Logique & Analyse **251** (2020), 231-260

MULTIPLE READABILITY IN PRINCIPLE AND PRACTICE: EXISTENTIAL GRAPHS AND COMPLEX SYMBOLS

Dirk Schlimm & David Waszek

Abstract

Since Sun-Joo Shin's groundbreaking study (2002), Peirce's *existential graphs* have attracted much attention as a way of writing logic that seems profoundly different from our usual logical calculi. In particular, Shin argued that existential graphs enjoy a distinctive property that marks them out as "diagrammatic": they are "multiply readable," in the sense that there are several different, equally legitimate ways to translate one and the same graph into a standard logical language. Stenning (2000) and Bellucci and Pietarinen (2016) have retorted that similar phenomena of multiple readability can arise for sentential notations as well. Focusing on the simplest kinds of existential graphs, called alpha graphs (AGs), this paper argues that multiple readability does point to important features of AGs, but that both Shin and her critics have misdiagnosed its source.

As a preliminary, and because the existing literature often glosses over such issues, we show that despite their non-linearity, AGs are uniquely parsable and allow for inductive definitions. Extending earlier discussions, we then show that that in principle, *all* propositional calculi are multiply readable, just like AGs: contrary to what has been suggested in the literature, multiple readability is linked neither to non-linearity nor to AGs' dearth of connectives. However, we argue that in practice, AGs are more *amenable* to multiple readability than our usual notations, because the patterns that one needs to recognize to multiply translate an AG form what we call *complex symbols*, whose structural properties make it easy to perceive and process them as units. Nevertheless, we show that such complex symbols, though largely absent from our usual notations, are not inherently diagrammatic and can be found in seemingly sentential languages. Hence, while ultimately vindicating Shin's idea of multiple readability, our analysis traces it to a different source and thus severs its link with diagrammaticity.

Keywords: Logic, Notations, Existential Graphs, Diagrams, Multiple Readability

Introduction

It is well known that different notations can be used to represent the same subject matter. For example, Leibniz and Newton developed different notations for analysis, and propositional and first-order logic can be represented by strings of symbols or by trees, but also by more graphical notations, such

doi: 10.2143/LEA.251.0.3288640 © 2020 by Peeters Publishers. All rights reserved.

as Frege's *Begriffsschrift* or Peirce's *existential graphs*, which Sun-Joo Shin's study (2002) brought to a wider audience. The question of how exactly these notational systems differ from each other has led to various attempts at classifying them. In particular, the distinction between "sentential" (or linear) and "diagrammatic" notations has been the focus of much attention.[1] Using Peirce's existential graphs as paradigmatic example, Shin (2002, 2011, 2015) has argued that "multiple readability" is a characteristic feature of diagrammatic representations. Roughly, her idea is that a typical existential graph admits several, equally legitimate translations (or "readings") into our standard logical notation: her "Multiple-Readings Algorithm," she writes, "allows us to translate one and the same graph into more than one sentence,"[2] while "a symbolic system is very careful to prevent multiple readings of a formula."[3] In the debate that ensued, Shin's thesis has been criticized on the grounds that similar phenomena of multiple readability can also arise for linear notations (Stenning 2000, Bellucci and Pietarinen 2016).

We believe that several distinct issues have been conflated in this debate, and that, as a result, the specificity of existential graphs has been misdiagnosed. Our goal is to disentangle these issues, and to show that, while Shin's "multiple readability" does indeed point to interesting features of existential graphs, these are independent of the properties usually invoked to explain it, such as non-linearity or "diagrammaticity." In our account, what lies behind Shin's remark is that AGs allow for the definition of what we shall call *complex symbols*, namely patterns of basic symbols with structural properties permitting their treatment as a unit. But complex symbols are not inherently "diagrammatic," in the sense that they can also be defined within notations usually seen as sentential or linear.

Let us begin by briefly introducing existential graphs.[4] For the sake of simplicity, we shall concentrate on the simplest kind of graphs, "alpha graphs" (henceforth AGs), which can be translated into what we now call *classical propositional logic*. AGs are composed of letters – corresponding to our propositional variables – and of closed, non-intersecting curves, called "cuts"

[1] See for instance Shimojima (1999) and Stenning (2000) for reviews.

[2] Shin (2002, 76).

[3] Shin (2002, 79).

[4] Peirce's writings on existential graphs, which were hard to access for a long time, are currently being edited and published in full (Peirce 2020–2021). (Among the smattering of manuscripts that were available before, good starting points are Peirce 1931–1958, §4.394–417 and Peirce 1902 reprinted as Peirce 1931–1958, §4.372–392 – as well as the manuscripts presented in Peirce 1976, vol. III/1, 405–446.) Roberts (1973) provides a thorough historical presentation; for short introductions from the perspective of modern logic, see Hammer (1996) or Shin (2002, chap. 3). For a discussion of Peirce's own goals in devising existential graphs, see Bellucci and Pietarinen (2016).

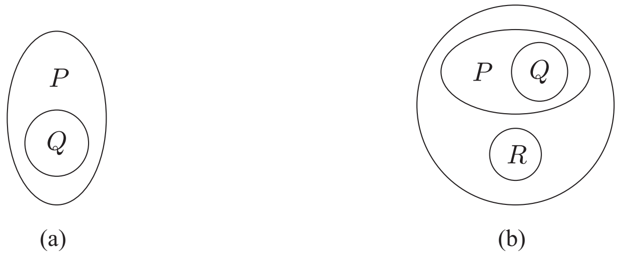


Figure 1. Two examples of alpha graphs (AGs).

(see Fig. 1 for examples); they are two-dimensional: letters can be placed anywhere on the page (within a specified area called "sheet of assertion" by Peirce) as long as they do not overlap cuts. AGs can be straightforwardly translated into our familiar propositional calculus by proceeding systematically from the outside in, translating cuts (curves) as negations and juxtapositions as conjunctions: for Fig. 1(a) and Fig. 1(b) this yields

$$\neg(P \wedge \neg Q) \qquad \text{and} \qquad \neg(\neg(P \wedge \neg Q) \wedge \neg R).$$

Let us call this the *basic* translation method.

As it turns out, the basic translation method is not well-defined, because it allows for multiple translations of one and the same AG into our standard notation for propositional logic. This is due to the following two features of AGs. First, the order in which the letters of an AG are inscribed on the sheet of assertion is irrelevant (AGs are not linear). Thus, in addition to the above translations, the AGs in Fig. 1(a) and Fig. 1(b) could equally well be translated as

$$\neg(\neg Q \wedge P) \qquad \text{and} \qquad \neg(\neg R \wedge (\neg Q \wedge P)),$$

respectively. Second, because any number of subgraphs can be juxtaposed simultaneously (i.e., the arity of conjunction in AGs is variable), they can be arranged in multiple ways in a target language that has only a *binary* connective for conjunction. For example, the AG '*PQR*' can be translated, among others, as $(P \wedge Q) \wedge R$ or $P \wedge (Q \wedge R)$ or as formulas that result from these by switching the conjuncts. These two peculiar features, *non-linearity* and the *variable arity of juxtaposition*, make it challenging to treat AGs with the standard tools of formal logic. For this reason, as we shall see, they have tended to be glossed over in previous efforts to describe AGs as formal expressions constructed by formation rules and susceptible of a unique parsing. Section 1 revisits the issue and argues that, with suitable care, one can in fact define a unique parsing of AGs.

However, Shin's notion of *multiple readability*, which we analyze in Section 2, is different. When translating AGs into our standard propositional calculus, one quickly recognizes that certain patterns of cuts correspond to formulas that are shorter than, but logically equivalent to, those obtained by the basic translation method.[5] For example, the AGs in Fig. 1(a) and Fig. 1(b) can also be translated as

$$P \to Q \qquad \text{and} \qquad (P \to Q) \to R.$$

The possibility of admitting additional translations like these on top of the basic one is, in a nutshell, what Shin calls "multiple readability." She ascribes this phenomenon to the non-linearity of AGs: in contrast, she writes that "linear representations cannot afford more than one reading for a given formula, since that could cause ambiguity" (Shin 2015, 57), referring to the fact that sentences of our usual propositional calculi are ascribed a unique *parsing* (a property that, confusingly, is often called "unique readability"). We argue that non-linearity cannot be the cause of multiple readability: not only can AGs be given a unique parsing, as shown in our first section, but insofar as they can also be given alternative parsings, so can sentences. Moreover, following Stenning (2000) and Bellucci and Pietarinen (2016), we show that multiple readings analogous to Shin's can be defined for sentential languages as well. We then argue that the alternate explanations these earlier works give of the phenomenon are not the full story either: Stenning (2000, 143) sees it as a "feature of translation from connectiveless to connectiveful calculi rather than of 2-D notation" and Bellucci and Pietarinen (2016, 230) claim that "in order to generate multiple readings the target-language must have *a richer logical vocabulary* than the source-language," but our systematic analysis shows that multiple readings can be defined for essentially any language, even when these conditions are not met. Our first conclusion is thus negative: multiple readings are extremely widespread and related neither to non-linearity nor to the various other explanations of it that have been proposed.

