{(in, out \cup \{l_2\}) : (in, out) \in (Z_{OUT} \cup Z_{SPLIT}) \cap ShZ(d_1)\},$$

4) the spiders are $S(d_1') = S(d_1)$, and
5) the habitat of each spider, $s' \in S(d_1')$, is

$$\eta_{d_1'}(s') = \{(in \cup \{l_2\}, out) : (in, out) \in (Z_{IN} \cup Z_{SPLIT}) \cap \eta_{d_1}(s)\} \cup \{(in, out \cup \{l_2\}) : (in, out) \in (Z_{OUT} \cup Z_{SPLIT}) \cap \eta_{d_1}(s)\}.$$

Then $d_1 \wedge d_2$ is logically equivalent to $d_1' \wedge d_2$.

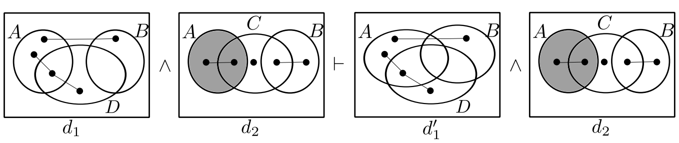

Fig. 11. The add a region inference rule.

When we have two unitary diagrams, d_1 and d_2, taken in conjunction there are circumstances under which we can enlarge the zone set (but not the shaded zone set) of one diagram without altering the semantics of the conjunction. This occurs when there is a set of zones, r, missing from d_1 that is a corresponding sub-region of the empty zones in d_2. Since d_2 implies that r represents $\emptyset$, we can remove this information from d_1 (by introducing the missing zones in r) without losing the information from $d_1 \wedge d_2$. This inference rule is illustrated in Fig. 11, where both d_1 and d_2 express that $A \cap B = \emptyset$. This information is removed from d_1 to give d_1', but it is still present in the diagram $d_1' \wedge d_2$.

Rule 7 (Add a Region): Let d_1 and d_2 be unitary diagrams. Let r_1 be a subset of $MZ(d_1)$ such that $r_1 \subseteq_c EZ(d_2)$. Let d_1' be the diagram identical to d_1 except that $Z(d_1') = Z(d_1) \cup r_1$. Then $d_1 \wedge d_2$ is logically equivalent to $d_1' \wedge d_2$.

For the copy shading inference rule, consider as an example $d_1 \wedge d_2$ in Fig. 12. Here, the region r_1 comprising all zones inside B in d_1 can be shaded, given the information in $d_1 \wedge d_2$. In particular, r_1 corresponds to the entirely shaded region r_2 comprising all zones inside B in d_2 and we can also 'match' the spiders in these regions: $s_{1,i}$ matches $s_{2,i}$ where $\eta_{d_1}(s_{1,i}) \supseteq_c \eta_{d_2}(s_{2,i})$. By match, we mean that there is a bijection, σ, between the relevant spiders (captured in Def. 8 below) that ensures the habitat of each spider, s, in r_1 is a corresponding super-region of the habitat of $\sigma(s)$. This matching ensures that whichever element is represented by $s_{2,i}$ it can also be represented by $s_{1,i}$, which is important to ensure the rule's soundness. The result of adding shading to B in d_1 can be seen in d_1'. If $s_{1,1}$ was not in d_1 then d_1' would assert $|B| = 1$ which cannot be deduced from $d_1 \wedge d_2$.

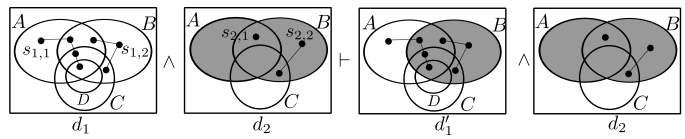

Fig. 12. The copy shading inference rule.

In order to define this rule, we introduce some notation to denote the set of spiders whose habitat includes zones of a region r in a unitary diagram d and the habitat outside of r corresponds to empty zones of d_2.

