Spider Diagrams of Order

Aidan Delaney* and Gem Stapleton†
Visual Modelling Group,
University of Brighton,
Brighton, United Kingdom BN2 4GJ

Abstract

Spider diagrams are a visual logic capable of makeing statements about relationships between sets and their cardinalities. Various meta-level results for spider diagrams have been established, including their soundness, completeness and expressiveness. Recent work has established various relationships between spider diagrams and regular languages, which highlighted various classes of languages that spider diagrams could not define. In particular, this work illustrated the inability of spider diagrams to place an order on certain letters in words. To overcome this limitation, in this paper we introduce *spider diagrams of order*, incorporating an order relation and present a formalisation of the syntax and semantics. Subsequently, we define the language of such a diagram and establish that the class of such languages includes that of the piecewise testable languages.

1 Introduction

Diagrams are often used to convey information and aid communication in a variety of areas, including software engineering, mathematics and every day life. Recently, the perception of the role of diagrams in logic has been overturned, with advances showing that diagrams can be given precise syntax and semantics with, subsequently, formal reasoning systems being built on them; for example [5, 6, 8, 14, 17]. As a result, the utility of diagrams is seen as broader, and some considerable effort is now being placed on exploring visual languages in the context of logic.

One such logic is the language of spider diagrams (see, for example [8, 16]). With regard to applications of spider diagrams, they have been used to assist with the task of identifying component failures in safety critical hardware designs [1] and (implicitly) in a variety of other areas, such as [2, 9, 18]. It has been established that spider diagrams have the expressiveness of monadic first order logic with equality by providing translations between these two languages that preserves semantics [16]. In this paper, we consider the expressiveness of spider diagrams in comparison with regular languages, building on results presented in [3] where a limitation is highlighted. In particular, regular languages often constrain the orders that letters may appear in a word of that language, but spider diagrams are unable to do this. To overcome this expressiveness limitation, we extend the spider diagram language to include facilities for ordering elements. Extending spider diagrams to include an order relation will allow them to be

*a.j.delaney@brighton.ac.uk
†g.e.stapleton@brighton.ac.uk

used in more application areas. For example, one may choose to use spider diagrams over finite state machines when defining languages; see [3] for further discussions on this relationship.

One of our goals in increasing the expressiveness of spider diagrams is to provide a specification tool for trace semantics and synchronisation expressions. Trace semantics and synchronisation expressions have existing formal language characterisation in [4] and [13] respectively. Our first step is to extend spider diagrams and examine the ramifications with respect to formal language theory. A longer term goal of this body of work is to examine whether diagrammatic logics make a more succinct 'programming language' for problems with solutions in regular language space. The results in [3] on the descriptional complexity of spider diagrams and finite state automata support this succinctness conjecture.

In more general terms, the study of the relationships between logics and formal languages has led to a range of important results related to decidability, the circuit synthesis problem and has provided new perspectives to the construction of non-terminating programs, discussed in [19]. In this vein, it may well prove fruitful to further our understanding of the relationship between spider diagrams and regular languages. For example, fragments of the spider diagrams language might correspond to classes of regular languages that are not naturally characterised in any other way. Consequently, this may provide a deeper understanding of the relationships between classes of regular languages themselves.

In section 2, we briefly overview the existing spider diagram notation. Section 3 introduces various concepts from formal language theory that are necessary for this paper and discusses the relationship between spider diagrams and regular languages. *Spider diagrams of order* are introduced and formalised in section 4. Finally, in section 5, we prove that a fragment of the language of spider diagrams of order gives rise to the well known class of piecewise testable languages also called level 1 of the Straubing-Thérin hierarchy.

2 Spider Diagrams

This section will provide a brief overview of the spider diagram syntax presented in [8]. In figure 1, the spider diagram d_1 contains two labelled *contours*, A and B. Contours are simple closed curves. The diagram also contains three minimal regions, called *zones*. There is one zone inside A, another inside B and the other zone is outside both A and B. Each zone can be described by a two-way partition of the contour label set. The zone inside the contour A can be described as inside A but outside B. A *region* is a set of zones. The two zones outside B contain a *spider*; spiders are trees whose vertices, called *feet*, are placed in zones (in this case, the spider has two feet). Spider diagrams can also contain *shading* placed in zones, as in d_2 (which contains two spiders and four zones of which one is shaded). The horizontal line connecting d_1 and d_2 in figure 1 denotes disjunction between diagrams; thus, the figure contains $d_1 \vee d_2$. Similarly, juxtaposition of two diagrams d_1 and d_2 with no connecting line denotes their conjunction, $d_1 \wedge d_2$.