Nevertheless, we believe that Shin's notion of multiple readability does point to interesting phenomena, which we discuss in Section 3. While any notation can be given multiple readings, some do seem to be better suited to it than others. To explain this, we note that multiple readings, in general, rely on certain "translation patterns" present in the source formulas. We claim that it is in the specific properties of these patterns that notations differ. Building on earlier work by Schlimm (2018), we identify two such properties, namely *contiguity* and *context-freedom*, and call patterns that have

[5] Following Shin, we are restricting ourselves to classical logic, here and throughout the paper.

them *complex symbols*. We go on to tentatively suggest a psychological explanation, based on the literature on perception, for why complex symbols in this sense are valuable: they can be perceived and processed as units. This is the case, in particular, of the patterns underlying the multiple readings of AGs, but not those of our usual sentential languages. Our analysis thus supports Shin's intuition that AGs are more amenable to multiple readability than our standard propositional calculus.

1. Non-linearity and unique parsing of AGs

The most obvious distinctive feature of AGs is that they are non-linear. This raises an immediate challenge for a formal treatment of AGs: our usual logical notions (such as the type-token distinction, formation rules, and parsing trees) often seem to presuppose linear notations.[6] In this section, we clarify how these notions apply to AGs. In particular, we show how to define a unique parsing tree for every AG – a slightly troublesome matter that is frequently glossed over (e.g., in Roberts 1973, Hammer 1996, Shin 2002). We then build on this discussion to show how AGs' non-linearity gives them a first kind of "multiple readability."

1.1. *The type-token distinction for AGs*

It is a crucial feature of our usual formal systems that they presuppose *a type-token distinction*: their users must be able to recognize whether two individual symbols – and thus also two strings of symbols – are "the same," that is, are two *tokens* of the same *type*. For example, we usually regard the formulas

$$\Large A \wedge B \qquad \normalsize\text{and} \qquad A \wedge B.$$

as two tokens of the same type, because we recognize the symbols and their order, and we have learned that the size of the letters plays no role in logic. But, because we have also learned that the order of the symbols makes a difference, we notice that the following are not tokens of the same formula type:

$$A \wedge B \qquad \text{and} \qquad B \wedge A.$$

[6] This only applies to standard treatments in the logical literature; Peirce himself, who introduced the type-token distinction, did not consider it to be limited to strings of symbols.

The type-token distinction is essential for the very idea of a formal system (understood here as consisting of inscriptions that can be manipulated without regard of their meanings)[7], because it allows one to claim that different people, working with different physical inscriptions at different times and places, are doing the *same* thing – working with the *same* formula (that is, with tokens of the same type), following the *same* rules (again at the level of types).

If we want to treat AGs formally, then, we also need a type-token distinction, which includes a way of specifying which figures are tokens of the same type. For Figures 1(a) and 1(b) the case is straightforward, because they contain different sets of letters, so they are tokens of different types. But, let us look at the AGs of Figure 2. At first sight, they are clearly different: in 2(a), the letters are aligned vertically; in 2(b), Q is to the right of P and in 2(c), to its left; while in 2(d), the letters are not properly aligned at all. Nevertheless, the only consistent way of drawing a type-token distinction here is to regard them as tokens of the same type: indeed, because the letters of an AG can be placed anywhere save on cuts, one can pass from one figure to the other through a continuous transformation made of well-formed AGs, along which one would be unable to draw a clear line separating two different types. To consider them as tokens of different types, we would need to add restrictions on the position of letters, for instance by requiring them to be aligned on successive lines, so as to forbid graphs like 2(d); this, however, goes against the usual definition of AGs and would thus amount to a different system. So, in general, we shall consider two AG tokens as being of the same type if they can be transformed into one another by moving letters (and cuts) around, without crossing cuts.

Another route to the same result would be, like Hammer (1996, 133), to start by defining formation rules for AGs, then say that two graph-tokens are of the same type if they can be constructed in the same way (i.e., by applying the same rules in the same order). There is a subtlety there, however: in this formulation, the modal language (*can be constructed*) is crucial. Indeed, as we shall see presently, the rules stated by most authors do not guarantee that any AG can be constructed in only one way, so that two tokens with different construction histories can still be of the same type as long as they *could have been* produced by the same rules. But since the whole question of whether different construction histories can produce graphs of the same type presupposes an understanding of the type-token distinction for graphs, we feel it is more logical to introduce it before turning to formation rules.

[7] This is what MacFarlane (2000, 32–36), in his study of the formality of logic, calls "syntactic" formality.

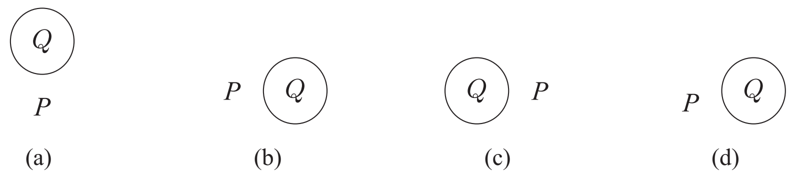


Figure 2. Four tokens of the same AG.

1.2. *Formation rules and uniqueness of parsing trees*

To continue our exploration of the system of AGs as a formal system, recall that formal languages are usually set up by stipulating a list of formation rules, such that any well-formed expression of the language can be constructed in only one way. This property is sometimes called "unique readability," but to avoid ambiguity, we shall refer to it as *unique parsability* instead. Our goal here is to show how AGs can be seen as uniquely parsable in this sense. Because of the already-noted peculiarities of AGs (non-linearity and the variable arity of juxtaposition), this will require us to alter the formation rules for AGs that are most commonly used in the literature.

Before we turn to AGs, some clarifications are needed about what it is, exactly, that is usually said to be "unique" in unique parsability and why it matters. Take a simple propositional calculus with connectives $\wedge$ and $\neg$. The following (in which Greek letters stand for arbitrary expressions) would be standard formation rules for well-formed formulas:

1. A propositional variable is a well-formed formula (wff), as are $\top$ and $\bot$.
2. If α is a wff, then so is $\neg\alpha$.
3. If α and β are wffs, then so is $(\alpha \wedge \beta)$.

Note that the abstract process of *constructing* a formula according to such rules is unrelated to the material process of *writing it down*, which need not (and usually does not) follow the same order. Now, the formula

$$\neg(\neg A \wedge \neg B) \tag{1}$$

(where Latin letters are propositional variables) can be constructed by applying rules 1 and 2 to get $\neg A$, then rules 1 and 2 again to get $\neg B$, then rule 3 to combine the previous formulas into $(\neg A \wedge \neg B)$, and then finally rule 2 to add the outer negation symbol. However, this sequence of applications of rules is not the only one that yields this particular formula; in other words, this construction sequence is not unique. One could also construct

$\neg B$ before $\neg A$, or start by applying rule 1 twice to get A and B, use rule 2 only then to get $\neg A$ and $\neg B$, and continue from there. Thus, different sequences of formation rules *can* yield the same formula. These sequences, however, only differ by their ordering: they are all made up of the same formation rules applied to the same subformulas. This defines, for any well-formed formula, a certain unique *abstract structure*, which we usually represent as a *parsing tree*.[8] For instance, the parsing tree of the formula (1) is represented in Fig. 3.

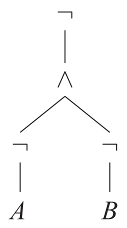


Figure 3. The parsing tree of the sentential formula $\neg(\neg A \wedge \neg B)$.

The great advantage of having unique abstract structures (usually represented by parsing trees) is that they allow for *inductive* definitions of properties of formulas. A good example of an inductive definition is that of the semantics of propositional logic: one first gives a semantics to propositional variables, and for each formation rule, one then defines how the semantics of the resulting formula depends on that of its constituents (usually in terms of truth tables). At first sight, this definition makes the semantics of a formula dependent, not just on the formula itself, but also on the formation rules used to construct it. This is where unique parsability comes in: the uniqueness of the abstract structure of a formula guarantees that inductive definitions yield properties of formulas that are intrinsic to them (i.e., belong to the formula independently of a specific construction history). Without unique parsability, inductive definitions would be much more cumbersome to work with, as one would have to survey all possible constructions of every formula and prove that the definition leads to the same result for each construction.

Now, with this understanding of unique parsability in place, let us examine whether we can define unique abstract structures (analogous to those of propositional formulas that we usually represent with parsing trees) for

[8] It should be kept in mind that, rigorously speaking, the abstract structure of a formula is not identical with its graphical representation in the form of a tree, but could be defined independently (for instance using set theory). For ease of expression, however, we shall sometimes speak of the parsing tree as if it was the abstract structure itself and not merely a representation of it.