Definition 8: Let d_1 and d_2 be unitary diagrams and r be a region in d_1. We define

$$S(r, d_1, d_2) = \{s \in S(d_1) : \eta_{d_1}(s) \cap r \neq \emptyset \wedge \eta_{d_1}(s) - r \subseteq_c EZ(d_2)\}.$$

Rule 8 (Copy Shading): Let d_1 and d_2 be unitary diagrams with regions r_1 and r_2 respectively such that:

1) $r_1 \subseteq_c r_2$,
2) r_1 contains at least one non-shaded zone in d_1, that is $r_1 - ShZ(d_1) \neq \emptyset$,
3) r_2 is entirely shaded in d_2, that is, $r_2 \subseteq ShZ(d_2)$,
4) in d_1, each spider, s, whose habitat includes a zone of r_1, that is, $\eta_{d_1}(s) \cap r_1 \neq \emptyset$, is also in $S(r_1, d_1, d_2)$,
5) in d_2, all of the spiders whose habitat includes a zone of r_2, that is, $\eta_{d_2}(s) \cap r_2 \neq \emptyset$, is also in $S(r_2, d_2, d_1)$, and
6) there is a bijection, $\sigma \colon S(r_1, d_1, d_2) \to S(r_2, d_2, d_1)$, such that for each spider, s, $\eta_{d_1}(s) \supseteq_c \eta_{d_2}(\sigma(s))$.

Let d_1' be the diagram identical to d_1 except that $ShZ(d_1') = ShZ(d_1) \cup r_1$. Then $d_1 \wedge d_2$ is logically equivalent to $d_1' \wedge d_2$.

The next new rule allows us to copy a spider from one diagram to another. This is illustrated in Fig. 13, where we can copy a spider from d_2 into d_1. From d_2 we can see that there are at least three elements in A, one of which is also in D. The diagram d_1 tells us that there is an element in A, so we can copy a spider from d_2 into d_1. In this case, we copy the spider inside D in d_2, as shown in d_1'.

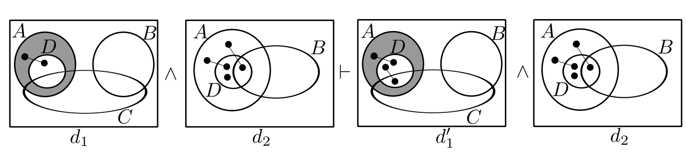

Fig. 13. The copy a spider inference rule.

Rule 9 (Copy a Spider): Let d_1 and d_2 be unitary diagrams with regions r_1 and r_2 respectively, and a region $r' \subseteq r_1$ in d_1 such that:

1) $r_1 \subseteq_c r_2$,
2) r' contains no shaded zones in d_1, that is, $r' \cap ShZ(d_1) = \emptyset$,
3) in d_1, each spider, s, whose habitat includes a zone of r_1, that is, $\eta_{d_1}(s) \cap r_1 \neq \emptyset$, is also in $S(r_1, d_1, d_2)$,
4) there exists an injective, but not surjective, function $\sigma\colon S(r_1, d_1, d_2) \to S(r_2, d_2, d_1)$ such that
   a) for each spider s, $\eta_{d_2}(\sigma(s)) \subseteq_c \eta_{d_1}(s)$, and
   b) there exists a spider, s_2, that is in $S(r_2, d_2, d_1)$ but is not mapped to by σ, such that $\eta_{d_2}(s_2) \subseteq_c r'$.

Let s_1 be a fresh spider. Let d_1' be the diagram identical to d_1 except that $S(d_1') = S(d_1) \cup \{s_1\}$ and the habitat of each spider, s', in $S(d_1')$ is

$$\eta_{d_1'}(s') = \begin{cases} \eta_{d_1}(s') & \text{if } s' \in S(d_1) \\ r' & \text{otherwise.} \end{cases}$$

Then $d_1 \wedge d_2$ is logically equivalent to $d_1' \wedge d_2$.