Our attention now turns to the semantics. Spider diagrams make statements about sets (represented by contours) and their cardinalities (by using spiders and shading). In figure 1, d_1 expresses that A and B are disjoint, because there are no points interior to both of the contours. Spiders assert the existence of elements, so d_1 specifies that there is (at least) one elements in either A or the universe outside both A and B. The spiders

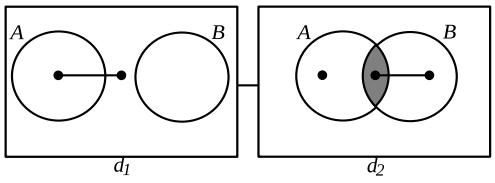


Figure 1: Two spider diagrams.

in d_2 assert that there are at least two elements, one of which is in $A - B$ and the other is in $A \cap B$ or $B - A$. Shading is used to place upper bounds on set cardinality: in the set represented by a shaded region, all of the elements are represented by spiders. For example, d_2 expresses that the set $A \cap B$ contains at most one element.

3 The Straubing-Thérin Hierarchy

In our previous work [3] we have studied the relationship between spider diagrams and star-free regular languages. We established that sets of words from a subset of star-free regular languages can be thought of as corresponding to *models* for spider diagrams (a model will be formally defined later). The Straubing-Thérin hierarchy serves as a fine-grained tool for describing various subsets of star-free regular languages. This hierarchy is infinite but it is an open question as to whether the hierarchy is proper above so-called 'level 2'.

Level 0 of the Straubing-Thérin hierarchy is the set of languages $\{\Sigma^*, \emptyset\}$ where Σ is an alphabet. Level $1/2$ is the well known *shuffle ideal* set, which is the polynomial closure of Level 0. Level 1 is defined as the boolean closure of $1/2$. This hierarchy has been extended by Pin to consider varieties of languages [10]: in general for any positive integer $n > 0$

level $n + \frac{1}{2}$ is the polynomial closure of level n, and

level $n + 1$ is the boolean closure of level $n + \frac{1}{2}$.

The boolean closure of a set of languages $\mathscr{L} \subseteq \Sigma^*$ is $B(\mathscr{L})$ which is formed by taking the union, intersection and complement of languages. The polynomial closure of a set of languages $\mathscr{L} \subseteq \Sigma^*$, $Pol(\mathscr{L})$, is the finite union of languages of the form $L_0 a_1 L_1 \ldots a_n L_n$ where $L_0, L_1, \ldots, L_n \in \mathscr{L}$ and $a_1, \ldots, a_n \in \Sigma$.

For our purposes, it is sufficient to state that a formal language is a set of words defined over an alphabet, Σ. The boolean operations $\cup, \cap$ and $\subset$ and the unary complement $\neg$ operator maintain their well understood semantics over sets of words. The additional boolean operation called the shuffle product, denoted $\sqcup$, will allow us to utilise characterisations of the Straubing-Thérin hierarchy more suitable for our definitions and theorems.

Definition 3.1. *The **shuffle product** of two languages L_1, L_2 denoted $L_1 \sqcup L_2$ informally takes all words from L_1 and intersperses letters from each word in L_2. More formally, the words in $L_1 \sqcup L_2$ are precisely those of the form $w_0 w_1 \ldots w_n$ where there exists a partition $I \cup J$ of $\{1, 2, \ldots, n\}$ with*

1. $I = \{p_1, p_2, \ldots, p_i\}, p_1 < p_2 < \ldots < p_i$,

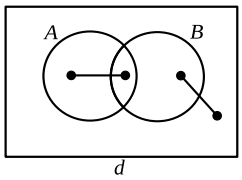


Figure 2: A spider diagram.

2. $J = \{q_1, q_2, \ldots, q_j\}, q_1 < q_2 < \ldots < q_j$ *(thus* $i + j = n$*), and*

3. $w_{p_1} \ldots w_{p_i} \in L_1$ *and* $w_{q_1} \ldots w_{q_j} \in L_2$.

As an example, the shuffle product of the sets of words $A = \{xy\}$ and $B = \{yz, y\}$ is the set of words $\{xyyz, xyzy, yxzy, yzxy, yxyz, xyy, yxy\}$. Languages of catenation level $1/2$, that is the shuffle ideals, are of the form $k \sqcup \Sigma^*$ where k is a finite set of words.

In this paper, we are concerned with the relationship between spider diagrams and regular languages. We have already established various relationships between spider diagrams and catenation hierarchy levels $1/2$ and $3/2$. In particular, we proved that spider diagrams give rise to languages that are closed under permutation of words and, thus, cannot constrain a language to contain words, w, in which certain letters must occur before others.