AGs as well. To do this, we first need to lay out inductive formation rules for AGs. (The non-linearity of AGs is not an intrinsic obstacle to this: as remarked above, the process of constructing a formula according to formation rules is unrelated to that of writing the formula down; while the latter may – but need not! – follow a linear left-to-right order, the former is not based on any such order, so is not limited to linear languages.) Let us start from the following formation rules, which are usual in the literature:[9]

1. An empty space is an AG and any letter is an AG.
2. The juxtaposition of two or more non-empty AGs is an AG.
3. The cut (i.e., the enclosing in a closed curve) of an AG is an AG.

Notice that we already relied on these rules, albeit implicitly, to define the basic translation method inductively: we explained how to give translations for juxtapositions and for cuts. (The empty graph, not yet mentioned, is translated as $\top$; an empty cut is thus translated as $\bot$.) As remarked above, however, this translation method is not well-defined. The reason for this is that parsing trees are *not* unique in the case of AGs, because of two difficulties: (1) AGs are non-linear and (2) they can contain juxtapositions of an arbitrary number of subgraphs at the same time.

First, consider Figure 2 again: as Figures 2(b) and 2(c) are tokens of the same AG, both trees shown in Figure 4 could be taken as its parsing tree. Incidentally, these trees lead to different formulas of propositional logic, namely

$$P \wedge \neg Q \qquad \text{and} \qquad \neg Q \wedge P.$$

In general, since AGs are non-linear, there is no prescribed order among juxtaposed sub-graphs, and each choice of order leads to a different tree.

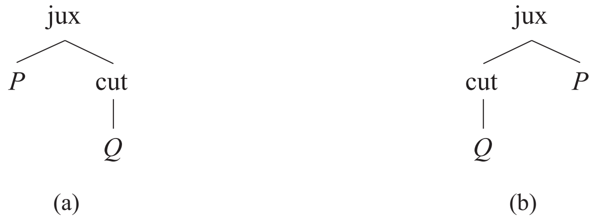


Figure 4. Parsing trees for the AG of Fig. 2.

[9] See for instance Shin (2002, 38), with the difference that we are restricting juxtaposition to *non-empty* graphs (following her approach on p. 65 – see also note 13 pp. 186-187), which is required for unique parsability. Similar rules are offered by Hammer (1996, 31) and (implicitly, as he describes the conventions stated by Peirce without isolating formation rules in particular) by Roberts (1973).

The second difficulty arises because an AG can contain juxtapositions of arbitrary numbers of subgraphs, while standard binary juxtaposition rules make juxtapositions of three graphs or more constructible in multiple ways. We shall start from the juxtaposition rule we listed above, which can be applied to any number ("two or more") of graphs at the same time, but there would be a similar issue if we only permitted the juxtaposition of two graphs at the same time (as Shin does in some contexts).[10] To understand the difficulty, consider the AG of Figure 5(a), where three graphs are juxtaposed. Keeping the order of subgraphs fixed (to separate the issues), this AG can be constructed by applying juxtaposition in different ways: we can apply it to the three subgraphs at once, yielding Figure 5(b), or we can limit ourselves to juxtaposing two subgraphs at the same time, which leads to two possible trees, Figures 5(c) and 5(d). In the latter cases, we can straightforwardly apply the basic translation method to the resulting trees and get the formulas

$$((\neg P \wedge Q) \wedge \neg R) \qquad \text{and} \qquad (\neg P \wedge (Q \wedge \neg R)),$$

respectively. In the former case, the basic translation method is not immediately applicable at all, unless we decide to use a target language with a conjunction of variable arity.

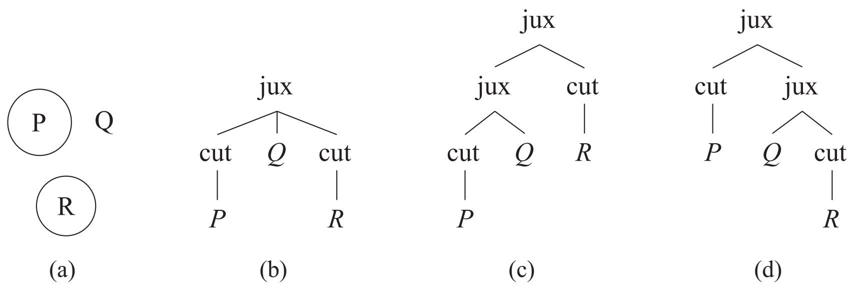


Figure 5. An AG with three possible parsing trees.

Surprisingly, these problems are rarely addressed in the literature on AGs. Shin claims that the formation rules she initially gives (essentially the same as those we started from) "guarant[ee] a unique building tree for each graph,"[11] glossing over the problems just mentioned. Other sources, like Hammer (1996) and Roberts (1973), tend to ignore unique parsability altogether. Curiously, the only thorough discussion we found is also the earliest, that

[10] See for instance Shin (2002, 74).

[11] See Shin (2002, 74–75), where she refers to the set of graphs defined inductively by the rules she gives on p. 38.

given in Jay Zeman's PhD thesis (Zeman 1964, unpublished but now available on the internet). But as we shall see, his treatment of the ordering issue is needlessly complicated, which may be why his careful formulation was ignored in later works.

At any rate, both difficulties (non-linearity and multiple juxtapositions) can be circumvented. To avoid the second one, we only need to alter the juxtaposition rule to preclude the juxtaposition of graphs that have themselves been obtained by juxtaposition; this amounts to prohibiting trees from having consecutive "jux" nodes ("jux" nodes with another "jux" node as a direct child). In other words, when constructing a specific AG, we always have to apply the juxtaposition rule to the greatest possible number of subgraphs at the same time, which rules out the parsing trees shown in Figures 5(c) and 5(d). Technically, one could make this explicit in the formation rules for AGs by adopting the following in place of Rule 2, as does Zeman (1964):

2′. The juxtaposition of two or more AGs that were obtained by rules 1 or 3 is an AG.

To resolve the first difficulty, Zeman (1964, §I.1) noted that, given an ordering of propositional symbols (letters), it is in principle possible to order all AGs. This suffices to guarantee that any AG admits a canonical parsing (in which juxtaposed subgraphs are always listed in order), but without exhibiting this parsing. While it is possible to *explicitly* construct a numbering of AGs (and thus explicitly specify a canonical parsing for every graph), doing so would be, in Zeman's own words, "long, tedious, and irrelevant." We suggest, instead, to change the kind of abstract structure (and thus the kind of parsing tree) that we use. Recall our discussion above: at bottom, the purpose of having a unique abstract structure (encoding the construction history of a formula in a unique way and representable as a tree) is to allow inductive definitions. It simply turns out that, in the case of AGs, the structures adequate for this task should not come with an ordering among juxtaposed subgraphs; in other words, to represent AGs' abstract structure, we need to use parsing trees *not equipped with any ordering among sibling nodes*. (Mathematically, one could describe them as trees without a planar embedding; at the level of the graphical representation of trees, it amounts to treating tree diagrams that only differ by the ordering of sibling nodes as tokens of the same type.) As discussed above, the main role of parsing trees is to permit inductive definitions. The slight change just described in the nature of parsing trees means that inductive definitions, too, should be slightly different for AGs: when defining a property inductively for the juxtaposition of several AGs, one should be careful *not* to rely on any ordering among the juxtaposed subgraphs. The "basic translation method" sketched in the introduction (using an implicit inductive definition) violates this restriction: as its output is in a linear language that requires an ordering among conjuncts, it has to rely on some choice of ordering among juxtaposed subgraphs.

In summary, AGs can be given unique abstract structures (representable as parsing trees) just like propositional formulas, despite the fact that they are non-linear and allow for simultaneous juxtapositions of an arbitrary number of subgraphs. Thus, one can use inductive definitions for AGs (as long as one is careful not to rely on any particular ordering among juxtaposed graphs). Nevertheless, the abstract structures of AGs are not exactly the same as those of linear languages. This has an important consequence: any translation of AGs into a notation that is linear and whose conjunction is binary (rather than of variable arity) is underdetermined. In other words, multiple translations of an AG into such a notation are in general possible. As we shall see presently, however, Shin's notion of "multiple readability" is altogether different.

2. Multiple readings for AGs and sentential notations

We now turn to what Shin calls the "multiple readability" of AGs. We first present her idea, distinguishing it from the phenomena discussed above and devoting some effort to clarifying the way she uses "linearized" AGs to bracket order issues (Section 2.1). We then argue that multiple readability can be replicated for *any* language of classical propositional logic, and thus that the various explanations of it advanced in the literature (be it by Shin or by her critics) are insufficient (Section 2.2). Finally, in addition to just refuting Shin's conclusion, as previous commentators have done, we also show what is wrong with Shin's *argument* that the multiple readability of AGs is due to the fact that their non-linearity allows for multiple *parsings* of them, while linear notations cannot be parsed in more than one way (Section 2.3).

