Synthese (2019) 196:1501–1528
https://doi.org/10.1007/s11229-016-1244-4

S.I.: UNITY OF STRUCTURED PROPOSITIONS

A general argument against structured propositions

Peter Pagin[1]

Received: 14 February 2016 / Accepted: 4 October 2016 / Published online: 14 November 2016

Abstract The standard argument against ordered tuples as propositions is that it is *arbitrary* what truth-conditions they should have. In this paper we generalize that argument. Firstly, we require that propositions have *truth-conditions* intrinsically. Secondly, we require strongly equivalent truth-conditions to be identical. Thirdly, we provide a formal framework, taken from *Graph Theory*, to characterize structure and structured objects in general. The argument in a nutshell is this: structured objects are too fine-grained to be identical to truth-conditions. Without identity, there is no *privileged* mapping from structured objects to truth-conditions, and hence structured objects do not have truth-conditions intrinsically. Therefore, propositions are not structured objects.

Keywords Propositions · Structure · Graph theory · Truth-conditions · Unity

1 Introduction

This paper concerns the question whether propositions are structured. The idea that propositions are structured has had both proponents and critics. Among models that have been suggested are

- Ordered tuples (e.g. Russell 1903 (ascribed); Cresswell 1985)
- Trees or facts about trees (e.g. Lewis 1970; King 2007, 2014a)
- Mental acts/speech acts (Soames 2010, 2014; Hanks 2011, 2015)
- Abstract procedures (e.g. Duzi et al. 2010)
- Elements of intensional algebras (e.g. Bealer 1993)

✉ Peter Pagin
peter.pagin@philosophy.su.se

1 Department of Philosophy, Stockholm University, 106 91 Stockholm, Sweden

Many have criticized the ordered-tuple model, for instance Bealer (1993, p. 22), Soames (2010, pp. 29–32), King (2014b), Jespersen (2003). Cresswell (2002) have argued on the basis of properties of logical operators that propositions cannot be significantly structured at all. Russell (1903, pp. 47–48) argued that analysing propositions into parts irreparably destroys the essential unity of the proposition.

A number of issues have been taken to be *the* or *a* problem of the unity of the proposition, or an *aspect* of the problem. King (2009, p. 258) lists three questions[1]:

(UNITY) a. What holds the constituents of the propositions together?
b. How does a structured complex have truth-conditions?
c. Why does it seem that some constituents can be combined into a proposition and others can't?

Jespersen (2012, p. 620) takes (a) above to be the main question of unity. Others, such as Soames (2014, p. 97) and Hanks (2015, pp. 42ff.) take (b) to be the main problem of unity. In this paper, too, I shall be concerned with the (b) question of unity. I shall focus on the problem of structured propositions and truth-conditions.

I shall here generalize the criticism directed at the ordered-tuple model or propositions, to argue that no entity can meet the requirements of being both a *structured object* and a proposition.[2] In so doing, it will of course be crucial what is meant by 'proposition' and what is meant by 'structure'. I shall say a little, but hopefully enough, about what propositions are, in the next section. I shall elaborate on what structure is in Sect. 3. In Sect. 4 I shall spell out the conditions for being a structured object, to a large extent by means of the account of structure in Sect. 3. The argument that is enabled by these accounts is offered in Sect. 5. The account is applied to King's model in Appendix 1. Appendix 2 provides a proof for a central claim in Sect. 3.

2 Propositions

So, what are propositions? A sound and well established strategy for answering questions of the type "What is X?" is to separate two aspects of the inquiry: Firstly, what are the *functional* properties, or "job description" of the concept of X? Secondly, which entities do, or could, *satisfy* these functional properties? When it comes to propositions, an answer to the first question would consist in giving a list of features like the following:

(PROP) *Propositions* are
(1) the objects/contents of propositional attitudes: beliefs, desires etc;

[1] King states the question specifically with respect to his own theory and his own example. I have changed the formulations to abstract from that.

[2] I want to declare at the outset that I am not against *structured meanings*. In fact, I think that both structured *and* unstructured meanings are needed for a full account of language and communication, e.g. in the semantics for belief sentences. I have in earlier work already combined structured and unstructured meanings (Pagin and Pelletier 2007; and more is in preparation). It is just that I think that propositions belong to the *unstructured* side, given their essential characteristics.

(2) intrinsically bearers of truth-conditions[3];
(3) the contents of assertions (and perhaps other speech acts);
(4) the meanings of meaningful (declarative) sentences (in context);
(5) the primary bearers of *modal* properties; being necessary or possible;
(6) the bearers of probabilities;
(7) the referents of *that clauses*.

I take the first item on this list to be close to a platitude. We do believe such things as *that it is raining (at a particular time and place)*, or *that London is the capital of the UK*, or again *that seven is a prime number*. When we specify *what* someone believes, we specify the *content* of her belief, and what we specify then is a proposition.

The second item is a little more technical, in two respects. The adverb 'intrinsically' is meant here to indicate that whether a proposition is true or false, in relation to Reality, or to a possible world, a world-time pair, or whatever the point of evaluation be, no arbitrarily selected *third* factor is needed: no interpretation, assignment, or projection that is it not *privileged*. If an interpretation or projection is privileged, then it is not chosen by stipulation, and it does not have alternatives that are equally good. This condition is clearly met if propositions simply *are* truth-conditions. More generally, we can say that for propositions to have truth-conditions *intrinsically* is for there to be a *privileged*, non-arbitrarily selected, universal function T such that for any proposition p, $T(p)$ is the truth-condition of p. In case propositions simply *are* truth-conditions, T is the identity function.[4]

Since the requirement that there be a privileged function from propositions to truth-conditions will be central to the argument, some clarifications are in order. I will *not* claim that there are no functions from propositions* (defined below) to truth-conditions. Plainly there are. The problem will be that there are *many*, and that none is *privileged*, i.e. none is determined as the *right*, or *appropriate* one. The idea of being a *privileged* function T is epistemic rather than metaphysical. I cannot rule out that there is an unknown or even unknowable metaphysical ranking among functions that makes some candidates better than others, even if we cannot understand why. But if there is one, it does not matter. What matters is that we can see a reason to rank some candidate function T above all others, and if we don't know of any such reason for any candidate, no candidate is privileged, in this epistemic sense. This is precisely the sense, I take it, in which authors such as Bealer (1993, p. 22) have criticized the ordered tuple model of propositions: for all we can see, there is no good *reason* to identify an ordered tuple with one proposition rather than another, or a proposition with one tuple rather than another.

This is of course also the strategy in Benacerraf's (1965) argument against identifying numbers with sets. I shall refer to arguments from the lack of a better candidate

[3] Some stress that propositions are the *primary* bearers of truth and falsity, or of truth-conditions, but this issue is immaterial in the present context.

[4] You might think that it is enough that for each proposition p *there is* a function T_p that maps p on its proper truth-conditions, but that there need not be a universal function that works for all propositions. But either T_p is not determined by p itself, in which case there is then no privileged function, or else T_p *is* determined by p and the value of some general function U from propositions to functions, and we define T so that for any proposition q, $T(q) = (U(q))(q)$.

as *Benacerraf-style* arguments. In Sect. 5 I shall spell out exactly where we need to appeal to a Benacerraf-style argument.

I am not here going to argue that (PROP2) *is* an ingredient in our general concept of a proposition, but rather take that for granted. I am aware that it is a controversial idea that an entity can have truth-conditions intrinsically, independently of human activity, although it does depend on what we are prepared to regard as an entity. I assume in this paper that we can we speak of truth-conditions themselves as if they are entities.[5]

The second technical point concerns the individuation of truth-conditions. When are the conditions *that* p *the same* as the conditions *that* q? Are the conditions *that not* p the same or different from the conditions *that not not not* p? How we individuate truth-conditions is crucial to the question whether propositions are structured. As it will turn out, if truth-conditions are fairly *coarse-grained*, then propositions will not be both structured and have intrinsic truth-conditions. And I will here take truth-conditions to be fairly coarse-grained, according to the general idea that equivalence amounts to identity:

(1) If truth-conditions c are logically/analytically equivalent to truth-conditions d, then c is identical to d.[6]

This principle is of course not completely precise as long as we do not specify the relevant logic, or what analyticity amounts to here. Luckily, this can be left open in the present context, since the nature of the argument will depend only on finding uncontroversial examples, irrespective of how analyticity or logicality be delimited.

The (PROP) list can be extended with more items of the kind *bearers of F*, where the noun F is related to a sentential context for *that* clauses. Thus (PROP) *may* contain too little to fully characterize propositions. It may also contain too much, in the sense that not all the items on (PROP) are uncontroversial. For instance, according to MacFarlane's (2014) brand of relativism, a proposition and a world are not together enough to determine a truth value. Over and above the world, or world-time pair, or whatever objective reality contributes, a *context of assessment* is needed to determine the truth value of a proposition, provided it does have assessment-relative ingredients. The context of assessment does not, on MacFarlane's account, constitute a factor that determines what propositions are, but is added on top of such factors for determination of truth value.