As an example, the diagram d in figure 2 represents a star-free language of words over the four-letter alphabet $\Sigma = \{AB, \overline{A}B, A\overline{B}, \overline{AB}\}$; here the alphabet has been obtained by considering the contours in the diagram, i.e. A and B, and the four possible combinations of being inside or outside the contours, with $\overline{A}$ denoting 'being outside A'. The diagram asserts, by way of the spiders, that there is an element in $A - B$ or $A \cap B$ (because of the spider placed inside A) and an element not in A (by the placement of the other spider). In terms of regular languages, we can take this diagram as asserting that all words contain letters corresponding to these possibilities given rise to by the spiders. The spider inside A, therefore, tells us that the words must contain either the letter $A\overline{B}$ or AB. The other spider tells us that words must contain either $\overline{A}B$ or the letter $\overline{AB}$. We further refine the notation to include square brackets, $[A\overline{B}]$ to aid readability of words. Words in the language of the diagram, denoted $\mathscr{L}(d)$, are precisely those that contain at least one letter from the first spider and one letter from the other spider, in either order. Using the characterisation of shuffle-ideal languages given above we may construct a set of words, k, such that $\mathscr{L}(d) = k \sqcup \Sigma^*$. Such a k is given by

$$\begin{aligned} k \;=\; & \{[A\overline{B}][\overline{A}B], [\overline{A}B][A\overline{B}], [A\overline{B}][\overline{AB}], \\ & [\overline{AB}][A\overline{B}], [AB][\overline{A}B], [\overline{A}B][AB], [AB][\overline{AB}], [\overline{AB}][AB]\} \end{aligned}$$

and we observe that k is closed under permutation. The language $\mathscr{L}(d) = k \sqcup \Sigma^*$ maintains the closure under permutation property of the set k.

Spider diagrams are a monadic first order logic with equality (MOFLe) [16]. We are interested in the relationship between logics and subsets of regular languages. The main body of literature discussing this relationship [7, 10, 11, 12, 19] assumes the existence of an order relation $<$ adjunct to the standard monadic first order operators of $\neg, \vee, \wedge, \iff$, the quantifiers $\exists$ and $\forall$ and predicates of the form $P_a(x)$ which

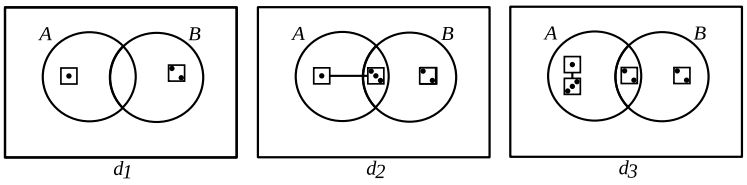


Figure 3: Generalising spider diagrams.

states that the letter a is at positive position x in a word w. Intuitively, if we do not have an order relation, $<$, as in MOFLe then any language corresponding to a formula will be closed under permutation. In other words, languages of the form, for example, $\Sigma^* A \Sigma^* B$ (A comes before B in every word) do not correspond to languages arising from formulae in MOFLe. Consequently spider diagrams do not contain facilities for ordering elements and it is this main body of literature, referenced above, that provides a motivation for generalising spider diagrams to include facilities for ordering elements.

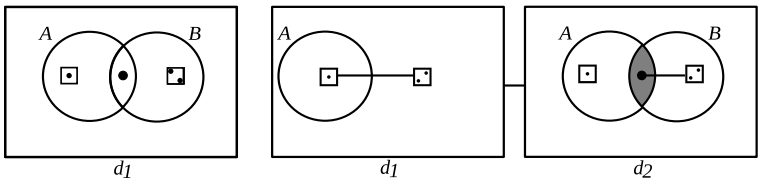
4 Generalising Spider Diagrams to Include an Order Relation

As just stated, spider diagrams are limited in their expressive power and cannot enforce any kind of order on elements represented by the spiders. Here, we generalise the spider diagram syntax and extend their semantics appropriately to overcome this expressiveness limitation. For example, the *spider diagram of order* labelled d_1 in figure 3 contains spiders whose feet are labelled with dots. The number of dots is used to place an order on the elements represented by the spiders (alternative syntax would simply label the feet with natural numbers as opposed to dots; such a change of syntax would have no impact on the work that follows and is merely a different means of visualisation). Thus, this diagram d_1 is interpreted as saying that there is an element, x, in $A - B$ and another element, y, in $B - A$ such that $x < y$. The semantics of the spider diagram of order labelled d_2 are a little more subtle. This diagram expresses that there is an element, x, in A and an element, y, in $B - A$ such that, if $x \in A - B$ then $x < y$, otherwise $y < x$. Here we see that the labels can be used to place an order on the elements represented by the spiders in the context of which sets those elements are located.