For the inference rule pertaining to spider's habitats, we begin by observing that if we add zones to a spider's habitat in d_1 that correspond to empty zones in d_2 then we have not changed the informational content of $d_1 \wedge d_2$. This is illustrated in Fig. 14, where the zone $(\{A\}, \{D\})$ represents the empty set, asserted by d_2, so adding spider feet to the corresponding region in d_1, as shown in d_1', does not change the semantics.

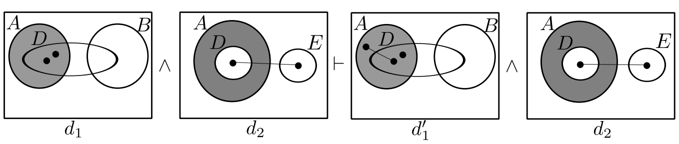

Fig. 14. The add zones to a spider's habitat.

Rule 10 (Add Zones to a Spider's Habitat): Let d_1 and d_2 be unitary diagrams with a spider, s_1, in $S(d_1)$ such that there exists a region r' where

1) $r' \cap \eta_{d_1}(s_1) = \emptyset$, and
2) $exp(r', L(d_1) \cup L(d_2)) \subseteq exp(EZ(d_2), L(d_1) \cup L(d_2))$.

Let d_1' be the diagram identical to d_1 except that the habitat of each spider, s', in $S(d_1')$ is

$$\eta_{d_1'}(s') = \begin{cases} \eta_{d_1}(s') & \text{if } s' \in S(d_1) - \{s_1\} \\ \eta_{d_1}(s') \cup r' & \text{otherwise.} \end{cases}$$

Then $d_1 \wedge d_2$ is logically equivalent to $d_1' \wedge d_2$.

Focusing now on tool support, our theorem prover Speedith [21] supports reasoning with spider diagrams. Speedith consists of five main components: an abstract representation of spider diagrams; a reasoning kernel which provides Speedith with its proof infrastructure that contains a collection of spider diagram inference rules, handles their application, and manages proofs; an external communication system which includes input and output mechanisms for spider diagrams and sentential formulae to enable external verification through existing general-purpose theorem provers; an 'iCircles' visualization algorithm for automatically drawing unitary spider diagrams when inference rules are applied, extending [20]; and a graphical user interface, which includes compound spider diagram visualization, user interaction with spider diagram elements, graphical user interface panels for interactive proof management and interactive application of inference rules. Speedith supports both forward-style and backward-style proofs.

The user interface allows the entry of spider diagrams and the construction of spider diagram proofs interactively. The user-led application of inference rules is performed at the concrete syntax level, whereby the user selects the rule to apply and the appropriate part of the diagram that is to be altered by the rule. For example, to apply 'add zones to a spider's habitat', the user would select the rule (called 'add feet' in Speedith for brevity), select the spider and select the zones which are to be added via the concrete diagram. Speedith then applies the rule at the abstract syntax level and automatically draws the resulting diagram.

VI. Impact on Proof Length

To evaluate the impact of our inference rules we can take a number of approaches. These include conducting user studies to ascertain whether our 'new' proofs are more understandable than the 'old' proofs, asking people whether the proofs are seemingly more natural, or ascertaining the impact on proof length. Designing appropriate empirical studies is difficult, due to the need to train participants in spider diagrams and logical reasoning. Whether a proof appears natural is subjective to the reader and any study attempting to ascertain the relative 'naturalness' of the proofs would need to use experts who are not available in large numbers. Finally, producing shorter proofs does not always result in more natural, or 'better', proofs. However, impact on proof length is not at all subjective and can be computed. Moreover, as we have conservatively extended the set of inference rules, the shortest proof with the new set of rules will never be longer than with the old set: proofs never get longer. Therefore, we have chosen to establish the impact on proof length as a method of evaluating our new inference rules.