The current assumption will only be that (PROP2) *does* characterize propositions, together with (1). Since the argument will be negative, this will be enough for present purposes. I shall use 'proposition*' for propositions as characterized by (PROP2) and (1).

[5] King (2009, pp. 259–60) says that he cannot see how "how propositions or anything else could represent the world as being a certain way *by their very natures and independently of minds and languages*". This is not a topic for the present paper. I argue that nothing that does have truth-conditions intrinsically is a structured object, and this is claimed to hold whether there are such things—propositions*—or not (trivially if not).

[6] I don't mean by this notation that truth-conditions are individuals, and that is also why I don't here use '='. However we elect to treat truth-conditions ontologically, identity must satisfy reflexivity and (a counterpart to) Leibniz's law in reasoning about them.

The second step of the strategy mentioned above is to consider independently specified candidates for being proposition*, i.e. for satisfying the functional role of propositions. I shall now exemplify this step with the most common type of proposal for structured propositions.

A standard approach to structured meanings is that of the set-theoretic construction of an ordered n-tuple $\langle a_1, \ldots, a_n \rangle$. Ordered n-tuples can be defined in terms of *unordered* sets in several ways.[7] What makes them *ordered* are their identity conditions: $\langle a_1, \ldots, a_m \rangle$ is identical to $\langle b_1, \ldots, b_n \rangle$ if, and only if, $m = n$ and for $1 \leq i \leq m$, $a_i = b_i$. Hence, it matters to the identity of an ordered tuple *where* in the tuple an element a occurs (it can occur in more than one place).

On the ordered tuple-approach (see, for instance Cresswell 1985), the proposition *that Rab* is modeled, for instance, as the triple $\langle R, a, b \rangle$, or the pair $\langle R, \langle a, b \rangle \rangle$, where the latter itself has an ordered pair as its second element. On the first alternative, the proposition is a thought of as a triple where the first element, R, is a relation (in extension or in intension), and the two following arguments, a and b, are either *individuals* or intensions (individual concepts) of individuals.

Although this is only a hint of what a full theory of propositions along these lines would look like, it is enough to illustrate the problems that have been pointed out. Firstly, are ordered n-tuples plausible candidates for being objects of belief? As has been stressed many times, the answer is no. We don't really understand what it would be to *believe* $\langle R, a, b \rangle$, as little as we understand in general what it is to believe any particular *set*. One can of course *add* the explanation that to believe $\langle R, a, b \rangle$ is just to believe *that Rab*. It is fine to add this explanation, but the problem is that it is needed. By adding this explanation we implicitly *extend* the meaning of 'believe' so that the expression 'believe $\langle x, y, z \rangle$' is understood as meaning the same as 'believe *that* xyz'. This does not answer the question what it was to believe an ordered tuple in the first place, prior to the extension, only adds a stipulation concerning a *new* sense of 'believe'.

This problem is clearly related to the second: what is it for a triple like $\langle R, a, b \rangle$ to be *true*? Intuitively, we would think that $\langle R, a, b \rangle$ is true just in case a is related by R to b. But, as has been pointed out many times, this is certainly not the only way to assign truth-conditions to $\langle R, a, b \rangle$. We might also say that this triple is true just in case b is related by R to a. Or we could have some function f such that $\langle R, a, b \rangle$ is true just in case $f(a)$ is related by R to $f(b)$, or yet something else altogether. The sky is the limit. The main point is that the predicate '…is true' is not defined for ordered tuples, and although the meaning of the predicate can be *extended* to cover ordered tuples, this can be done in many different ways. None of these ways is *privileged*, whether or not selected by any feature of the tuples themselves as the right way. Hence, ordered tuples are clearly not *intrinsically* bearers of truth and falsity. We need to add something to make the connection, like a redefinition of 'true' or an assignment of a new significance to the ordered tuple operation. We can characterize such extensions as providing a function T from ordered tuples to truth-conditions, and the problem is that it is arbitrary which such function T to select.

[7] Standardly, $\langle a, b \rangle$ is defined as $\{\{a\}, \{a, b\}\}$.

We conclude that the ordered tuple version of structured meanings does not satisfy the functional properties of propositions. But maybe there are *other versions* of structured meanings that do. For instance, King (2007, 2014a, b) claims to have provided a model that does combine internal structure with intrinsic truth conditions. In this paper, I shall argue that *no* model of structured propositions can have intrinsic truth-conditions. I shall then turn to King's proposal in the light of this argument (see Appendix 1).

It should be noted that it is not only significantly structured models of propositions that fail to satisfy (PROP2). The standard possible-worlds model, i.e. a set of possible worlds, fails as well. That a world w is a member of a set of worlds Q is an intrinsic property of Q, given the extensionality of sets. But that Q is *true* at w requires that we map the condition of membership on the condition of truth (usually effected indirectly in formal truth definitions). Typically, Q is taken to be true at w just in case $w \in Q$, but this is clearly not the only possibility. The obvious alternative is to take Q to be true at w just in case $w \notin Q$. This gives the dual of the standard notion. It is a more cumbersome and less natural alternative, but perfectly workable.[8]

3 Structure

I shall here suggest a formal framework for talking about structured entities. The most basic idea is that if an object is structured, then it has at least one *proper part*, i.e. distinct from the (main) object of which it is a part. An object that has a proper part is thereby *complex*. An absolutely unstructured object is *simple*. A *minimally* structured complex object is one where there are no relevant differences between the parts, and a *maximally* structured object is one where there are relevant differences between *all* the parts. The framework will allow a more precise characterization of partially structured objects.

To set out these intuitive ideas with greater mathematical precision I shall avail myself of *Graph Theory*.[9] A *graph* G is a pair (V_G, E_G) of a set of *vertices*, or *nodes*, V_G and a set of *edges* E_G. E_G is a binary relation over V_G. In the basic case, the edge relation E is symmetric, irreflexive, and non-transitive. Graphs are abstract and in themselves non-visual, but are standardly pictorially represented. For instance, let the graph G be given as in Fig. 1. In this case, the set of vertices is $V_G = \{a, b, c, d\}$, and the set of edges $E_G = \{(a, b), (a, c), (a, d), (b, c), (b, d)\}$ (sometime more compactly written '$\{ab, ac, ad, bc, bd\}$'). Every node is connected to every other except c to d.

The graph G in Fig. 1 is a *labeled* graph, in that the vertices carry labels (the letters), and it is *connected* graph, in that every vertex is reachable from every other along the

8 Do the characteristic functions, i.e. functions from worlds to truth values, fare better? Analogously, we need to stipulate conditions for functions to be true, and the non-traditional conception that takes a function as true at w iff it maps w on 0, or on Falsity, or $\bot$, is perfectly coherent. In additions, there is a problem with (PROP1), both for sets and functions, for it is not clear what it consists in to believe a set or a function. Again, you can stipulate what it should amount to, but that wouldn't answer the question what it does amount to before the stipulation.

9 Chartrand and Zhang (2012).

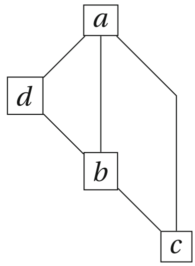


Fig. 1 A basic, connected, and cyclic graph

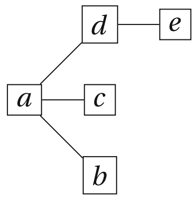


Fig. 2 A basic tree

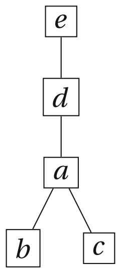


Fig. 3 The same tree as in Fig. 2

edges. That is, we can define the *transitive closure* $\mathbf{E}_G$ of E_G, and every pair (v_i, v_j) of distinct vertices in V_G is in $\mathbf{E}_G$.[10]

G in Fig. 1 is also *cyclic*. That is, it contains at least one *cycle*, leading from a vertex v along edges in E_G back to v. That is, there is at least one vertex v (at least two in the case E is irreflexive) such that $\mathbf{E}_G(v, v)$.

In characterizing structured objects we shall be interested in *trees*. A *tree* T is a connected *acyclic* graph (not containing cycles), for instance as depicted in Fig. 2.

The same tree could equally well be depicted as in Fig. 3. There is no privileged vertex or orientation in a basic tree. We shall, however, be interested in *rooted* trees, where one particular vertex, the *root* of the tree, has a privileged role. Here the root of a tree will be depicted as circled, and it is customary to depict rooted trees with the root at the top, "growing" downwards. Letting the a vertex be the root, we would thus get the depiction of Fig. 4. With a selection of a privileged vertex, a rooted tree T is a triple (V_T, E_T, v) of a set of vertices V_T, a set of edges E_T over V_T, and a privileged member $v \in V_T$.