A further modification is to allow spiders to have more than one foot placed in each zone, an idea first raised in [15] but not in the context of ordering elements. For example, in figure 3, d_3 expresses that there is an element, x, in $A-B$, another element, y, in $B - A$ and a third element, z, in $A \cap B$. The element x satisfies either $x < y$ and $x < z$ or $y < x$ and $z < x$. The elements y and z are both represented by spiders that have the same label (i.e. 2) and we interpret this as expressing y and z can be in either order: $y < z$ or $z < y$.

Sometimes we might want to express an order on certain elements (as in the examples we have just seen) but not on other elements; in the previous example, we did not specify an order on y and z. Suppose we want express the following:

1. there exist three distinct elements, x_1, x_2 and x_3,



(a) A unitary diagram. (b) A compound diagram.

Figure 4: Spider Diagrams of Order.

2. x_1 is in the set $A - B$,
3. x_2 is in the set $B - A$,
4. x_3 is in the set $A \cap B$, and
5. $x_1 < x_2$.

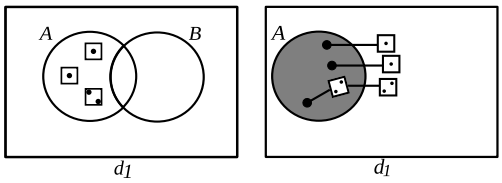
To allow this statement to be expresses succinctly by spider diagrams, we allow the use of non-labelled feet as in the original notation. A diagram of order making this statement can be seen in figure 4(a), where the spider placed in $A \cap B$ has no label, thus indicating we do not mind whether $x_1 < x_3$ or $x_3 < x_1$, for example.

With regard to shading, it is interpreted in the same way as the original notation: in a shaded region, all of the elements are represented by spiders. As with the spider diagram language in [8], we allow diagrams to be taken in disjunction and conjunction, forming *compound diagrams*. In addition, the compound diagram $\neg d$ is also allowed. We have just provided various examples of unitary spider diagrams of order. The *compound diagram* in figure 4(b) represents the disjunction of two unitary diagrams $d_1 \vee d_2$. The horizontal line joining d_1 to d_2 denotes disjunction; the juxtaposition of the unitary diagrams would represent conjunction. For the remainder of this section, we provide a formalisation of the syntax and semantics of spider diagrams of order.

4.1 Syntax

Each spider diagram of order consists of contours (closed, plane, labelled curves), spiders which are trees whose nodes (called *feet*) are a character such as $\bullet, \boxdot, \boxdot, \boxdot, \ldots$ and shading. For example, the diagram d_1 in figure 4(a) contains two contours labelled A and B and three spiders, one with a foot labelled $\boxdot$, another with a foot labelled $\bullet$, and the other with a foot labelled $\boxdot$. In general, any given spider may contain both *ordered feet* (those of the form $\boxdot$) and *unordered feet* (those of the form $\bullet$).

Formally, the syntax is defined at an abstract level, extending that given in [16]. The contour labels in spider diagrams are selected from a finite set $\mathcal{L}$. A **zone** is defined to be a pair, (in, out), of finite disjoint subsets of $\mathcal{L}$. The set in contains the labels of the contours that the zone is inside whereas out contains the labels of the contours that the zone is outside. The set of all zones is denoted $\mathcal{Z}$. To describe the spiders in a diagram, it is sufficient to say how many spiders there are with any given foot arrangement. For example, in figure 5(a), there are two spiders inside A with a single foot labelled 1 and another spider also inside A but with single foot labelled 2. Thus, our abstract definition of a spider diagram will specify the labels used, the zones, identify which



(a) Simple example. (b) More complex example.

Figure 5: Illustrating the syntax.

zones are shaded and use a set of spider identifiers to describe the spiders. We have adopted this approach because it directly extends the abstract syntax presented in [16].

To begin our formalisation, we start by defining spider feet, which may be ordered, denoted with an integer index, or unordered, denoted with a $\bullet$ character and subsequently we define spiders. When we formalise the semantics, it is useful to have access to the region in which a spider is placed, called its *habitat*.

Definition 4.1. *A **spider foot** is an element of the set $(\mathbb{Z}^+ \cup \{\bullet\}) \times \mathcal{Z}$ and the set of all feet is denoted $\mathcal{F}$. A **spider**, s, is a set of feet together with a number: $s \in \mathbb{Z}^+ \times (\mathbb{P}\mathcal{F} - \{\emptyset\})$ and the set of all spiders is denoted $\mathcal{S}$. The **habitat** of a spider $s = (n, p)$ is the region $habitat(s) = \{z : \exists k\,(k, z) \in p\}$.*

Spiders are numbered because unitary diagrams can contain many spiders with the same foot set; essentially, we view a unitary diagram as containing a bag of spiders.