2.1. *Shin's multiple readings*

Let us revisit the AGs shown in Fig. 1(a) of the introduction. The basic translation method yields $\neg(P \wedge \neg Q)$ and $\neg(\neg Q \wedge P)$ as possible translations. Now, notice that these formulas are logically equivalent to $P \to Q$: Shin's idea is that we can directly translate similar graphs, where one cut is enclosed in another, as implications.[12] (Historically, Peirce regarded implication as the most fundamental connective and thus considered this pattern, where one cut is enclosed in another, as primitive; he called it a "scroll" and often drew it as a unit by connecting the inner and the outer

[12] See Shin (2002, 72–74).

cut.[13]) This leads to alternative translations for the AG in Fig. 1(b): in addition to the basic ones given above, we can get $(P \to Q) \to R$, or $\neg((P \to Q) \wedge \neg R)$, or $\neg R \to (P \wedge \neg Q)$, depending on which of the various scrolls we choose to consider; since $\neg(P \wedge \neg Q)$ is logically equivalent to $P \to Q$, all of these translations are logically equivalent as well.

In the same spirit, the basic reading of the AG in Fig. 6 is $\neg(\neg P \wedge \neg Q)$, which is logically equivalent to $P \vee Q$; Shin thus suggests that the pattern exhibited here – in which several cuts are enclosed in a bigger one – can be directly translated as a disjunction.[14] (Here again, Peirce himself sometimes emphasized this disjunction pattern by drawing the inner cuts connected to the outer cut.[15])

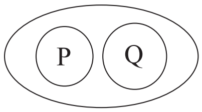


Figure 6.

Shin codifies this flexibility with regard to alternative translations by formulating a list of "reading" rules (which are translation rules from AGs into our propositional calculus with connectives $\neg$, $\wedge$, $\vee$, $\to$), shown in Table 1.[16] To understand this table, a few preliminaries are in order. First, it is important to note that Shin phrases her rules in terms of *linearized* AGs: she relies on the observation, already made by Peirce,[17] that any AG can be reorganized so that all letters lie on a line, and can then be rewritten as a string of symbols, with square brackets [,] instead of cuts – for example, the AG in Fig. 1(a) then becomes $[P[Q]]$.[18] (This device allows Shin to hide the problems raised by the non-linearity of AGs discussed in the previous section, but, as we shall see below, it is somewhat ambiguous.) Furthermore, the following conventions are used in Table 1: Greek letters stand for arbitrary AGs; the translation of α by any of the rules is written as α'; and propositional variables, denoted by capital Latin letters, are always left invariant by the translations. To illustrate the translation rules, here is how Rule 1 from Table 1 (which can be read

[13] See for instance Peirce (1976, III/1, 409–416) or Peirce (1931–1958, §4.564); for discussion, see Roberts (1973, 34–35) or Bellucci and Pietarinen (2016, 218–220).

[14] See Shin (2002, 64–66).

[15] See Peirce (1931–1958, §4.457).

[16] See Shin (2002, 75) for her list of rules.

[17] Peirce (1902) indeed starts by writing AGs on a line with brackets, braces and parentheses instead of cuts, before moving on to his usual notation in more complicated cases. The reasons may have been typographical.

[18] When using this linear notation, one should keep in mind that, as discussed above, AGs do not come equipped with any ordering among their letters: $[P[Q]]$ and $[[Q]P]$ are the same AG, namely that of Fig. 1(a). See our discussion below as well as Hammer (1996, 133–134).

as "If α' is a translation of α, then $\neg\alpha'$ is a translation of $[\alpha]$") is applied to the AG $[B]$: After parsing this graph as $[\alpha]$, where α is the AG B, the rule yields $\neg\alpha'$, where α' is the result of applying the rules to B. Since propositional variables are not affected by a translation,we finally get the formula $\neg B$ as translation of the AG $[B]$.

The crucial feature of Shin's rules is that they are deliberately redundant: when translating (or in Shin's terminology, "reading") an AG, it will often happen that several rules are applicable. The idea is that a user of AGs can choose to apply one or the other, depending on their goals or on which translation patterns happen to strike their eye.[19] Notice that the first two rules correspond to what we earlier referred to as the basic translation method, the third to Peirce's scrolls (read as implications), the fourth and fifth to two ways of recognizing disjunctions. To see how this leads to multiple translations for a single AG, let us walk through the example given in Table 1 just below the second horizontal line.[20] Starting from $[[A][B]]$, the basic reading method amounts to applying rule 1 to the full graph (so $\alpha = [A][B]$), rule 2 to $[A][B]$ (so $\alpha = [A]$ and $\beta = [B]$), and rule 1 twice to $[A]$ and $[B]$ (so $\alpha = A$ and $\alpha = B$, respectively); this leads to the first reading, $\neg(\neg A \wedge \neg B)$. The second reading comes from applying rule 3 to the full graph (with $\alpha = [A]$ and $\beta = B$) and rule 1 to $[A]$. The third also comes from applying rule 3, but this time with $\alpha = [B]$ and $\beta = A$, followed by rule 1. We get the fourth reading by applying rule 4 with $\alpha = [A]$ and $\beta = [B]$ and rule 1 twice. Finally, the fifth reading comes from applying rule 5 with $\alpha = A$ and $\beta = B$.

Ex. 1		**AGs**	**Prop. calc. with $\neg, \wedge, \vee, \rightarrow$**
Rules	1.	$[\alpha]$	$\neg\alpha'$
	2.	$\alpha\ \beta$	$\alpha' \wedge \beta'$
	3.	$[\alpha[\beta]]$	$\alpha' \rightarrow \beta'$
	4.	$[\alpha\ \beta]$	$\neg\alpha' \vee \neg\beta'$
	5.	$[[\alpha][\beta]]$	$\alpha' \vee \beta'$
Example of MR:		$[[A]\ [B]]$	$\neg(\neg A \wedge \neg B)$
			$\neg A \rightarrow B$
			$\neg B \rightarrow A$
			$\neg\neg A \vee \neg\neg B$
			$A \vee B$

Table 1. Shin's reading rules for AGs, with an example of multiple readability.

[19] See Shin (2002, 76–80) for her discussion of this.

[20] Note that when presenting the results of a translation, we omit outer parentheses for easier readability.

Upon closer inspection, Shin's use of linearized AGs reveals some ambiguities. Remember that Figures 2(b) and 2(c) ought to be seen as tokens of the same type; now what about, say, the linearized AGs $[P[Q]]$ and $[[Q]P]$? There, both options are open: we can take them either as tokens of the same type – as if they were AGs whose letters just happened to lie on a line, which was Peirce's own approach[21] – or as tokens of different types – treating them as we would treat expressions of a standard formal language. Depending on which option we choose, we get alternative interpretations of Shin's rule to translate the graph $[\alpha\ \beta]$ as $\alpha' \wedge \beta'$. On the one hand, if the linearized AGs $[\alpha\ \beta]$ and $[\beta\ \alpha]$ are tokens of the same type, then the translation of the underlying AG could be $\beta' \wedge \alpha'$ just as well as $\alpha' \wedge \beta'$, and the rule would be underspecified: one would be free to choose the order when translating, which yields multiple possible translations from applications of the same rule. If, on the other hand, linearized graphs are treated like a standard symbolic language, this problem does not arise, as the ordering of the linearized AG determines the ordering of the translation; this is probably what Shin intended. However, the indeterminacy of the ordering is pushed back to an implicit first step: before translating an AG, one would need to choose one linearization among several.[22]

Regardless of the previous considerations, it is clear that Shin's rules can be interpreted as providing multiple readings for a *linear* language, that of linearized AGs (in which the order of the symbols matters). Thus, it follows from Shin's own translation rules that the phenomenon of multiple readability has nothing to do with non-linearity in and of itself, as she sometimes seems to suggest. We can ask further: is multiple readability specific to AGs at all, and if so, what features of AGs does it depend on? Our goal in the remainder of this paper is to answer this question.

2.2. *Multiple readings of propositional calculi*

At first sight, Shin's "readings" that we just discussed are nothing more than translations that can be produced by a set of rules. (As we shall see in the next subsection, there is more to her idea, which explains why she speaks of "readings" rather than just of translations; but let us provisionally stick with this simple interpretation.) As pointed out by Bellucci and Pietarinen (2016, 230), this seems to make the multiple readability of a notation like AGs *relative* to a certain *target language* – in Shin's case, a propositional calculus with connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$. Her thesis would then boil down to the following: what makes AGs special is that there exists a

[21] See Peirce (1902, 645).

[22] Note that this implicit first step would play an important role for scroll patterns: under this interpretation, $[\alpha[\beta]]$ and $[[\beta]\alpha]$ are different inputs as far as Shin's rules are concerned, but there is no rule covering the second; so, before translating a scroll pattern as an implication, one (implicitly) has to choose a linearization in which the cut destined to become the consequent appears at the end of the formula.

set of rules yielding multiple possible translations of at least some AGs into our usual propositional calculus.