[10] The *transitive closure* $\mathbf{R}$ of a relation R is defined inductively as the smallest set X of pairs such that (i) $R \subseteq X$; (ii) if (a, b) and (b, c) are in X, then (a, c) is in X.

Fig. 4 A basic rooted tree

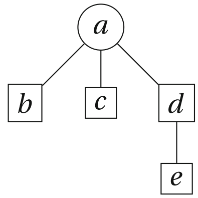
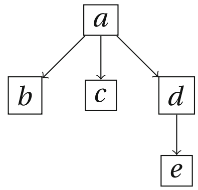


Fig. 5 A directed graph



The selection of a root imposes a direction on the edges, since in each pair of adjacent vertices, one is closer to the root than the other. A rooted tree is therefore equivalent to an un-rooted tree where all the edges are *directed* (arrows), all pointing towards, or all pointing away from, some particular vertex, as in Fig. 5.

In a *directed graph* G, the edge relation E_G is not symmetric, and if all edges are directed and there is only one edge between every two nodes, *asymmetric*, as in Fig. 5.

Here, we shall work with rooted trees, since it is natural to let the root represent the whole structured object itself, the edge relation a relation of *immediate part of*, and the non-root vertices the parts, immediate or mediate, of the whole object. We shall from now on take the edge relations to be asymmetric. With an eye to the intuitive part-of relation, we shall let the second argument of E be the one closer to the root. Therefore, if v_0 is the root T, it holds that there is no $v \in V_T$ such that $(v_0, v) \in E_T$ [we shall sometimes write $E_T(x, y)$].

The *structure* of a graph is exactly what it has in common with any other graph that has *the same* structure. The relation of being *same-structured* is then more basic than the non-relational idea of a structure. Not surprisingly, two graphs G and H will be said to have the same structure iff they are *isomorphic*, i.e. iff there is an *isomorphism* between G and H.

An *isomorphism* between two basic graphs G and H is bijection $f : V_G \longrightarrow V_H$ such that for any $v, u \in V_G$, it holds that

$$E_G(v, u) \quad \text{iff} \quad E_H(f(v), f(u))$$

An isomorphism f between two *rooted* trees T and U must also satisfy the condition that where v is the root of T and u the root of U, $f(v) = u$. This condition follows from the basic isomorphism condition for rooted trees where the edge relation is asymmetric. For there is exactly one vertex in each tree that satisfies the condition that it is edge-related to no vertex (although one or more are edge-related to it), and by the basic isomorphism condition, this vertex in the one tree one must be mapped on the corresponding vertex in the other.

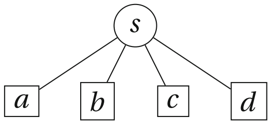


Fig. 6 A minimally structured rooted tree

A special class of bijections are *permutations*, i.e. 1–1 functions $f : V_G \longrightarrow V_G$, from a vertex set onto itself. A permutation that is also an isomorphism from a graph to itself is called an *automorphism*. That is, f is an automorphism iff it holds for any $v_i, v_j \in V_G$ that $E_G(f(v_i), f(v_j))$ iff $E_G(v_i, v_j)$.

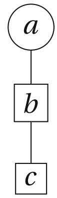
An automorphism of course maps the root on itself. In general, we shall say that a permutation f on the vertex set of a rooted tree T is *root invariant* iff $f(v) = v$ in case v is the root of T. We can define a function F_f on trees in terms of a root-invariant permutation f such that where $T = (V_T, E_T, v)$, $F_f(T)$ is defined to be $(f(V_T), f(E_T), f(v)) = (V_T, f(E_T), v)$. Here $f(E_T) = \{(f(v_i), f(v_j)) : (v_i, v_j) \in E_T\}$. To exemplify, let T be the tree of Fig. 4, and let f be the permutation that is the identity function on all vertices except that $f(e) = b$, $f(b) = e$. Then $f(E_T)$ is the set $\{ea, ca, da, bd\}$. f is here not an automorphism, since $(b, a) \in E_T$ but $(f(b), f(a)) = (e, a)$, and $(e, a) \notin E_T$. Now, in the case a permutation f *is* an automorphism, then $(f(v_i), f(v_j)) \in f(E_T)$ iff $(v_i, v_j) \in E_T$ (by the definition of F), which holds iff $(f(v_i), f(v_j)) \in E_T$ (since f is an automorphism). Hence, $f(E_T) = E_T$. So, in case f is an automorphism, the corresponding F_f is the identity function on rooted trees.

By means of the concept of an automorphism, we can characterize a *graded* notion of being structured, ranging from minimally structured to fully structured. We can now say that a rooted tree T is *minimally structured* iff *every* root-invariant permutation f on V_T is an automorphism. In such a case, there are no relevant differences between the non-root vertices of T. For instance, the tree T given in Fig. 6 has an edge relation $E_T = \{as, bs, cs, ds\}$. There are thus no structural differences between the lower vertices a, b, c, d. Each is characterized by only being edge-related to s, and so clearly any root-invariant permutation will preserve these properties. The depiction of the tree presents a left-right order between the lower vertices, but that order is immaterial, since no permutation of the lower vertices will change the edge relation. The tree of Fig. 6 therefore presents the structure of a simple (unordered) *set* s with the members a, b, c, d.

A rooted tree is *maximally* structured iff there is only one automorphism, the identity permutation (mapping every vertex on itself). A minimal example is given in Fig. 7. For this tree there are only two root-invariant permutations: the identity permutation and the permutation f such that $f(a) = a$, $f(b) = c$, $f(c) = b$. But the latter is not an automorphism, since the edge relation contains (b, a) but not $(f(b), f(a))$.

There are intermediately structured graphs. For instance, the tree in Fig. 4 admits as automorphisms *both* the identity function and the function f which is like the identity function except that $f(b) = c$, $f(c) = b$, but no other root-invariant permutation is an automorphism on this graph. In general, we can say that a graph G is *more structured* than a graph G' iff the proportion of automorphisms among the permutations of G is lower than that of G'.

Fig. 7 A maximally structured rooted tree



To be clear, that a graph G is *maximally structured* does not entail that G is very complex (a rooted tree with only one other vertex is maximally structured), nor does it entail that G has as much structure as a graph can have. We can always add other features, like a second edge relation, or an additional *order* between vertices (see below). There is no upper limit to what can be added to make graphs more structurally *complex*.

To say that a graph G is maximally structured, in the present sense, is to say that every vertex in G can be identified by its *position* in G, i.e. by which other vertices it is related to. In Fig. 7, a is uniquely identified by being the root, b by being the only vertex related to the root, and c by being the only non-root not related to the root. In Fig. 4, d is the only vertex both related to the root and to a second vertex, but b and c have the same edge properties, and cannot be distinguished except by the labels.

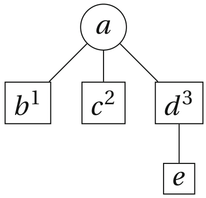
Isomorphic graphs are identical except for the labels, which in a sense are arbitrary. We can therefore regard a labeled graph as simply representing the structure that is shared with the graphs it is isomorphic to. We can also extend the isomorphism relation to hold between graphs and other complex entities that have parts according to some relevant part-whole relation. This was already exemplified above with an unordered set s, in Fig. 6. Putting these two ideas together, we can speak of some particular graph as *the structure* of a complex object to which it is isomorphic. As will be spelled out below, this will allow us to speak of structured objects more generally, without having to rely on any particular kind of objects, like ordered tuples.

For the structured objects we are interested in, however, of a representational kind, we need two additional features of graphs. The first is that of *ordering*. In a rooted tree, if a is edge-related to b, and b is closer to the root (or is the root), a is said to be a *daughter* of b and b the *mother* of c. If both b and c are daughters of one and the same vertex, b and c are said to be *sisters*. Thus, a–d are sisters in the tree of Fig. 6.

In an ordinary rooted tree, the sister relation is unordered. In an *ordered* tree, there is an ordering relation $\prec$ between sister vertices. An ordered (hence rooted) tree T is therefore a quadruple $T = (V_T, E_T, \prec_T, v)$. We write $(a, b) \in \prec_T$ as $a \prec_T b$, or simply as $a \prec b$ if the context is clear. An ordinary rooted tree is the special case where $\prec$ is empty. The definition of an *isomorphism* must also be extended. For ordered trees we must in addition to the previous conditions add that f is an isomorphism between T and U only if it holds that for any vertices v_i, v_j of T it holds that $v_i \prec_T v_j$ iff $f(v_i) \prec_U f(v_j)$.