Definition 4.2. *A **unitary spider diagram of order** is a quadruple $d = \langle L, Z, ShZ, SI\rangle$ where*

$L = L(d) \subseteq \mathcal{L}$ *is a set of contour labels,*

$Z = Z(d) \subseteq \{(a, L - a) : a \subseteq L\}$ *is a set of zones,*

$ShZ = ShZ(d) \subseteq Z(d)$ *is a set of shaded zones,*

$SI = SI(d) \subsetneq \mathcal{S}$ *is a finite set of **spider identifiers** such that for all $(n_1, p_1), (n_2, p_2) \in SI(d)$,*

$$(p_1 = p_2 \implies n_1 = n_2) \wedge habitat(n_1, p_1) \subseteq Z(d).$$

The symbol $\perp$ is also a unitary spider diagram. We define

$$L(\perp) = Z(\perp) = ShZ(\perp) = SI(\perp) = \emptyset.$$

*If d_1 and d_2 are spider diagrams then $(d_1 \wedge d_2), (d_1 \vee d_2)$ and $\neg d_1$ are **compound spider diagrams of order**.*

We observe that the set of spider identifiers is just a set of spiders, but with at most one spider with any given foot arrangement present. If, for example, the spider identifier set contains the pair $(2, \{(\bullet, z)\})$ then this would tell us that d contains two spiders in zone z whose feet are of the form $\bullet$. As a more concrete example, the spider diagram of order in figure 5(b) has abstract syntax:

1. labels $L(d) = \{A\}$

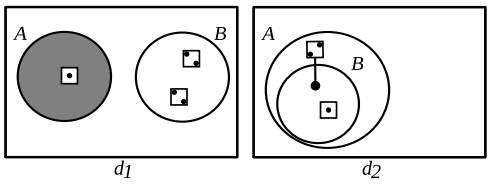


Figure 6: Illustrating the semantics.

2. zones $Z(d) = \{z_1 = (\{A\}, \emptyset), z_2 = (\emptyset, \{A\})\}$
3. shaded zones $ShZ(d) = \{z_1\}$
4. spider identifiers $SI(d) = \{(2, \{(\bullet, z_1), (1, z_2)\}), (1, \{(\bullet, z_1), (2, z_1), (2, z_2)\})\}$.

It is useful to identity the set of spiders present in a diagram, which is implicit in the spider identifier set and to be able to arbitrarily select feet of spiders. For example, when defining the semantics, each spider, s, represents and element and the feet place a disjunction of constraints on that element; thus to identify whether and *interpretation* (see below) is a model for a unitary diagram there needs to be a choice of foot for which s satisfies the constraint imposed.

Definition 4.3. *The set of **spiders** in unitary diagram d is defined to be*

$$S(d) = \{(i, p) : \exists (n, p) \in SI(d)\, 1 \leq i \leq n\}.$$

*Let $FootSelect\colon S(d) \to \mathcal{F}$ be a function. If, for all $(n, p) \in S(d)$, $FootSelect(s) \in p$ then $FootSelect$ is called a **foot selection function** for d.*

It is further useful to identify which zones could be present in a unitary diagram, given the label set, but are not present; semantically, *missing* zones provide information.

Definition 4.4. *Given a unitary diagram, d, a zone (a, b) is said to be **missing** if it is in the set $\{(a, L - a) : a \subseteq L\} - Z(d)$ with the set of such zones denoted $MZ(d)$. If d has no missing zones then d is in **Venn form** [8].*

4.2 Semantics

Regions in spider diagram represent sets and the spatial arrangement of the contours places constraints on the relationships between those sets. For example, in figure 6, the diagram d_1 contains two contours, A and B, that do not overlap, indicating that the sets they represent are disjoint. Spiders make existential statements about elements. In particular, each spider denotes the existence of a particular element in the set represented by the spider's habitat, with distinct spiders denoting distinct elements. From d_1 we can deduce that there is an element, x, in A and two elements, y and z, in B. The numbers on the spiders feet provide information on the ordering of elements in the universe. With regard to d_1, we deduce that $x < y$ and $x < z$. The shading tells us that all of the elements are represented by spiders, so d_1 asserts that A contains a single element, namely x. The diagram d_2 expresses $B \subseteq A$ (because B is placed inside A) and there is an element, x, in A and another element, y, in B such that if $y \in A - B$

then $x < y$. The foot labels have particular significance as they are used to place a restriction on the order of elements.

To formalise the semantics, we need to first interpret the contour labels as sets and interpret the order relation $<$. Thus, we extend the definition of an interpretation given in [8] to spider diagrams of order.