We conjecture that savings in proof length are likely to be more significant when the proof task (i.e., premise and conclusion diagram) contain more syntax, particularly spiders with large habitats. Thus, by choosing simple examples to demonstrate savings in proof length, any bias in the evaluation is likely to favour the original inference rule set. It is, though, unfortunate that no standard corpus of examples exists to evaluate proof length in a truly unbiased way. Having access to a standard corpus of examples would also provide insight concerning the frequency with which the new rules reduce proof lengths.

We already demonstrated in section III that the inference rules for spider diagrams given in [18] result in overly long proofs, even in cases where the diagram to be proved (the theorem) obviously follows from the diagram assumed to be true (the assumption). We now briefly look at the impact on proof length for the examples given in the paper to illustrate the benefits of our new inference rules. For each example, the lengths of the shortest proofs that we have been able to find in order to show $d_1 \wedge d_2 \vdash d_1' \wedge d_2$ and $d_1' \wedge d_2 \vdash d_1 \wedge d_2$ are given in Table I. Using our new rules, establishing the equivalence takes just one step (and needs only one proof). It is possible to use the inference rules in [18] that are logical equivalences to establish the semantic equivalence of $d_1 \wedge d_2$ and $d_1' \wedge d_2$ in all of these examples, thus yielding just a single proof, but the number of proof steps required to do so is considerably larger than the total number required to write two proofs. In any case, we can see that even in these simple examples the number of steps required in proofs using the rules of [18] is at least ten times as many as we now need. It is not hard to construct example proof tasks that are only slightly more complex than the simple examples we have given where the saving in proof length is substantially greater.

Task	$d_1 \wedge d_2 \vdash d_1' \wedge d_2$	$d_1' \wedge d_2 \vdash d_1 \wedge d_2$	Saving
Fig. 2	24	1	24
Fig. 11	3	11	13
Fig. 12	10	1	10
Fig. 13	9	1	9
Fig. 14	1	11	11

TABLE I. SAVINGS IN PROOF LENGTH

VII. CONCLUSION

We have demonstrated that existing inference rules for spider diagrams can result in unnatural, overly long proofs. A particular consequence of unnatural proofs can be an obfuscation of proof strategy. To achieve a fundamental goal of the diagrams community, of making inference more accessible, it is not only important to provide diagrams that are effective modes of communication but that are also equipped with inference rules that result in accessible proofs. To this end, we presented five new rules for spider diagrams, all of which are sound, and allow seemingly natural proof steps to be made. In order to define these new rules, we had to provide an understanding of when syntactically different regions represent sets that are equal or in a subset relationship. In short, the novel contributions of this paper are the provision of the syntactically defined correspondence relations on regions, and new inference rules that allow substantially shorter proofs to be written.

In fact, the notion of corresponding regions was *central* to our approach of defining inference rules. The correspondence relations, whilst defined at the abstract syntax level, allowed us to readily capture information that is visually obvious in concrete (drawn) diagrams and which people can see to be true (such as curve containment reflecting a subset/superset relation between the represented sets). We argue that developing diagrammatic inference rules that allow the use of information that is visually displayed in diagrams is paramount to being able to make obvious proof steps. This is related to the notion of *observation* devised by Swoboda and Allwein, who considered when information conveyed in symbolic logic sentence was semantically entailed by Euler/Venn diagrams [22]. We believe that utilising such an approach to defining inference rules (i.e., using visually displayed information) takes us a step closer to realizing the full potential of diagrammatic logics. We hope to see similar approaches adopted for other notations.

Looking to the future, we plan to define more new inference rules that operate on diagrams taken in disjunction, as well as enlarging the set of logical operators to include $\neg$, $\Rightarrow$ and $\Leftrightarrow$, which were not considered in the sound and complete spider diagram logic in [18]. Secondly, it may well be beneficial to attempt more rigorous evaluations using, perhaps, cognitive dimensions or other approaches as outlined in section VI, to establish the accessibility of the new inference rules in contrast with the proofs produced by the original rule set. Lastly, a major goal is to devise strategies that guide inference rule choice in order to produce proofs that are accessible to people.