Thus interpreted, Shin's thesis is not true, as we shall see by discussing the examples of Table 2. Each of these examples gives a set of translation rules from a source language into a target language, followed by an example of multiple readability obtained from them; we use the same conventions as for Shin's rules above: Greek letters stand for arbitrary expressions in the source language; α' stands for any translation of obtained from the rules; and propositional variables, denoted by capital Latin letters, are unchanged by any translation.

First, as has already been pointed out by Bellucci and Pietarinen (2016, 230 sq.), we seem to have multiple readability whenever translating from a language with fewer connectives into a language with more connectives. Take Ex. 2: to translate a sentence with $\wedge$, $\vee$ into a sentence with $\wedge$, $\vee$, $\rightarrow$, a trivial method is simply to copy it without change (rules 1 and 2); but if we also learn to read the pattern $\alpha \vee \neg\beta$ as $\beta \rightarrow \alpha$, as anyone with some experience in propositional logic probably does, then we start getting multiple translations for the same sentence (rules 3 and 4). Ex. 3 is analogous, with $\wedge$ in the place of $\vee$; the patterns we need to recognize to apply the given rules are a little bit more complicated, but it is a matter of degree rather than principle.

		SOURCE LANGUAGE	TARGET LANGUAGE
Ex. 2		**Prop. calculus with $\neg, \vee$**	**Prop. calculus with $\neg, \vee, \rightarrow$**
Rules	1.	$\neg\alpha$	$\neg\alpha'$
	2.	$\alpha \vee \beta$	$\alpha' \vee \beta'$
	3.	$\neg\alpha \vee \beta$	$\alpha' \rightarrow \beta'$
	4.	$\alpha \vee \neg\beta$	$\beta' \rightarrow \alpha'$
Example of MR:		$\neg A \vee \neg B$	$\neg A \vee \neg B$ $A \rightarrow \neg B$ $B \rightarrow \neg A$
Ex. 3		**Prop. calculus with $\neg, \vee$**	**Prop. calculus with $\neg, \wedge, \rightarrow$**
Rules	1.	$\neg\alpha$	$\neg\alpha'$
	2.	$\alpha \wedge \beta$	$\alpha' \wedge \beta'$
	3.	$\neg(\alpha \wedge \neg\beta)$	$\alpha' \rightarrow \beta'$
	4.	$\neg(\neg\alpha \wedge \beta)$	$\beta' \rightarrow \alpha'$
Example of MR:		$\neg(\neg A \wedge \neg B)$	$\neg(\neg A \wedge \neg B)$ $\neg A \rightarrow B$ $\neg B \rightarrow A$

Table 2. Examples of multiple readability: embedding a language into a richer target language.

		Source language	Target language
Ex. 4		**Prop. calculus with $\neg$, $\wedge$**	**Prop. calculus with $\neg$, $\vee$**
Rules	1.	$\neg\alpha$	$\neg\alpha'$
	2.	$\alpha \wedge \beta$	$\neg(\neg\alpha' \vee \neg\beta')$
	3.	$\neg(\alpha \wedge \beta)$	$\neg\alpha' \vee \neg\beta'$
	4.	$\neg(\neg\alpha \wedge \neg\beta)$	$\alpha' \vee \beta'$
Example			$\neg\neg(\neg\neg A \vee \neg\neg B)$
of MR:		$\neg(\neg A \wedge \neg B)$	$\neg\neg A \vee \neg\neg B$
			$A \vee B$
Ex. 5		**Prop. calculus with $\neg$, $\vee$**	**Prop. calculus with $\mid$**
Rules	1.	$\neg\alpha$	$\alpha' \mid \alpha'$
	2.	$\alpha \vee \beta$	$(\alpha' \mid \alpha') \mid (\beta' \mid \beta')$
	3.	$\neg\alpha \vee \beta$	$\alpha' \mid (\beta' \mid \beta')$
	4.	$\neg\alpha \vee \neg\beta$	$\alpha' \mid \beta'$
Example			$((A \mid A) \mid (A \mid A)) \mid ((B \mid B) \mid (B \mid B))$
of MR:		$\neg A \vee \neg B$	$A \mid ((B \mid B) \mid (B \mid B))$
			$A \mid B$

Table 3. Examples of multiple readability, where the target language has the same number or fewer connectives.

However, the multiple readability phenomenon is not limited to translations into richer languages, as suggested in previous studies. First, it also arises when changing the connectives without increasing their number, as shown in Ex. 4, where we replace $\wedge$ with $\vee$. Here again, the patterns one needs to notice in order to apply the rules are not exotic; experienced users of propositional logic already recognize them because of De Morgan's laws. Second, one can even have multiple translations when *reducing* the number of connectives: translating from some usual propositional calculus into one with a single connective, the so-called Sheffer stroke, also yields multiple translations, as is illustrated in Ex. 5.

In fact, it turns out that Shin's reading rules for AGs (see Ex. 1) closely parallel Ex. 3 and 4, above. To see this, remember that the basic reading method allows one to straightforwardly translate an AG into a sentence with $\wedge$ and $\neg$ only (in fact, starting from an AG in linear notation, this translation can be done almost symbol for symbol: one just has to insert $\neg$'s in front of all opening brackets, which correspond to cuts, then add $\wedge$'s in between juxtapositions[23]). Under this correspondence, the $[\alpha[\beta]]$ pattern of Shin's

[23] More precisely, a $\wedge$ should be added wherever there is either a letter or a] followed by either a letter or a [, assuming a language with a conjunction of variable arity or with priorities

		Source language	Target language
Ex. 6		**Prop. calculus with $\neg, \vee, \wedge, \rightarrow$**	**Alpha graphs**
Rules	1.	$\neg\alpha$	$[\alpha']$
	2.	$\alpha \wedge \beta$	$\alpha' \beta'$
	3.	$\alpha \vee \beta$	$[[\alpha'][\beta']]$
	4.	$\alpha \rightarrow \beta$	$[\alpha'[\beta']]$
	5.	$\neg\alpha \vee \beta$	$[\alpha'[\beta']]$
	6.	$\neg\alpha \vee \neg\beta$	$[\alpha' \beta']$
Example of MR:		$\neg A \vee \neg B$	$[[[A]][[B]]]$ $[A[[B]]]$ $[A\ B]$

Table 4. Example of multiple readability from propositional calculi to AGs.

rule 3 (see Ex. 1) becomes the $\neg(\alpha \wedge \neg\beta)$ pattern of rule 3 from Ex. 3 – both yielding the same translation. Similarly, the $[\alpha\ \beta]$ and $[[\alpha][\beta]]$ patterns of Shin's rules 4 and 5 correspond to the patterns $\neg(\alpha \wedge \beta)$ and $\neg(\neg\alpha \wedge \neg\beta)$ of the last two rules of Ex. 4, again with the same translation. So one could argue that the multiple readability of AGs is precisely the same as that obtained by combining the rules of examples Ex. 3 and 4.

All of our examples so far have a sentential language as target; while they prove that multiple readability is not limited to AGs, they leave open the possibility that there is an asymmetry between AGs and sentential languages. This is what Shin seems to suggest: after explaining the multiple readability of AGs, she proceeds to offer "inverse" rules to translate sentences of our usual propositional calculus into AGs, but those always produce a *unique* translation[24] – thus (implicitly) reinforcing the impression that each AG corresponds to a set of sentences, but that, conversely, each sentence corresponds to no more than one AG. Ex. 6 refutes this: in addition to the inverse rules given by Shin (which are the first four of the example), one can add further ones and get multiple translations of sentences into AGs, as well. These extra reading rules serve the same purpose as Shin's original multiple readability rules, in that they allow us to exploit patterns in the source language to directly produce shorter translations in the target language.

Shin, however, argues that multiple readings like those of AGs should *not* be possible for sentential languages. To conclude this section, we discuss her argument, which clarifies why she uses the word "reading" and

instead of parentheses. This translation process is not strictly symbol for symbol, but is not far from it; from the point of view of parsing trees, it amounts to a mere relabeling of the nodes.

[24] See Shin (2002, 93–94).

will help us better understand the status of the translation rules just given. Although we conclude that her argument is not sufficient as it stands, it will ultimately help us pinpoint the specificities of AGs.

2.3. *Multiple translations and redundant formation rules*

Up to now, we have treated Shin's multiple readings as mere translations into a particular target language (our usual propositional calculus). In fact, however, Shin sees them as more than that: for her, they are made possible by multiple *parsings* of AGs, which are intrinsic to the specific notation of AGs rather than relative to a target language. This is why she speaks of multiple "readings" and sometimes also of multiple "carvings."