Definition 4.5. *An **interpretation** is a triple $(U, \Psi, <)$ where U is a universal set and $\Psi\colon \mathcal{L} \to \mathbb{P}U$ is a function that assigns a subset of U to each contour label and $<$ is an irreflexive, antisymmetric and transitive relation on U. The function Ψ can be extended to interpret zones and sets of regions as follows:*

1. *each zone, $(a, b) \in \mathcal{Z}$, represents the set $\bigcap_{l \in a} \Psi(l) \cap \bigcap_{l \in b} \overline{\Psi(l)}$ and*
2. *each region, $r \in \mathbb{P}\mathcal{Z}$, represents the set which is the union of the sets represented by r's constituent zones.*

For brevity, we will continue to write $\Psi\colon \mathcal{L} \to \mathbb{P}U$ but assume that the domain of Ψ includes the zones and regions also. Given an interpretation we wish to know whether it is a model for a diagram; in other words, when the information provided by the interpretation agrees with the intended meaning of the diagram. Informally, an interpretation is a *model* for unitary diagram d ($\neq\bot$) whenever

1. all of the zones which are missing represent the empty set,
2. all of the regions represent sets whose cardinality is at least the number of spiders placed entirely within that region and
3. all of the entirely shaded regions represent sets whose cardinality is at most the number of spiders with a foot in that region.
4. the elements represented by the spiders obey the ordering imposed on them by the spiders' feet.

We now make this notion precise.

Definition 4.6. *Let $I = (U, \Psi, <)$ be an interpretation and let d ($\neq\bot$) be a unitary spider diagram of order. Then I is a **model** for d if and only if the following conditions hold.*

1. ***The missing zones condition*** $\bigcup_{z \in MZ(d)} \Psi(z) = \emptyset$.
2. ***The function extension condition*** *There exists an extension of Ψ to spiders, $\Psi : \mathcal{L} \cup S(d) \to \mathbb{P}U$ which ensures the following further conditions hold.*
   (a) ***The habitats condition*** *All spiders represent elements (strictly, singleton sets) in the sets represented by their habitats:*
   $$\forall s \in S(d)\ \Psi(s) \subseteq \Psi(habitat(s)) \wedge |\Psi(s)| = 1.$$
   (b) ***The distinct spiders condition*** *Distinct spiders denote distinct elements:*
   $$\forall s_1, s_2 \in S(d) : \Psi(s_1) = \Psi(s_2) \implies s_1 = s_2.$$

(c) ***The shading condition*** *Shaded regions represent sets containing elements denoted by spiders:*

$$\Psi(ShZ(d)) \subseteq \bigcup_{s \in S(d)} \Psi(s).$$

(d) ***The order condition*** *The ordering information provided by the spiders agrees with that provided by $<$: there exists a foot selection function, $FootSelect\colon S(d) \to \mathcal{F}$, for d such that*

- *for all $s \in S(d)$, $FootSelect(s) = (n, z)$ implies $\Psi(s) \subseteq \Psi(z)$*
- *for all $s_1, s_2 \in S(d)$ with $FootSelect(s_1) = (n_1, z_1)$ and $FootSelect(s_2) = (n_2, z_2)$, if $n_1 \neq n_2$ then either*
  - *i. $n_1 < n_2$ and $x < y$ where $\Psi(s_1) = \{x\}$ and $\Psi(s_2) = \{y\}$ or*
  - *ii. $n_2 < n_1$ and $y < x$ or*
  - *iii. $n_1 = \bullet$ or*
  - *iv. $n_2 = \bullet$.*

*If $\Psi\colon \mathcal{L} \cup S(d) \to \mathbb{P}U$ ensures that the above conditions are satisfied then Ψ is a **valid** extension to spiders for d. A foot selection function, $FootSelect\colon S(d) \to \mathcal{F}$, that ensures the above conditions are satisfied is also called **valid**. If $d = \bot$ then the interpretation is not a model for d.*

For compound diagrams, the definition of a model extends in the obvious inductive way.

Theorem 4.1. *Let d be a unitary spider diagram. Then d has a model.*

Proof. (Sketch) Take $U = S(d)$ and any foot selection function $FootSelect\colon S(d) \to \mathcal{F}$ for d. Using $FootSelect$, define $\Psi\colon \mathcal{L} \to \mathbb{P}U$ ensuring that $\Psi(l)$ contains precisely the set of spiders whose selected foot lies in a zone contained by l, whenever l is in $L(d)$. Further, use $FootSelect$ to define $<$ in the obvious way: for spiders s_1 and s_2, $s_1 < s_2$ if and only if both of the feet selected for s_1 and s_2 have integer labels, n_1 and n_2 respectively, and $n_1 < n_2$. □