ACKNOWLEDGMENT

This work was supported by EPSRC Advanced Research Fellowship GR/R76783 (Jamnik), EPSRC Doctoral Training Grant and Computer Laboratory Premium Research Studentship (Urbas).

REFERENCES

[1] S. Oviatt, A. Cohen, A. Miller, A. Mann, "The impact of interface affordances on human ideation, problem solving, and inferential reasoning," *ACM Trans. on Computer-Human Interaction*, 19(3) 2012.

[2] Y. Sato, K. Mineshima, R. Takemura, "The efficacy of Euler and Venn diagrams in deductive reasoning: Empirical findings," in *Diagrams*. Springer, 2010, pp. 6–22.

[3] A. Rector et al., "OWL pizzas: Practical experience of teaching OWL-DL: Common errors and common patterns," in *Engineering Knowledge in the Age of the Semantic Web*. Springer, 2004, pp. 63–81.

[4] S.-J. Shin, *The Logical Status of Diagrams*. CUP, 1994.

[5] C. Peirce., *Collected Papers*. Harvard University Press, 1933, vol. 4.

[6] E. Hammer, *Logic and Visual Information*. CSLI Publications, 1995.

[7] S.-J. Shin, *The Iconic Logic of Peirce's Graphs*. Bradford Book, 2002.

[8] F. Dau, "Constants and functions in Peirce's existential graphs," in *Conceptual Structures*, 2007, pp. 429–442.

[9] M. Erwig, "Abstract syntax and semantics of visual languages," *Journal of Visual Languages and Computing*, vol. 9, p. 461483, 1998.

[10] J. Howse, F. Molina, S.-J. Shin, J. Taylor, "Type-syntax and token-syntax in diagrammatic systems," in *2nd Int. Conference on Formal Ontology in Information Systems*. ACM Press, 2001, pp. 174–185.

[11] P. S. di Luzio, "Patching up a logic of Venn diagrams," in *6th CSLI Workshop on Logic, Language and Computation*. CSLI, 2000.

[12] S. Kent, "Constraint diagrams: Visualizing invariants in object oriented models," in *Proc. OOPSLA97*. ACM, 1997, pp. 327–341.

[13] A. Fish, J. Flower, J. Howse, "The semantics of augmented constraint diagrams," *J. of Visual Languages and Computing*, 16:541–573, 2005.

[14] K. Mineshima, M. Okada, R. Takemura, "A diagrammatic inference system with Euler circles," *J. of Logic, Language and Information*, 21(3):365–391, 2012.

[15] N. Swoboda, G. Allwein, "Using DAG transformations to verify Euler/Venn homogeneous and Euler/Venn FOL heterogeneous rules of inference," *J. on Software and System Modeling*, 3(2):136–149, 2004.

[16] J. Gil, J. Howse, S. Kent, "Formalising spider diagrams," in *IEEE Symposium on Visual Languages*. IEEE, 1999, pp. 130–137.

[17] G. Stapleton, J. Masthoff, J. Flower, A. Fish, J. Southern, "Automated theorem proving in Euler diagrams systems," *J. of Automated Reasoning*, 39:431–470, 2007.

[18] J. Howse, G. Stapleton, J. Taylor., "Spider diagrams," *LMS Journal of Computation and Mathematics*, 8:145–194, 2005.

[19] J. Howse, G. Stapleton, J. Flower, J. Taylor, "Corresponding regions in Euler diagrams," in *Diagrams*. Springer, 2002, pp. 76–90.

[20] G. Stapleton, J. Flower, P. Rodgers, J. Howse, , "Automatically Drawing Euler Diagrams with Circles," *Journal of Visual Languages and Computing*, 23(3):164-193, 2012.

[21] M. Urbas, M. Jamnik, G. Stapleton, J. Flower, "Speedith: A diagrammatic reasoner for spider diagrams," in *Diagrams*. Springer, 2012, pp. 163–177.

[22] N. Swoboda and G. Allwein, "Modeling heterogeneous systems," in *Diagrams*. Springer, 2002, pp. 131–145.