We can indicate ordering by means of numerical indices, here appearing as superscripts, such that $v_i^k \prec v_j^l$ iff $k < l$. With the convention that if $v_i \prec v_j$, we write v_i to the left of v_j, an ordered update of the tree of Fig. 4 will be:

Fig. 8 An ordered tree



Th ordered tree T of Fig. 8 is fully structured. The function f that interchanges b and c is not an automorphism, because it does not preserve the vertex ordering: $b \prec_T c$ but $f(b) \nprec_T f(c)$.

The second feature to be added is needed because of the fact that in structured abstract entities a particular part may *occur* several times. The name 'Mary' occurs twice in the sentence

(1) John likes Mary and Mary likes Bill.

There is only one type 'Mary', but the type 'Mary' has two *occurrences* in the type (1). Other examples come from the standard definition of an ordered pair in terms of unordered sets:

(2) $\langle a, b\rangle =_{\text{def}} \{\{a\}, \{a, b\}\}$.

In the set denoted on the right-hand side, there are two occurrences of the *object* a.

In the case of entities with a type-token distinction, we can briefly characterize the ontology of occurrences by saying that the number of occurrences of x in the type y is exactly the number of *tokens* of x in any one *token* of y. Thus, any token of (1) contains two tokens of 'Mary'. We can specify them by talking of the *first* occurrence and the *second* occurrence of 'Mary' in (1), but this only works as long as we have a linear order. In more complex cases we will have to be able to specify the *position* of the occurrence in the structured abstract type, but this in turn only works if we *already* possess tools for specifying the structure, for instance by means of syntactic trees or grammatical terms, which allow you to identify the position of an occurrence.[11]

We cannot invariably specify properties of *occurrences* of objects by means of providing the properties of the objects they are occurrences of, since a particular object may have several properties, but only in different occurrences. This is illustrated in the tree of Fig. 9. Here the situation is depicted where c occurs twice, as direct part of a and as direct part of b. So, d is part of c and c is part of b, but d is *not* part of b. Thus, if we are dealing with complex objects that can contain multiple occurrences of objects, and objects are allowed to freely vary properties between occurrences, we cannot even capture the transitivity of the parthood relation by speaking only of the properties of *objects* as opposed to properties of the *occurrences* of objects.

There are two ways to handle this problem. The first is to simply treat the concept of an *occurrence* as primitive, and accept an ontology of occurrences as basic. Every occurrence is an occurrence in a type. Every type a occurs exactly once in the type a itself. Only abstract types can contain more than one occurrence of anything. If a

[11] This is spelled out e.g. in Pagin and Westerståhl (2010).

Fig. 9 Failure of transitivity

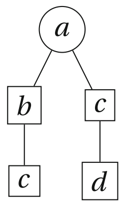
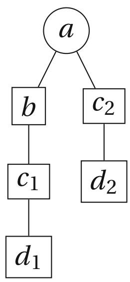


Fig. 10 Transitivity restored, from Fig. 9



physical or mental entity has parts, those parts have only one occurrence in that entity. One can introduce a function O from occurrences of objects to the objects they are occurrences of. As long as there is only one occurrence of an object, one can simplify the presentation by speaking simply of the object itself.

On this alternative, one treats the parthood and ordering relations on parts as relations over occurrences. In the tree of Fig. 9, (the occurrence of) d is part of an occurrence of c and an occurrence of c is part of (the occurrence of) b, but since these are different occurrences of c, there is no violation of transitivity in the fact that d is not part of b.

The second alternative is to *restrict* labeled trees the following way: whenever a tree contains two vertices with the same main label, these two vertices have both different indices *and* isomorphic *sub-trees* where corresponding vertices also have the same label. With this restriction, the tree in Fig. 9 is not admissible. The proper version is given in Fig. 10.

With the restriction, knowing that there is a vertex d_2 that is edge-related to a vertex c_2 and a vertex c_1 that is edge-related to a vertex b, we can infer that there also a vertex with main label 'd' that is edge-related to c_1. Thus, transitivity is restored, in the sense that if the tree represents a structured object where d is a part of c and c is a part of b, the object is also represented to the effect that d is (an indirect) part of b.

The restriction is well motivated in the sense that our ordinary conception of abstract objects is such that wherever a complex object occurs, it does occur with occurrences of its parts; they are part and parcel of *it*.

Both alternatives allow us to prove the main point, that we can use multi-labeled trees, up to isomorphism, as *the* structure of structured objects. The proof is more straightforward on the first alternative, but it carries the ontological cost of taking occurrences as primitive. There is a further difference, in that the second alternative does not allow us to use multi-labeled trees to provide the structure of so-called *multisets* (taken as primitive), i.e. sets where elements can occur more than once, as in

Fig. 11 A tree with "double" vertices

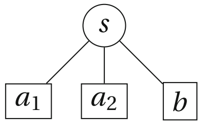
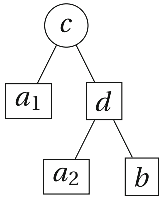


Fig. 12 A tree with multiple labels



$\{a, a, b\}$. This set can be represented on the first alternative by the tree in Fig. 11. The characteristic feature of this tree is that two vertices that have *no* structural differences still have the same main label. On the second alternative, this tree is ruled out by the uniqueness condition, either in itself, or as being the structure of an object. I see no way of accommodating multisets except by treating occurrences as basic (or instead working with a model of multisets that itself does not use multisets). Not seeing the need to handle these in the present context, I shall go for the second alternative, letting vertices directly represent objects rather than occurrences of objects.

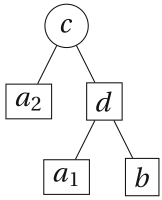
We cannot capture the phenomenon of multiple occurrences of parts in complex objects within standard graph theory. A *label* pertains to a particular vertex, and no sense is provided for letting two vertices have the same label. I shall propose here, as has already been exemplified, to use *complex* labels, with a main label and an index, such as 'b_2', where the numeral is the index. The main label can be shared by several vertices, while the complex label itself must be unique to each vertex. We shall say that trees with this feature are *multi-labeled* trees. We shall require that the root has a unique main label. By means of this device, we can represent the unordered set $\{a, \{a, b\}\}$ by the tree in Fig. 12.

Now we can introduce the idea of letting the *main label* indicate what object a vertex *represents*. That is, two vertices with the same main label, such as a_1 and a_2 in the tree in Fig. 12, will represent the same object. In addition, let the edge relation in this tree *represent* the element relation $\in$. Then d represents the set $\{a, b\}$, and c represents the set $\{a, \{a, b\}\}$.

Multi-labeled trees are still trees of the familiar kinds in the sense that each complex or simple label (without index) still is unique to a vertex. However, adding the feature of *sharing* main label amounts to adding a one-place *property* of vertices that provides a new structural dimension. We shall include the ingredient of a *labeling* function L in the definition of a multi-labeled tree. For instance, $L(a_2) =$ 'a' in Fig. 12.

Next, consider the *set* of main labels (including simple labels) in Fig. 12: that is the set {'a', 'b', 'c', 'd'}. We extend the function L to trees, giving the set of main labels of all vertices as values. We can now say that two multi-labeled graphs G and H are *label-related* iff there exists a a bijection $g : L_G(V_G) \longrightarrow L_H(V_H)$, i.e. iff the main-label sets have the same size. I shall sometimes drop the subscript on 'L.'

Fig. 13 The result of a label-preserving permutation



(MLTree) An ordered (rooted) multi-labeled tree T, or *ML* tree, is a tuple

$$(V_T, E_T, \prec_T, L_T, v)$$

where V_T is the vertex set of T, E_T the edge relation, $\prec_T$ the ordering relation between sister vertices, L_T the *labeling* function, assigning labels to vertices, and v the root of T.

With the addition of the labeling function, we also have a new requirement on isomorphisms between ordered multi-labeled trees: f is an isomorphism from T to U only if f maps same-labeled vertices on same-labeled vertices. That is,

(MLIso) Given two ML trees $T = (V_T, E_T, \prec_T, L_T, v_0)$ and $U = (V_U, E_U, \prec_U, L_U, u_0)$, the function $f : V_U \longrightarrow V_T$ is an (ML) isomorphism iff f is a bijection and
- (i) $f(u_0) = v_0$.
- (ii) For all $u_k, u_l \in V_U$, $E_U(u_k, u_l)$ iff $E_T(f(u_k), f(u_l))$.
- (iii) For all $u_k, u_l \in V_U$, $u_k \prec_U u_l$ iff $f(u_k) \prec_T f(u_l)$.
- (iv) For all $u_k, u_l \in V_U$, $L_U(u_k) = L_U(u_l)$ iff $L_T(f(u_k)) = L_T(f(u_l))$.