5 Regular Languages and Spider Diagrams of Order

The finite models of spider diagrams of order give rise to words. We take our alphabet Σ to be a finite set of zones, $(a, b) \in \mathcal{Z}$, such that $a \cup b = \mathcal{L}$ and further assume that all unitary diagrams have a label set $\mathcal{L}$ (all unitary diagrams are semantically equivalent to another unitary diagram with label set $\mathcal{L}$, therefore expressiveness is not affected). Each spider s in a unitary diagram d is said to *give rise to* a letter in word w by selecting a foot of s using some fixed $FootSelect$ function for d. An unordered foot $(\bullet, z_i) = FootSelect(s)$ specifies that the letter z_i appears in w. An ordered foot of the form $(\boxdot, z_i) = FootSelect(s_1)$ specifies that z_i appears at a position in w before the letter z_j of an ordered foot of the form $(\boxdot, z_j) = FootSelect(s_2)$, and so-forth.

For example, the diagram d_1 in figure 4(a) has a model $U = \{x, y, z\}$ where A represents the set $\{x, y\}$ and B represents the set $\{y, z\}$ and $< = \{(x, z)\}$. Taking $Z(d) = \Sigma$, but using the notational convention established in section 3 rather than the formal syntax, then the following words

$$[A\overline{B}][AB][\overline{A}B], [A\overline{B}][\overline{A}B][AB]$$

arise from this model. By contrast, $\overline{A}AB\overline{AB}A$ does not arise from this model for d_1 as there must be an occurrence of the letter $\overline{A}B$ at some index in w after the letter $A\overline{B}$ as $(x, z) \in <$. In other words if $A\overline{B}$ were at index 1 (the first letter) then $\overline{A}B$ must occur at index 2 or 3. Conversely, given a word, we want to establish whether it arises from a model for some given diagram. We say such words *conform* to the diagram.

Definition 5.1. *Let $w = w_1w_2...w_n$ be a word in $\Sigma = Z(d)$ and d ($\neq\perp$) be a unitary diagram. The bag (or multiset) of letters of which w consists is denoted bag(w). The word w **conforms**, to d if and only if there exists a function, $f : S(d) \to (\mathbb{Z}^+ \cup \{\bullet\}) \times bag(w)$ satisfying*

1. *$f(s)$ is a foot of s,*
2. *f is injective with regard to $bag(w)$: for all $(n_1, w_i), (n_2, w_i) \in im(f)$, $n_1 = n_2$,*
3. *f is bijective with regard to $bag(w)$ when the image is restricted to the maximal sub-bag of w whose elements are shaded zones in d, denoted $shadebag(w)$: for all $w_i \in shadebag(w)$ there exists $s \in S(d)$ such that $f(s) = (n, w_i)$ for some n.*
4. *for all $s_1, s_2 \in S(d)$, if $f(s_1) = (n_1, w_i)$ and $f(s_2) = (n_2, w_j)$ are distinct feet and $n_1 < n_2$ then the $i < j$*

For $d = \perp$, no words in Σ^* conform to d.

So, w conforms to unitary diagram $d(\neq\perp)$ provided, for each spider, s, in d,

1. each spider in d gives rise to a letter in w by way of selecting a foot,
2. a low ranked foot gives rise to a letter appearing at a lower index of w than a letter corresponding to a higher ranked foot,
3. for each shaded zone, z, the number of occurrences of z in w is precisely the number of spiders whose selected foot is z.

Definition 5.2. *The **language of a unitary spider diagram of order**, denoted $\mathscr{L}(d)$, is the set of words which conform to the diagram d.*

Definition 5.3. *When considering the language of a compound spider diagram of order:*

- $\mathscr{L}(d_1 \wedge d_2) = \mathscr{L}(d_1) \cap \mathscr{L}(d_2)$,
- $\mathscr{L}(d_1 \vee d_2) = \mathscr{L}(d_1) \cup \mathscr{L}(d_2)$,
- $\mathscr{L}(\neg d_1) = \Sigma^* - \mathscr{L}(d_1)$.

We can describe the language of d_1 in figure 4(a) in the form $k \sqcup \Sigma^*$ where k is the finite set of words generated only by spiders i.e.

$$k = \{[A\overline{B}][AB][\overline{A}B], [AB][A\overline{B}][\overline{A}B], [A\overline{B}][\overline{A}B][AB]\}$$

As d_1 contains no shading or missing zones there are no letters prevented from being in words of $\mathscr{L}(d_1)$, thus $\mathscr{L}(d_1) = k \sqcup \Sigma^*$.