Shin's argument for the difference between AGs and linear languages is that one should countenance a variety of formation rules for AGs in addition to cut and juxtaposition: for instance, to the formation rules listed on p. 8 above, she adds what we can call a "scroll rule" to directly construct $[\alpha[\beta]]$ from two AGs α and β, and similarly, rules to form $[\alpha\ \beta]$ and $[[\alpha][\beta]]$.[25] This profusion of rules yields many new parsing trees, on which Shin's translation rules can be defined inductively. Consider again the AG of Fig. 1(b). In addition to the parsing tree shown in Fig. 7(a), we can parse the AG using Shin's scroll rule to get Fig. 7(b); this parsing allows for a straightforward inductive translation as $(P \to Q) \to R$. In this setting, of course, a typical AG will have *several* parsing trees, not just one; each of those will correspond to a different translation (and even to several if we take order and multi-juxtapositions into account).

Such multiple parsings, Shin argues, are not possible for sentential languages:

> It is interesting to notice that this issue of flexibility does not arise in a linear symbolic language. On the contrary, a symbolic system is very careful to prevent multiple readings of a formula, since it would yield ambiguity. Sentential languages are defined so that each sentence may have one and only one way of being read off, and the semantics is built on this unique readability.[26]

Here, Shin refers to what we called "unique parsability" in Section 1.2; her view is that, while it is a requirement for our usual sentential languages, "there is no need to keep the unique-readability principle in the Alpha system,"[27] and that it should be abandoned to be faithful to AGs' flexibility.[28]

[25] See Shin (2002, 74) for the full list.

[26] Shin (2002, 79).

[27] Shin (2002, 79).

[28] Although Shin leaves implicit the subtleties related to the variable arity of juxtaposition and the non-linearity of AGs that we discussed in section 1.2, she does claim incidentally that one can impose unique parsability upon AGs (see Shin 2002, 74–75); her argument is not that AGs cannot be made uniquely parsable, but rather that it is better to treat them as multiply parsable.

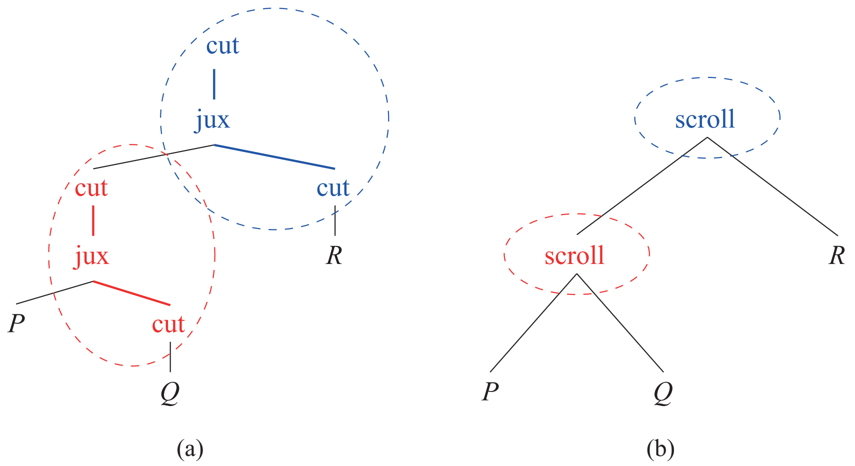


Figure 7. Parsing trees for the AG of Fig. 2; the red and blue scroll nodes of (b) can be seen as collapsed versions of the tree fragments of (a) of the corresponding color.

But why could the same not be done for sentential notations? In a sense, Shin's new formation rules are simply condensed versions of sequences of cuts and juxtapositions: as shown by the dashed lines of Fig. 7, her scroll rule corresponds to a tree fragment composed of three successive rule applications. Such condensed rules, however, can be introduced for propositional calculi as well. Take Ex. 3 from Section 2. The sentence $\neg(\neg A \wedge \neg B)$ would usually be parsed as in Fig. 8(a), but if we add a new "left-implication" rule ("from formulas α and β, construct $\neg(\alpha \wedge \neg\beta)$"), then the same formula can be parsed as in Fig. 8(b). The third translation rule of Ex. 3, which maps $\neg(\alpha \wedge \neg\beta)$ to $\alpha' \to \beta'$, can then be rephrased as a mapping from left-implication(α, β) to $\alpha' \to \beta'$.

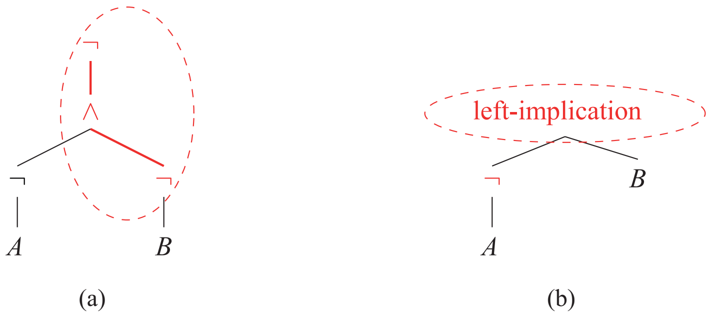


Figure 8. Parsing trees for the formula $\neg(\neg A \wedge \neg B)$: (a) with the usual formation rules; (b) with a new "left-implication" rule, corresponding to the subtree highlighted in (a).

In fact, most of the translation rules defined above may be seen to rely on what we could call "translation patterns" (or "reading patterns"): they allow for direct translations for complex combinations of basic symbols that correspond to particular fragments of parsing trees (and could be introduced by a tailor-made, additional formation rule). This, then, is the negative part of our argument: if there is a difference between AGs and our usual propositional calculi, it cannot be in the possibility of multiple translations; it can only reside in the *kinds* of patterns that underlie these translations. In the next section, we turn to such differences among patterns.

3. Contiguous and context-free translation patterns: Complex symbols

While in principle, both AGs and sentential languages admit of multiple translations, we believe that Shin's intuition is essentially correct: it is "natural" (as she phrases it)[29] to treat AGs as multiply readable in a way it is not for sentential languages. We have shown above that in both cases, multiple readability relies on translation patterns. The difference that underlies this "naturalness," we believe, can only lie in the particular properties of the translation patterns of each notation.

To investigate this issue, our strategy in Section 3.1 is to systematically compare, for various notations, patterns that can be translated as implication. Building on earlier work by Schlimm (2018) about Frege's logical notation, we identify two structural features that the translation patterns of AGs have but those of standard propositional calculi do not, namely *contiguity* and *context-freedom*. As we shall see, however, these properties are not intrinsically "diagrammatic": they are shared by patterns in various notations, some of which would doubtlessly be classified as "sentential" whichever way the distinction is drawn (if it can be drawn at all).

Section 3.2 then offers a tentative explanation, based on the psychology of perception, of why patterns that are contiguous and context-free are valuable: they form visual-semantic units that can be perceived and treated as single "complex symbols"; in this way, they facilitate multiple readings of the notations that have them.

3.1. *Translation patterns for implication in various notations*

To understand what makes AGs' translation patterns special, let us start with a series of examples. To facilitate the comparison, we focus on patterns

[29] See Shin (2002, 76–80).

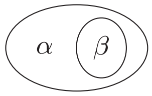
	Language	Sample pattern for $\alpha \to \beta$
1.	Prop. calc. with $\wedge$ and $\neg$	$\neg(\alpha \wedge \neg\beta)$
2.	Linearized AGs	$[\alpha[\beta]]$
3.	Prop. calc. with $\wedge$ and $\neg$, Polish notation	$\neg \wedge \alpha \neg \beta$
4.	(same language)	$\neg \wedge \neg \beta \alpha$
5.	Prop. calc. with $\wedge$ and overlines for $\neg$	$\overline{\alpha \wedge \overline{\beta}}$
6.	Linearized AGs with overlines for cuts	$\overline{\alpha\overline{\beta}}$
7.	AGs	

Table 5. Examples of patterns, in various languages, that can be translated as implications in our usual propositional calculus.

that can be translated as implication in our usual sentential language and we limit ourselves to languages that are based on conjunction and negation.[30] For each language, Table 5 shows a simple expression that can be translated as the implication $\alpha \to \beta$, using the same convention as for the translation rules of the previous section, namely that Greek letters are placeholders for arbitrary subexpressions. The pattern that needs to be recognized in each case consists of all symbols other than α and β.