We get the corresponding strengthening of the notion of *automorphism* for ordered multi-labeled trees. When it comes to characterizing the structure of complex objects by means of ML trees, the requirement of an automorphism for ML trees is sometimes too strong. Thus, consider the permutation f that is the identity permutation except that it maps a_1 and a_2 on each other in vertex set of the tree in Fig. 12. The result is the tree in Fig. 13, Clearly, f is not an automorphism, since $E_T(a_1, c)$ but not $E_T(f(a_1), f(c))$. The new tree, however, although it has a different edge relation, represents the set $\{a, \{a, b\}\}$ equally well. There is an ML *isomorphism* between the vertex sets of the two trees, and f is indeed such an isomorphism. This exemplifies the fact that characterizing structure by means of ML trees is unique only up to isomorphism.

In order to make this work, we need to introduce the restriction that same-labeled vertices also have the same sub-trees where corresponding vertices have the same main label. To be more precise, we need to define the notion of a subtree:

(SubTree) Given a rooted ordered tree T with root v_0, the *subtree* U determined by the vertex $u \in V_T$ is a rooted tree with root u such that $V_U = \{u\} \cup \{v \in V_T : \mathbf{E}_T(v, u)\}$ and such that for all $v_i, v_j \in V_U$: $E_U(v_i, v_j)$ iff $E_T(v_i, v_j)$, and $v_i \prec_U v_j$ iff $v_i \prec_T v_j$.

This allows us to define the notion of an *S-tree*, i.e. a tree fit for capturing structure:

(STree) An *S-tree* T is an ML tree such that for any two distinct vertices $v_i, v_j \in V_T$ it holds that if $L_T(v_i) = L_T(v_j)$

(i) There is an ML isomorphism f between the subtree T_i determined by v_i and the subtree T_j determined by v_j such that for any vertex $v_k \in V_{T_i}$ it also holds that $L(f(v_k)) = L(v_k)$

(ii) There is no vertex $v_k \in V_T$ such that $E_T(v_i, v_k)$ and $E_T(v_j, v_k)$.

We can now state the main definition.

(STRUC) Let A be a complex object, R be a part-whole relation on A, and R^* the transitive closure of R on A. Let A^* be the set of objects standing in R^* to A (the direct and indirect parts of A, including A). Let S be a partial ternary relation on A^* such that for any $a, b, c \in A^*$, if $S(a, b, c)$, then $R(b, a)$ and $R(c, a)$ and not $S(a, c, b)$; that is, S is asymmetric in its second and third arguments. Let $T = (V_T, E_T, \prec_T, L_T, v_0)$ be an S-tree. We say that, *up to isomorphism*, T is *the structure of* A iff there is a surjection r from V_T onto A^* such that

(i) For any $v \in V_T$, $r(v) = A$ iff $v = v_0$

(ii) For any $v_i, v_j \in V_T$, $r(v_i) = r(v_j)$ iff $L(v_i) = L(v_j)$

(iii) For any $v_i, v_j \in V_T$, if $E_T(v_i, v_j)$, then $R(r(v_i), r(v_j))$; and for any $a, b \in A^*$ such that $R(a, b)$, there are $v_i, v_j \in V_T$ such that $r(v_i) = a, r(v_j) = b$ and $E_T(v_i, v_j)$

(iv) For any $v_j, v_k \in V_T$, if $v_i \prec_T v_j$, then there is a vertex $v_i \in V_T$ and a part $a \in A^*$ such that $E_T(v_j, v_i)$, $E_T(v_k, v_i)$, and $S(r(v_i), r(v_j), r(v_k))$; and for any any $a, b, c \in A^*$ such that $S(a, b, c)$, there are vertices $v_i, v_j, v_k \in V_T$ such that $r(v_i) = a, r(v_j) = b, r(v_k) = c$, $E_T(v_j, v_i)$, $E_T(v_k, v_i)$, and $v_i \prec_T v_j$.

The definition makes sense in virtue of the following fact:

Fact 1 Given a tree $T = (V_T, E_T, \prec_T, L_T, v_0)$ that by (STRUC) is the structure of a complex object A, and and a tree $U = (V_U, E_U, \prec_U, L_U, u_0)$, U is the structure of A iff T and U are isomorphic.

The left-to-right and right-to-left parts of this fact are stated separately and proved in Appendix 2.

We can note, finally, that from a given structured object A, we can easily construct the structure itself, i.e. an S-tree that is (up to isomorphism) the structure of A. The exact recipe can be extracted from the definition of the ML-isomorphism. In a slightly simplified form it runs as follows:

(Recipe) For a complex object A, assign a representation of A as the label of the root; given a label 'v_i' of a vertex representing a part a_i of A, for an immediate part of a_i, assign a label 'v_j' to the corresponding daughter of v_i; make sure to use the same main label with a new index for any new occurrence of one and the same part. For a new vertex v_n with the same label as a previous vertex v_m, copy the entire subtree of v_m as the subtree of v_n, assigning the same main label to corresponding vertices. Continue until all occurrences of all parts of A are represented on the tree.

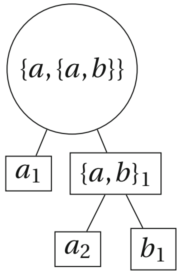


Fig. 14 Applying the recipe to $\{a, \{a, b\}\}$

We can illustrate the recipe as applied again to $\{a, \{a, b\}\}$, in Fig. 14, using standard representations of the sets and elements as labels. Start with assigning '$\{a, \{a, b\}\}$' itself as the label of the root. The two immediate elements are a and $\{a, b\}$, so assign 'a_1' and '$\{a, b\}_1$' as labels to the daughters of the root, without superscripts, since there is no order relation, but with default subscripts: the lowest positive whole number numerals so far not used in the tree. Next, since the second daughter (in order of the presentation) itself is complex with two elements, generate two new daughters of that vertex. Since one is again a, generate as label for the corresponding vertex 'a_2', since the index '1' has already been used. And since the other element is b, choose 'b_1', as the index. Now the leaves of the tree, i.e. the lowermost vertices, are all simple, and we are done.

As the exercise has illustrated, we needed no more information than the specification of the object itself: the relevant part-of relation ($\in$), and the instances of this relation in the object: the immediate elements, their respective immediate elements, and so on, until the simple elements. This recipe does result in an S-tree that is unique up to isomorphism.

4 Structured and unstructured objects

Since the structure of a structured object, as far as it can be represented by S-trees, is unique up to isomorphism, and since nothing more than a specification of the *intrinsic* properties of the object are needed to generate an S-tree that is its structure, we can say the following: there is (given a domain of structured objects) a general function S that maps any structured object o on the *isomorphism class* S_o of trees that are the structure of o.

This highlights a particular property of structured objects: they are *intrinsically structured*. Being structured is not a relational property. A non-empty set is intrinsically structured, in that, by the extensionality of sets, it is part of the conditions of identity of that particular set that it has those particular parts, *and* that the relevant parthood relation of the set is the *element-of* relation.[12] Note that it is not in virtue of a relation

[12] This is not inconsistent with the fact that, with set abstraction, we can *specify* which the elements are in a relational way, which may even require empirical knowledge. For instance the set represented by '$\{x : x$ is a son of George$\}$' is a set of four elements that can also be presented as by '{W, Jeb, Neil, Marvin}'. In the latter case, it can be determined from the representation of the set which the elements are, while background empirical knowledge is needed in the first case. But because of the extensionality of sets, the two

Fig. 15 A little family tree

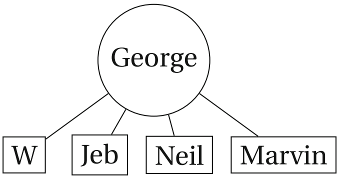
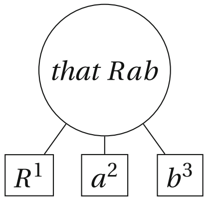


Fig. 16 Perhaps a structured proposition?



to S-trees that an object is structured, but it is in virtue of its intrinsic properties that it stands in a relation to S-trees: it is the *definition* of structure that is given in terms of S-trees.

As will become important later on, if a complex object is specified in part by way of *representational properties* of its parts, i.e. by way of relational properties, this object is not a structured object in the present sense. What the representative part-of relation consists in for that object, is not transparent, or intrinsic to the object. This fact again highlights a crucial property of S-trees (and graphs in general): that the edge relation is *neutral*. This is precisely why the tree can be said to be the structure of a structured object: the same S-tree can be the structure of distinct complex objects, even objects with different part-of relations.