Theorem 5.1. *The set of languages for spider diagrams of order whose unitary parts contain no shaded zones is the boolean closure of shuffle-ideal languages, that is level 1 of the Straubing-Thérin hierarchy.*

Proof. (Sketch) Let l be a level 1 language. Then l is a boolean combination of shuffle ideals. We show that each shuffle ideal is the language of a spider diagram. The result that l can be represented then follows by the inductive construction of l and the spider diagram language. An arbitrary shuffle-ideal language can be seen as the finite disjunction:

$$k \sqcup \Sigma^* = (k_1 \sqcup \Sigma^*) \cup (k_2 \sqcup \Sigma^*) \cup \ldots \cup (k_n \sqcup \Sigma^*)$$

where $k_1, k_2, \ldots, k_n$ is a partition of k where each $|k_i| = 1$ and $\Sigma \subseteq \mathcal{Z}$. We may then draw a spider diagram of order for each k_i with zone set Σ and one single-foot spider for each letter in k_i placed in the appropriate zone and having the appropriate rank. The language of the disjunction of these diagrams is $\mathscr{L}(k \sqcup \Sigma^*)$. Conversely, we must show each such spider diagram has a level 1 language, which is left to the reader. □

6 Conclusion

The main contributions of this paper are two-fold. First, we introduced spider diagrams of order, increasing the expressiveness of the spider diagram language. We then defined the language of a spider diagram of order and established the set of such languages includes all of level 1 of the Straubing-Thérin hierarchy. This builds upon the relationships previously established in [3]. The exact relationship between spider diagrams of order and the Straubing-Thérin hierarchy remains to be determined. We conjecture that there are languages in the class $3/2$ that are not the language of any spider diagram of order. In the future we wish to endow spider diagrams with the full power of monadic first order logic with equivalence of order. We further plan to establish whether spider diagrams are a natural specification technique for use in trace analysis.

Acknowledgement: Gem Stapleton is supported by a Leverhulme Trust Early Career Fellowship and by UK EPSRC grant EP/E011160/1 for the Visualization with Euler Diagrams project.

References

[1] R. Clark. Failure mode modular de-composition using spider diagrams. In *Proc. of Euler Diagrams 2004*, volume 134 of *Electronic Notes in Theoretical Computer Science*, pages 19–31, 2005.

[2] R. DeChiara, U. Erra, and V. Scarano. A system for virtual directories using Euler diagrams. In *Proc. of Euler Diagrams 04*, volume 134 of *Electronic Notes in Theoretical Computer Science*, pages 33–53, 2005.

[3] Aidan Delaney and Gem Stapleton. On the descriptional complexity of a diagrammatic notation (to appear). In *Visual Languages and Computing*, 2007.

[4] Volker Diekert and Yves Métivier. *Partial Communication and Traces*, pages 457–531. Springer-Verlag New York, Inc., 1997.

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

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

[7] Pierre-Cyrille Héam. On shuffle ideals. *Theoretical Informatics and Applications*, 36:359–384, 2002.

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

[9] J. Lovdahl. *Towards a Visual Editing Environment for the Languages of the Semantic Web*. PhD thesis, Linkoping University, 2002.

[10] Jean-Eric Pin. *Syntactic semigroups*, pages 679–746. Springer-Verlag New York, Inc., 1997.

[11] Jean-Eric Pin and Pascal Weil. Polynomial closure and unambiguous product. *Theoretical Computer Science*, 30:1–39, 1997.

[12] Alexander Rabinovich. Star free expressions over the reals. *Theor. Comput. Sci.*, 233(1–2):233–245, 2000.

[13] Kai Salomaa and Sheng Yu. Synchronization expressions and lanugages. *J. of Universal Computer Science*, 5(9), 1999.

[14] S.-J. Shin. *The Logical Status of Diagrams*. Cambridge University Press, 1994.

[15] G. Stapleton and J. Howse. Enhancing the expressiveness of spider diagram systems. In *Proc. of Distributed Multimedia Systems, Intl. Workshop on Visual Languages and Computings*, pages 129–138. Knowledge Systems Institute, 2006.

[16] G. Stapleton, S. Thompson, J. Howse, and J. Taylor. The expressiveness of spider diagrams. *J. of Logic and Computation*, 14(6):857–880, 2004.

[17] N. Swoboda and G. Allwein. Heterogeneous reasoning with Euler/Venn diagrams containing named constants and FOL. In *Proc. of Euler Diagrams 2004*, volume 134 of *ENTCS*. Elsevier Science, 2005.

[18] J. Thièvre, M. Viaud, and A. Verroust-Blondet. Using euler diagrams in traditional library environments. In *Euler Diagrams 2004*, volume 134 of *ENTCS*, pages 189–202. ENTCS, 2005.

[19] Wolfgang Thomas. *Languages, automata, and logic*, pages 389–455. Springer-Verlag New York, Inc., 1997.