Let us note from the outset that our comparison shall leave aside an important advantage that the complex symbols of AGs discussed above (namely the scroll and the pattern for disjunctions) have over the corresponding patterns for propositional calculi, namely that they apply uniformly to a wider range of cases. To clarify this idea, let us concentrate on the scroll. In the table above, we compared it to

$$\neg(\alpha \wedge \neg\beta).$$

This, however, glosses over some other features of AGs. First, the scroll pattern does not change when one adds further conjuncts, whereas the

[30] In a way, AGs can be understood as such a language, with cuts corresponding to negations and juxtapositions to conjunctions (of variable arity). At the level of parsing trees, translating into a sentential language with negation and conjunction (of variable arity) is as simple as replacing every cut node by a negation and every juxtaposition node by a conjunction (and fixing an ordering among sibling nodes).

additional parentheses would make the sentential pattern much more involved (unless we agree to drop parentheses when several successive conjunctions are involved, thus *de facto* switching to a language with a conjunction of variable arity). Second, the scroll pattern is also independent of order, whereas in the sentential case, one should technically consider not just one, but at least two different patterns (and more if more conjuncts are involved), including also

$$\neg(\neg\beta \wedge \alpha).$$

These advantages are related to the peculiarities of AGs' syntax discussed in Section 1 (non-linearity and the variable arity of juxtaposition), but are orthogonal to the main thrust of our argument, and we shall not discuss them further.

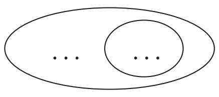
Returning to Table 5, the first feature to note is that, in examples 1 to 3, the patterns one needs to recognize are not *contiguous*: they are composed of several symbols that, in complex formulas, can end up separated from each other by many other symbols. This can be seen more clearly if we replace the place-holders for arbitrary subexpressions by dots:

$$\neg(\ldots \wedge \neg \ldots) \qquad [\ldots[\ldots]] \qquad \neg \wedge \ldots \neg \ldots$$

We see here how the basic symbols that constitute these patterns do not form a contiguous whole, but are separated by dots, which can stand for very long subexpressions. This makes it more difficult to recognize at a glance the patterns in question.

In examples 4 to 7, on the other hand, the components of the respective translation patterns do not have anything between them but white space. If we represent the subexpressions in these patterns by dots, as we have done above, we can see how the symbols that constitute the patterns are contiguous; in other words, one can draw a line connecting these symbols without having to "jump over" some dots:

$$\neg \wedge \neg \ldots\ \ldots \qquad \overline{\ldots\ \wedge\ \overline{\ldots}} \qquad \overline{\ldots\ \overline{\ldots}}$$



AGs (example 7) are something of a limit case here: while, in a sense, the inner cut need not be directly next to the outer cut, it remains the case that there must be a path going from the inner to the outer cut without crossing any other curve, otherwise we would not have a correct instance of the pattern at all. As mentioned above, Peirce himself considered our implication pattern (which he called a "scroll") as primitive and often drew it by connecting the inner and the outer cut, thus literally turning the two cuts into a single primitive symbol.[31]

A second feature, which is related but not identical to contiguity, is that the patterns in examples 4 to 7 are *context-free*, in the following sense: whenever one sees the particular arrangement of symbols that form the patterns, one can be sure that it is an actual instance of the pattern, that is, that the expressions corresponding to α and β are indeed subformulas (provided, of course, that the formula as a whole is well-formed). In examples 1 to 3, on the other hand, the visual pattern alone is not enough: on top of recognizing the pattern, one also needs to check that the expressions corresponding to α and β are actually subformulas. The following purported instance of pattern number 1, for example, is *not* actually an instance of it:

$$\begin{array}{ll} \text{Formula:} & \neg\,(\,A \vee (\,B \wedge \neg\, C\,)) \\ \text{(Purported) pattern 1:} & \neg\,(\quad \ldots \quad \wedge \neg \ldots) \end{array}$$

Indeed, the parentheses and the conjunction symbol picked out here do not belong together (remember that the formation rules of our usual propositional calculus only introduce parentheses in tandem with a binary connective; in the formula above, for instance, the first opening parenthesis would have been introduced together with the disjunction). Thus, the first dots would match $A \vee (B$, which is not a subformula. To ascertain whether a "visual" instance of patterns 1 through 3, like the one above, is an actual instance of them, one therefore needs to take the context of the symbols into account; in fact, one needs to fully parse the subformulas, even though the formula as a whole is well-formed. In examples 4 to 7, on the other hand, this is not needed: whenever one recognizes a scroll pattern in a (well-formed) AG, for instance, one can be sure that it can be translated as an implication, without attending to what is actually contained within the cuts.

Note that, against what our previous examples might suggest, contiguity and context-freedom are independent. The first is a property of the appearance of patterns, while the second depends on whether mere appearance is enough to recognize genuine instances of the pattern. Accordingly, one can have contiguity without context-freedom and context-freedom without contiguity, as the following examples show.

To get context-freedom without contiguity, start from the pattern

$$\neg(\ldots \wedge \neg \ldots)$$

discussed above. Remember that it is not context-free because one might see spurious instances of it by picking out parentheses and a conjunction

[31] See our discussion in Section 2.1 above.

symbol that do not belong together. To avoid this, one can change the formation rules of the language so as to tag parentheses and connectives, for instance replacing formation rule 3 from p. 7 by the following:

3′. If α and β are wffs all of whose tags are different, and n is a tag that does not appear in α and β, then $({}_n\alpha \wedge_n \beta)_n$ is a wff.

With formation rules like the preceding, our pattern would become

$$\neg({}_n \ldots \wedge_n \neg \ldots)_n$$

(where n stands for an arbitrary tag) and the formula that gave us a spurious match above would become, say,

$$\neg({}_2A \vee_2 ({}_1B \wedge_1 \neg C)_1)_2$$

to which the new pattern does not apply. More generally, the new pattern – though no more contiguous than before – is now context-free, as the formation rules introducing tags guarantee that the dots will always match subformulas.

One can also get contiguity without context-freedom. Take a propositional calculus with three connectives ($\neg$, $\wedge$ and $\vee$) that uses *priorities* instead of parentheses, with $\neg$ binding stronger than $\wedge$, which binds stronger than $\vee$. Then the pattern

$$\beta \vee \neg\, \alpha,$$

which can be translated as $\alpha \rightarrow \beta$, is contiguous but not context-free. Indeed,

$$A \vee \neg\, B \wedge C$$

is not an instance of it, as it is equivalent (in a notation with parentheses) to

$$A \vee ((\neg\, B) \wedge C).$$

This rewriting makes it clear that the negation symbol does not negate the full subformula following the disjunction, and so that the pattern does not actually apply.

Finally, note that contiguity and context-freedom do not straightforwardly map to simpler properties of notations, such as two-dimensionality, the number of symbols, the use of explicit connectives, or the use of explicit grouping devices like parentheses.

First, one can have contiguity and context-freedom in linear notations, as Example 4 demonstrates. Conversely, one can find patterns in two-dimensional notations that are not contiguous nor context-free. Indeed, think of a variant of Example 6 in which one can place the lines that correspond to cuts not just above the letters, but also below the letters: in the following formula

$$\overline{\mathrm{A}\underline{\underline{\mathrm{BC}}\mathrm{D}}}$$

the topmost and bottommost lines (highlighted in red) form a pattern that can be translated as an implication, but they are not contiguous.

Second, the contiguity of AGs' translation patterns is not merely due to the fact that AGs use very few explicit signs (basically, there is only one: the cut). As an extreme example, consider a propositional calculus based on a single connective, the Sheffer stroke, but written using the Polish (prefix) notation so that no parentheses are needed. This language would use a single sign beyond the propositional symbols, but the translation of

$$\mid \alpha \mid \beta\beta$$

as $\alpha \rightarrow \beta$ corresponds to a pattern that is not contiguous.

Third, Example 4 again shows that one can have contiguous and context-free patterns even when using explicit symbols for connectives (rather than merely using juxtaposition for conjunction and combining negation with a grouping device, as AGs arguably do).

Fourth, one might suspect from our examples that the use of "punctuation" signs that only serve for grouping and do not express connectives – like parentheses – is what makes patterns non-contiguous or non-context-free. But while patterns involving parentheses will indeed always be non-contiguous (but can be made context-free using labeling, as discussed above), this is not true for punctuation signs in general, as the overlines of Examples 5 and 6 show.

More generally, contiguity and context-freedom cut across any purported sentential–diagrammatic contrast. Examples 2 and 6 look like two ways of linearizing AGs, yet the scroll pattern is contiguous and context-free in one case and not in the other. Examples 3 and 4, using Polish notation, strikingly show that one can have translation patterns with different properties even in one and the same language.

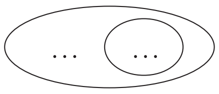
3.2. *The value of complex symbols: multiple readings and visual efficiency*

Tentatively, we propose that patterns that are both contiguous and context-free are particularly valuable for the following reasons: contiguity makes them *visually salient* by allowing them to be perceived as a unit; context-freedom guarantees that one can trust their (salient) visual appearance without further parsing. This means that in practice, such patterns can be perceived and manipulated as a single unit. Analogously, meaningful units made from simpler elements are called "chunks" in the psychological literature on expert reasoning,[32] in which they are conceived as reducing the cognitive difficulty of operating with complex structures.