Furthermore, in virtue of this neutrality, an S-tree can represent a partial order on a collection of objects that do *not* form a structured object, since the partial order is not itself a *part-of* relation. To give a vivid example, consider the tree in Fig. 15. Suppose, as might suggest itself, that the labels represent a father and his four sons, respectively. We may thus take the edge relation in Fig. 15 to represent for the *son-of* relation, but there is no reasonable sense in which a son is *part of* his father. The tree faithfully represents a small family, or part of a family, but it does so because of contingent facts.[13] Being the father of W is a relational property of George, and is not determined by George's intrinsic properties.

We are now ready to pose the question: Can propositions* be structured objects? After all, we *can* represent the apparent structure of a proposition* *that Rab* as in Fig. 16. There is, after all, nothing wrong with the tree in Fig. 16. The root does represent a proposition*, the proposition *that Rab*, and the labels of the leaves represent what we easily can think of as the "parts" of this proposition.

Footnote 12 continued
expressions denote the same set. So, the property of sets to be intrinsically structured should not be confused with the property that it can be determined from any correct *specification* (description true uniquely of the object) of the object which the parts are.

[13] Some might want to say (as people did say) that it is an essential property of W to be the son of George, but hardly anyone would claim that it is an essential property of George to be the father of W.

As we have just seen, however, the fact that we can represent a proposition, together with elements from which it is intuitively formed, by means of an S-tree, this does not entail that the proposition* is a structured object. It is a structured object only if the edge relation of the tree represents (or can represent) a part-of relation. However, all we know from the adequacy of the tree in Fig. 16 is that there exists *some* function P from conceptual elements to propositions such that $P(R, a, b)$ is exactly the proposition* *that* Rab. The edge relation of the tree then represents the relation between the arguments, in left-to-right order, and the value, of the P function. This does not show that the value is a structured object with arguments of the function as parts, nor indeed a structured object at all.

In a sense, the situation is even worse, for the value of a function for some particular argument or arguments necessarily *underdetermines* the function and argument(s). The number 5 is the value of any number of functions on various number arguments, such as $2 + 3$, $12 - 7$, and $10/2$. The notation '$f(a)$' gives the impression of having a determination of both function and argument, whereas what is denoted is just the value, which is equally well the value $g(b)$, etc.

The situation cannot be improved upon, on pain of circularity. This was essentially pointed out by Dummett (1973, pp. 293–294), concerning Church's (1951) interpretation of Frege. Dummett writes

> On the model of sense considered in Chapter 7, the sense of a predicate is the criterion for recognizing that the predicate applies to a given object. The thought expressed by the sentence which results from putting a proper name in the argument-place of the predicate is: that the criterion may be recognized to be fulfilled for an object which has been recognized as the bearer of the name. Now the sense of the predicate does indeed determine, for any name whose sense is known, what thought is expressed by the sentence which results from filling the argument-place of the predicate with that name. But the sense of the predicate cannot be thought of as being given by means of the corresponding function, because if we did not already know what the sense of the predicate was, we could not know what was the thought which was the value of the function for the sense of some name as argument (Dummett 1973, p. 293).

Dummett claims that we cannot understand a sentence except by understanding its syntactic constituents and the way they are combined. Against that background, Dummett argues in the quoted passage, the sense of a predicate cannot be a function that maps the sense of a term on the sense of the sentence that is formed from combining the term and the predicate. For, in order to know which function that is, I would need to know what value it gives for term-senses as arguments. But since I need to know that predicate sense first in order to know the sentence sense, i.e. the value of the function, if the sense is the function itself, I need to first know the function in order to know the function. There is thus a circularity, framed in terms of understanding a sentence.

The corresponding circularity comes out directly, without appeal to understanding, if we think of the thought/proposition as structured. For if the sense of a predicate F is a function f that, for the sense of a term t as argument, gives the sense of the sentence

Ft as value, then this sense cannot also be a structured object that contains f as a part, for f would then be defined partly in terms of *itself*.

Of course, we *can* accept a proposition as having parts provided the function that maps the parts on the proposition is not itself a part of the proposition. For instance, letting $[\![\cdot]\!]$ be the *sense* function, assume that there is a function f that maps the sense of F and the sense of t on the sense of Ft. As long as the f is distinct from the sense of F (and from the sense of t), this is not circular. However, in this case, the underdetermination kicks in. For nothing says that $f([\![F]\!], [\![t]\!])$ is not equal to $g([\![G]\!], [\![u]\!])$, for some function g, predicate G and term u, or from the sense of some completely different function applied to much more complicated arguments. With such underdetermination, we cannot infer from the proposition $[\![Ft]\!]$, even if it *has* parts, what the *relevant* conceptual elements are that were mapped on it. Since the function f is not determined by $[\![Ft]\!]$, we cannot rule out that what is relevant for interpretation in some particular case is really g, $[\![G]\!]$ and $[\![u]\!]$ instead.

If we now try to stick f into the proposition as a part, in order fix the function, then again, either f would be defined in terms of itself, and hence the circularity is back, or else we would need a distinct function h that maps $[\![F]\!]$, $[\![t]\!]$, and f, on some complex that has all three arguments as parts, but then h would not be determined by this complex, and hence we are back to underdetermination. We seem to be stuck in a dilemma where underdetermination and circularity are the horns, and both are unacceptable.[14]

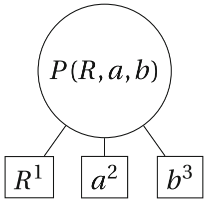
This cannot be the whole story, however, for it seems to rule out structured objects altogether. It seems to rule out $\{a, b\}$, since $\{\cdot\}$ would need to be construed as a function that maps a and b on $\{a, b\}$, and therefore it would already need to be determined what $\{a, b\}$ is for the function to be defined, even though in this case, the function is not itself an element.

There are (at least) two ways of resolving this apparent problem. The first way is the way of axiomatic set theories, like ZFC. On this approach, we already assume a domain of sets over which we can quantify, and over which the relation $\in$ is defined. This domain has certain closure properties. For instance, if a and b are sets, then, according to the Axiom of Pairing, there is a set c such that any element of c is either a or b. c is thus a set that essentially has a and b as its only members. We can regard it as structured. And we can regard $\{\cdot\}$ as a function that maps a and b on c. Alternatively, the notation '$\{\cdot\}$' can be given a contextual definition, which allows its elimination from any true sentence of ZFC.

The second way is to think of $\{\cdot\}$ as a *primitive operation* of collecting objects into sets of objects, an operation that is part of *defining* a domain of objects (like the successor operation in arithmetic) as distinct from functions that are defined (e.g. by recursion) *over* a domain already given. On this view, the set $\{a, b\}$ is *essentially* formed from a and b by $\{\cdot\}$. This operation is not defined by what values it gives to

[14] In fact, this argument can be seen as supporting Russell's argument (Russell 1903, pp. 47–48) that we lose the essential unity of the proposition by analysing it into parts. Short of circularity, there is in general no unique way mapping the parts on a unified proposition.

Fig. 17 Perhaps a structured proposition?



which arguments, but must be understood intensionally, prior to providing the result of an application.

Both these approaches are applicable in the case of propositions. On the first alternative, then, we can assume an already given domain of structured propositions, and a function P that maps conceptual constituents on such propositions. The problem with this approach is that if there is one such function, there are many (P_1, P_2, ...). For any structured proposition p there will be many such functions that give p as value for various conceptual constituents as arguments. A proposition p does not fix any of these functions as *privileged*. In short, there is underdetermination. If the meaning $[\![Ft]\!]$ of the sentence Ft is a structured proposition, we cannot rule out that this structured proposition has $[\![G]\!]$ and $[\![u]\!]$ as parts. Again, you cannot solve it by sticking a function f as an additional part of the proposition, for on pain of circularity, the function that maps the parts on the proposition must be distinct from f.

The other approach is the primitive operation approach. We then imagine a primitive operation P that e.g. takes as arguments R, a, b, in that order, and gives as result the maximally structured proposition $p = P(R, a, b)$. On this approach, P has a privileged status with respect to $P(R, a, b)$, for it is precisely by the application of this operation that the proposition exists in the first place. In virtue of the privileged status of P, there is no underdetermination. We have the structured propositions that is represented in Fig. 17, where the superscripts indicate the *order* between the constituents, which are the arguments to P. The tree T_{17} of Fig. 17 is the structure of $P(R, a, b)$. And the general structure function S is defined for p. $S(p)$ is the isomorphism class of T_{17}.

On the assumption of a primitive proposition-forming operation P, there is no underdetermination. $P(R, a, b)$ essentially is the object formed by P from R, a, and b, in that order. For no other arguments does P give $P(R, a, b)$ as value. Neither is there circularity, for P, on this suggestion, is not defined extensionally, in terms of arguments and values, but intensionally as a primitive operation that we know prior to its application. We may ask what the edge relation in T_{17} corresponds to in this case, i.e. what the relevant part-of relation is, and the answer could be that it is the relation *conceptual-constituent-of*. Maybe we don't have to understand what that more exactly amounts to for accepting it as potentially a reasonable answer.