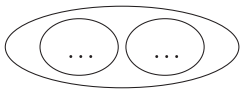
We therefore follow Schlimm (2018) in calling contiguous and context-free patterns *complex symbols*, since for most purposes, they behave as if they were a single atomic symbol. Starting from the contrast between AGs and propositional calculi, the remainder of this section explores the advantages for a notation to have such complex symbols, in particular how they can make a notation amenable to multiple readability in Shin's sense.

From this point of view, it may ultimately turn out that contiguity is too rough a notion to cover the whole range of differences in visual salience among patterns, which is largely a psychological matter. The psychology of perception recognizes so-called "Gestalt principles" that tend to determine whether individual elements are perceived as a whole.[33] Our notion of contiguity is closely related to two of them, namely the principles of "proximity" and "connectedness" (the latter ties in with Peirce's way of drawing scrolls and disjunction patterns as connected). This supports our claim that contiguity is an important factor in the visual salience of various patterns, or in other words, that contiguous patterns are easier to perceive as units. But it also means that the other Gestalt principles, which we are leaving aside here, might play an important role in some other cases.

In the case of AGs, one can easily see that Shin's multiple readability rules (presented in Table 1 above) are based on translation patterns for implications and disjunctions that are, in fact, complex symbols:



and



.

In other words, because they are contiguous, these translation patterns can be easily recognized as a group, and because they are context-free, they allow us to rely on their visual appearance without having to fully parse the AG. This agrees well with Shin's own terminology, as she refers to certain translation patterns of AGs, such as that of the scroll, as "visual features" (Shin 2002, 71).

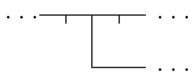
The multiple readability rules we suggested for propositional calculi (see Tables 2 and 3), on the other hand, largely rely on translation patterns that are *not* complex symbols: recognizing these patterns is harder and requires a full parsing. It seems plausible to argue on these grounds that Shin's multiple readability rules are indeed more natural, as she claims.

[32] See, for example, Miller (1956) and Chase and Simon (1973).

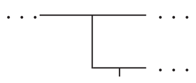
[33] For an introduction, see Rock and Palmer (1990) or Palmer (2002).

So far, we discussed AGs' complex symbols in the context of translation patterns; but their advantages are not limited to multiple readability. Indeed, the very same pattern (or complex symbol) that can be used to recognize that a translation rule is applicable can also serve to recognize premisses for any kind of inference rule. In her book, Shin discusses at length how Peirce's transformation rules for AGs can be made more "visually efficient" or "intuitive."[34] For instance, she simplifies one case of a particularly complex rule by formulating it in terms of scroll patterns: roughly, her version states that whenever you have a scroll pattern, you can add any letter located in the outer part of the scroll to the inner part of the scroll.[35] In this case, a certain pattern, the scroll, that turns out to be what we call a complex symbol, allows for a visually effective transformation rule rather than for an efficient parsing or translation.

The previous observations are not restricted to AGs: similar considerations apply to any notation that allows for the definition of complex symbols, such as the propositional calculus that uses lines (vincula) to denote negation (Example 5 in Table 5). It is also worth pointing out that other notational systems for which the authors themselves claim multiple readability, do so on the basis of complex symbols. Frege's *Begriffsschrift* notation, for example, whose propositional fragment is based on the connectives of negation and implication allows for contiguous and context-free patterns,



and



which can be interpreted as complex symbols for conjunction and disjunction (Schlimm 2018, 58–60). These complex symbols could also be used to define inference rules whose conditions of application would be particularly easy to recognize.

Conclusion

In this paper, we first showed that linear languages for propositional logic and Peirce's Alpha Graphs are more similar than initially appears. On the one hand, despite AGs' peculiarities (non-linearity and the variable arity of juxtaposition), we showed that they are uniquely parsable and thus allow for inductive definitions of properties or translations, just like propositional calculi (Section 1). On the other hand, we argued that in principle, propositional calculi are multiply readable, just like AGs (Section 2).

[34] See Shin (2002, 80–93).
[35] See Shin (2002, 89).

We then compared the patterns that underlie the multiple readings of AGs and of our usual sentential notations, and argued AGs are indeed more amenable to multiple readability because of structural features of their translation patterns: they are contiguous and context-free (Section 3). We believe that the presence of such patterns, which we call "complex symbols," is what underlies Shin's intuition that AGs are multiply readable while our usual propositional calculi are not. Yet, against Shin, there does not seem to be anything intrinsically "diagrammatic" about these properties: languages that we would not usually consider diagrammatic (like a propositional calculus with an overhead line as negation) can have patterns that are contiguous and context-free, too.

Acknowledgments

We wish to thank Moritz Bodner, Jessica Carter, Silvia De Toffoli, Valeria Giardino, John Mumma, Julien Ouellette-Michaud, and two anonymous reviewers for discussion and comments on an earlier draft of this paper. This research was supported by the Social Sciences and Humanities Research Council, Canada.

References

[1] Bellucci, F. and Pietarinen, A.-V. (2016), 'Existential graphs as an instrument of logical analysis: Part I. Alpha', *The Review of Symbolic Logic* **9**(2), 209–237.

[2] Chase, W. G. and Simon, H. A. (1973), 'Perception in chess', *Cognitive Psychology* **4**, 55–81.

[3] Hammer, E. (1996), Peircean graphs for propositional logic, *in* G. Allwein and J. Barwise, eds, 'Logical Reasoning with Diagrams', number 6 in 'Studies in Logic and Computation', Oxford University Press, New York and Oxford, pp. 129–147.

[4] MacFarlane, J. G. (2000), What Does It Mean to Say That Logic Is Formal?, PhD thesis, University of Pittsburgh, Pittsburgh.
URL: *https://www.johnmacfarlane.net/dissertation.pdf*

[5] Miller, G. A. (1956), 'The magical number seven, plus or minus two: Some limits on our capacity for processing information', *Psychological Review* **63**, 81–97.

[6] Palmer, S. E. (2002), Visual perception of objects, *in* A. F. Healy and R. W. Proctor, eds, 'Experimental Psychology', number 4 *in* I. B. Weiner, ed, 'Handbook of Psychology', Wiley, New York, pp. 179–211.

[7] Peirce, C. S. (1902), Symbolic logic or algebra of logic [passage on existential graphs], *in* J. M. Baldwin, ed., 'Dictionary of Philosophy and Psychology', Vol. II, Macmillan, New York and London, pp. 645–650. Reprinted as Peirce (1931–1958, §4.372–392).

[8] Peirce, C. S. (1931–1958), *Collected Papers*, ed. by C. Hartshorne, P. Weiss, and A.W. Burks, The Belknap Press of Harvard University Press, Cambridge, MA.

[9] Peirce, C. S. (1976), *The New Elements of Mathematics*, Mouton Publishers, The Hague, and Humanities Press, Atlantic Highlands, N.J.

[10] Peirce, C. S. (2020–2021), *Logic of the Future: Writings on Existential Graphs*, ed. by A.-V. Pietarinen, numbers 1–3 *in* 'Peirceana', De Gruyter, Berlin.

[11] Roberts, D. D. (1973), *The Existential Graphs of Charles S. Peirce*, number 27 *in* 'Approaches to Semiotics', Mouton, The Hague and Paris.

[12] Rock, I. and Palmer, S. (1990), 'The legacy of Gestalt Psychology', *Scientific American* **263**(6), 84–90.

[13] Schlimm, D. (2018), 'On Frege's Begriffsschrift notation for propositional logic: Design principles and trade-offs', *History and Philosophy of Logic* **39**(1), 53–79.

[14] Shimojima, A. (1999), 'The Linguistic-Graphic Distinction: Exploring Alternatives', *Artificial Intelligence Review* **13**(4), 313–335.

[15] Shin, S.-J. (2002), *The Iconic Logic of Peirce's Graphs*, Bradford Books—MIT Press, Cambridge, Mass. and London.

[16] Shin, S.-J. (2011), 'Peirce's alpha graphs and propositional languages', *Semiotica* **186**, 333–346.

[17] Shin, S.-J. (2015), 'The mystery of deduction and diagrammatic aspects of representation', *Review of Philosophy and Psychology* **6**(1), 49–67.

[18] Stenning, K. (2000), Distinctions with differences: Comparing criteria for distinguishing diagrammatic from sentential systems, *in* M. Anderson, P. Cheng and V. Haarslev, eds, 'Theory and Application of Diagrams (First International Conference, Diagrams 2000)', number 1889 *in* 'Lecture Notes in Artificial Intelligence', Springer, Berlin and Heidelberg, pp. 132–148.

[19] Zeman, J. J. (1964), The Graphical Logic of C. S. Peirce, PhD thesis, University of Chicago, Chicago.
URL: *https://users.clas.ufl.edu/jzeman/graphicallogic/*

Dirk Schlimm
Department of Philosophy, McGill University
dirk.schlimm@mcgill.edu
https://www.cs.mcgill.ca/~dirk/

David Waszek
Department of Philosophy, McGill University
david.waszek@mcgill.ca
https://www.normalesup.org/~waszek