So, from an ontological point of view, there seems to be room for a category of structured propositions. Let's call them *P-propositions*. The question is whether they can be propositions*.

5 Are P-propositions propositions*?

The defining characteristic of propositions*, from Sect. 2, was that propositions* intrinsically have truth-conditions. And this in turn means that there is a privileged function T that maps propositions on their proper truth-conditions, where T might be identity.

We shall proceed by stipulating some restrictions on and ranking principles over candidates for T. Firstly, we shall require that T be recursively specifiable. Here this will mean two things: that the value of T for a P-proposition p shall depend uniformly on the structure of p, and that if p if has one or more P-propositions $q_1, \ldots, q_n$ as proper parts, the value of T for p depends (uniformly) on the value of T for $q_1, \ldots, q_n$. Here we shall spell out only the atomic case, where the immediate parts of p are a relation and individual concepts. The recursive step for complex P-propositions is not needed for the argument below.

Secondly, we shall require that T be *surjective*: every proposition* is the value of T for some P-proposition.

Thirdly, we shall require that the value of T for a P-proposition p depend only on the constituents of p, i.e. not on concepts that are constituents in *other* P-propositions but not in p.

Fourthly, we shall require that T be as *simple* as possible. In general, simplicity should here be cashed out in terms of the recursive definitions, and it will induce a partial order on the domain of functions from P-propositions to propositions*. In this context, however, we only need to care about the absolutely simplest functions: the identity function Id and the constant functions, i.e. functions which give the same value for all arguments. We can observe immediately, that T cannot be constant function, since that violates the second requirement that T be surjective, given that there are more than one proposition*.

These informally stated requirements motivate the following

(T-Rest) (i) For each n there is a function i_n such that, where p is $P(R^n, a_1, \ldots, a_n)$, $T(p)$ is *that* $R^n(a_{i_n(1)}, \ldots, a_{i_n(n)})$
(ii) If possible, T is Id.

In (T-Rest-i), it is required that for some uniform ordering of the individual arguments or the P-proposition, the value simply is the proposition* formed from relation R^n as applied to the individual arguments of the P-proposition, in that ordering. This is motivated in part from the uniformity requirement: all P-propositions formed from an n-ary relation and n individual concepts are mapped on a proposition*, uniformly depending on the order between the individual arguments to P.

(T-Rest-i) is also motivated in part by the requirement that T be surjective. Since the orders between the arguments to the n-place relation is 1–1 correlated with the orders between the constituents of the corresponding P-propositions, every proposition* formed from an n-ary relation and n arguments, hence every atomic proposition, will be in the range of T.

In addition, (T-Rest-i) is partly motivated by the third requirement that the value of T depend only in the constituents of p, a condition that is clearly met.

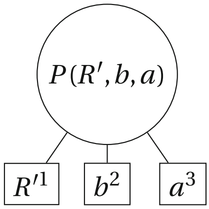


Fig. 18 Perhaps a structured proposition?

Finally, (T-Rest-i) is partly motivated by simplicity: even if T is not identity, no function is involved in defining T for atomic P-propositions other than that of selecting the arguments to P and applying one of them to the rest.

Now for the argument. Its first step is to establish that even the restriction of T to the domain of atomic P-propositions is not identity. Let's first take a quick look at a binary relation. Define a relation R' from a binary relation R to the effect that for any a, b that can be arguments to R, $R'ab$ is true iff Rba is true. Assuming that there are a, b such that Rab is true but Rba false, R and R' are not identical. And the order between a and b matters. Hence, the P-proposition $P(R', b, a)$, represented in Fig. 18, is distinct from $P(R, a, b)$.

Now, suppose that $P(R, a, b)$ is identical to the truth-conditions *that* Rab and that $P(R', b, a)$ is identical to the truth-conditions *that* $R'ba$. By assumption, these truth-conditions are logically/analytically equivalent, and then by the (1) principle, they are identical. Since the two P-propositions are not, they cannot both be identical the corresponding truth-conditions.

Perhaps a way out is to say that $P(R, a, b)$ is identical to the truth-conditions *that* Rab while $P(R', b, a)$ is identical to the truth-conditions *that* $R'ab$. This avoids the contradiction. It is of course completely *ad hoc*. But worse, the move is not even applicable in more complex cases.

So assume we have six ternary relations, $R_1, \ldots, R_6$, defined to the effect that they interlock as is given in (3):

$$(3)\quad R_1(a, b, c) \quad \text{iff} \quad \begin{cases} R_2(a, c, b) \\ R_3(b, a, c) \\ R_4(b, c, a) \\ R_5(c, a, b) \\ R_6(c, b, a) \end{cases}$$

Suppose that the order between the arguments matters in all cases. The relations are all distinct, and supposing that the arguments are as well, we can form 36 P-propositions but only six distinct truth-conditions. If every P-proposition is identical to some truth-condition, it must in most cases be truth-conditions that depend on concepts distinct from these relations and arguments. But that violates (T-Rest-i), as well as the first, third, and fourth informal requirement that motivates it. Hence, already in the domain of proposition formed from ternary relations, T is not Id.[15]

[15] It is possible that a more complicated definition would satisfy the uniformity condition, but then clearly not the third and fourth requirement.

This leaves us with the option of a many-one function T from P-propositions to truth-conditions. For instance, T could map both $P(R_1, a, b, c)$ and $P(R_2, c, a, b)$ on the truth-conditions *that* $R_1(a, b, c)$. The problem now is that there are several equally good candidates for such a function T. For any permutation π on ordered triples, the function $T_\pi = \textit{that } R(a_{\pi(2)}, a_{\pi(3)}, a_{\pi(1)})$ satisfies (T-Rest-i). Hence, assuming that (T-Rest) is both a necessary and sufficient condition, there is more than one optimal function, and hence no uniquely privileged one.

The remaining option [short of rejecting (T-Rest-i)] is to suppose that we could add a further requirement that would distinguish between the T_i:s. We could imagine an additional restriction that would select the identity permutation π_1 above the others, thus making $T_1(P(R, a, b, c)) = \textit{that Rabc}$ the privileged function. But over and above an intuitive naturalness, there is, or seems to be, no mathematical reason for ranking T_1 above the other candidates. It is not simpler in a computational sense and does not satisfy (T-Rest-i) in any higher degree otherwise. Thus, by appeal to the Benacerraf-style argument of Sect. 2, neither T_1, nor any of the other T_i:s is privileged. Hence, given the requirements, there is no privileged function from P-propositions to proposition*.

The upshot is that P-propositions do not satisfy the condition of having intrinsic truth-conditions, (PROP2). Therefore, P-propositions are not propositions*. And therefore, again, propositions* are not structured . Structured objects are too fine-grained to be *identical* to truth-conditions, and as long as they are not, there is in general no privileged way assigning truth-conditions to them.

Acknowledgements I have benefitted from comments at the conference in Barcelona by Manuel García-Carpintero, Marie Duži, Peter Hanks, Bjørn Jespersen, Lorraine Keller, Jonathan Keller, Jeff King, Bryan Pickel, and Scott Soames, as well as from comments by two anonymous referees and from discussions with Kathrin Glüer-Pagin. The research was supported by a grant from The Tercentenary Foundation of the Swedish National Bank (Riksbankens Jubileumsfond) for the project Interpretational Complexity.

Appendix 1: King's proposal of structured propositions

In this appendix, I'll compare the present approach with King's account of structured propositions.[16] King (2007) claims to have solved the problem of providing structured propositions with truth-conditions. It goes as follows. The sentence

(4) Rebecca swims

expresses the proposition *that Rebecca swims*. That proposition is structured: it is a structured *fact* with Rebecca and the property of swimming as components. Graphically, King represents it (2007, p. 38) as in Fig. 19. The left and right hand branches in Fig. 19 are part of a *syntactic* tree (for some language L), corresponding to the syntactic

16 For reasons of space, I shall not do the same for other proposals, but the general idea should be clear.

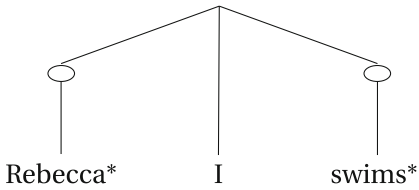


Fig. 19 The structured fact that Rebecca swims

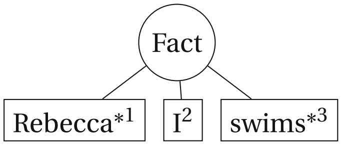


Fig. 20 King's fact represented by an S-tree

construction type: called "the sentential relation R". The ellipses are place-holders for lexical items (of L). The vertical lines from the ellipses represent semantic relations: Rebecca* and swims* are the semantic values. The central vertical line ending with 'I' represents what the syntactic construction type semantically *encodes* (in L), viz. *instantiation*.

Not taking context dependence into account the fact represented is the fact that:

> There are lexical items *a* and *b* of some language L occurring at the left and right terminal nodes (respectively) of the sentential relation R that in L encodes the instantiation function, where the semantic value of *a* is Rebecca and the semantic value of *b* is the property of swimming (King 2007, pp. 37–38)

According to King, this fact *is* the proposition *that Rebecca swims*.

In his 2014, King updates the model. Relativity to assignments has been added. The syntactic relation is existentially quantified: there is *some* relation satisfying the condition. The relation now is *ascription* rather than *instantiation*.[17] There is also a *propositional relation*, the relation that holds between the semantic values of the syntactic leaves in virtue of the syntactic relation between the leaves.

In the revised model, the proposition is the old fact *together with* the propositional relation's having the relational property of encoding ascription (King 2014, p. 52). Call this *having-property* H. On this updated model, the proposition is rather the *pair* $\langle$Fact, $H\rangle$ of the old fact and the new *having*. Since the difference between the old and the revised model is irrelevant to my point, I'll focus on the original model.

In the present framework, we would let King's structured entity be represented by an S-tree, as in Fig. 20: The parthood relation, corresponding the edge relation of the tree, is the relation between the components of the fact and the fact itself: that is, the relation …*is a component of* ….

Now, we first ask whether King's fact as represented in Figs. 19 and 20 are structured objects in our sense. The answer is no, since the Fact is a structured object only in virtue of standing in a relation to some language L that has a sentence with syntactic constituents whose interpretations are the first and third component in the Fact, and a

[17] According to King (pc), there is no significant difference.

syntactic construction whose semantic significance is the second node. Hence, since the current account requires the property of being a structured object to be *intrinsic*, on King's account it isn't, the Fact is *not* a structured object in the present sense.

In the Appendix of King (2007), the account is spelled out slightly differently. Here, the notions of *propositions* and *propositional frame* are introduced, and it is *taken for granted* that propositions and propositional frames have argument places (and hence some internal structure) (King 2007, p. 219). The account specifies components in terms of the kind of the category of the syntactic units that express them, and there is no mention of *facts*.

It is less clear how to read this. If King means it to *remain* the case that being a conceptual element depends on being related to syntactic units of some language, then the same conclusion as above stands. If, on the other hand, we read King's references to syntax as only a means of *specifying* propositional components whose properties are still intrinsic to it, then propositions as spelled out in the appendix *do* qualify as structured objects. As far as I can see, this second alternative is available to King.

Are King's constructions propositions*? In the Appendix of King (2007, p. 221), a kind of truth definition for propositions is provided. The first clause is

(5) A proposition of the form $[\mathrm{R\ o}^1, \ldots, \mathrm{o}^\mathrm{n}]$ is true at w iff $< \mathrm{o}^1, \ldots, \mathrm{o}^\mathrm{n} >$ belongs to $\mathrm{ext}_w(\mathrm{R})$.

The others are similar, all looking like a standard truth definition for a syntactically specified language, and thereby like a stipulation. If indeed it *is* a stipulation, as it seems, then the truth definition is not privileged over alternative possible truth definitions, in which case King's propositions also do not have their truth conditions intrinsically.

The reasonable conclusion therefore is that so far as they have been presented, King's propositions are neither structured objects (judging from the official position in the main text), nor propositions* (judging from the truth definition in the appendix).[18]

Appendix 2: Proof of Fact 1

We state the right-to-left part of Fact 1 as Fact 2 and the left-to-right part as Fact 3.

Fact 2 Given a tree $T = (V_T, E_T, \prec_T, L_T, v_0)$ and a tree $U = (V_U, E_U, \prec_U, L_U, u_0)$, if T by (STRUC) is the structure of a complex object A, and $f : U_T \longrightarrow V_T$ is an isomorphism, then $r \circ f$ satisfies (STRUC) (i)-(iv), which makes U the structure of A.

Proof (i) By (MLIsoi), $f(u_0) = v_0$. Hence $r(f(u_0)) = A$, satisfying (i).

(ii) Let $u_i, u_j \in V_U$, and suppose that $r(f(u_i)) = r(f(u_j))$. By clause (ii) of (STRUC), $L(f(u_i)) = L(f(u_j))$. Since f is an isomorphism, by (MLIsoii) it also holds that $L(u_i) = L(u_j)$, satisfying (ii).

(iii) Let $u_i, u_j \in V_U$, and suppose that $E_U(u_i, u_j)$. Since f is an isomorphism, by (MLIsoii), $E_U(u_i, u_j)$ iff $E_T(f(u_i), f(u_j))$. By (iii), if $E_T(f(u_i), f(u_j))$ then

[18] For further discussion of King's views, see Pickel (2015).

$R(r(f(u_i)), r(f(u_j)))$.
For the second part, assume that for $a, b \in A^*$, $R(a, b)$. By (iii) there are $v_i, v_j \in V_T$ such that $r(v_i) = a$, $r(v_j) = b$ and $E_T(v_i, v_j)$. Since f is an isomorphism, by (MLIsoii) there are $u_k, u_l \in V_U$ such that $f(u_k) = v_i$, $f(u_l) = v_j$ and $E_U(u_k, u_l)$. Since $r(f(u_k)) = a$ and $r(f(u_l)) = b$, the second condition is met. Hence, (iii) is satisfied.

(iv) Analogous to the proof of (iii).

□

For the converse direction, we will need an additional definition, that will be applied to sets of vertices in paths in a tree:

(CLOSE) For any relation R defined on a set M

- (i) $R^0(a, b)$ iff $R(a, b)$
- (ii) $R^{k+1}(a, b)$ iff there is an object c such that $R^k(a, c)$ and $R(c, b)$
- (iii) $R^i_a = \{a\} \cup \{b \in M : R^i(a, b)\}$.

Fact 3 Given two ML trees $T = (V_T, E_T, \prec_T, L_T, v_0)$ and $U = (V_U, E_U, \prec_U, L_U, u_0)$, if by (STRUC) both T and U are the structure of a complex object A, then there is an isomorphism $f\colon V_U \longrightarrow V_T$.

Proof Let r be a surjection from V_T to A^* and s a surjection from V_U to A^*. We define a mapping $f\colon U_T \longrightarrow V_T$ inductively, as follows:

(*) (1) Let $f(u_0) = v_0$.
(2) Assume that $f(u_k) = v_i$ has already been assigned, and further that $E_U(u_l, u_k)$ and $E_T(v_j, v_i)$. Then let $f(u_l) = v_j$ iff $s(u_l) = r(v_j)$.

We show that f is an isomorphism by induction over path length. Let $f \restriction (E^i_u)$ be the restriction of f to the set of vertices E^i_u, i.e. to the set $\{u\} \cup \{u' \in V_u : E^i(u, u')\}$, according to definition (CLOSE). Then we show by induction:

Base step: $f \restriction (E^0_{u_0})$ is an isomorphism. This is trivial, since $\{u \in V_U : E^0(u_0, u)\} = \varnothing$, it holds that $E^0_{u_0} = \{u_0\}$. As regards u_0, (MLIsoi) requires the $f(u_0) = v_0$, which is met by the definition of f.

Induction step. Assume that $f \restriction (E^n_{u_k})$ is an isomorphism, and that $E_U(u_l, u_k)$. We want to show that $f \restriction (E^{n+1}_{u_l})$ is an isomorphism. We take the clauses of (MLIso) in turn.

(i) Satisfied by the base step.

(ii) By (STRUCiii), since $E(u_l, u_k)$, it holds that $R(s(u_l), s(u_k))$. By (STRUCiii) for T, it then holds that there are vertices $v_i, v_j \in V_T$ such that $E_T(v_j, v_i)$, $r(v_i) = s(u_k)$ and $r(v_j) = s(u_l)$. Also, by the induction hypothesis, there is a vertex $v_h \in V_T$, such that $f(u_k) = v_h$. By clause (2) of the construction (*) of f, $r(v_h) = s(u_k)$.
There is yet no guarantee that $v_i = v_h$. If this is true, we are done, so suppose it is not. By clause (ii) of (STRUC), since $r(v_i) = r(v_h)$ it also holds that $L_T(v_i) = L_T(v_h)$. By clause (i) of (STree), it holds that since $E_T(v_j, v_i)$ there is a vertex $v_g \in V_T$ such that $E_T(v_g, v_h)$ and that $L_T(v_g)) = L_T(v_j)$. By clause (ii) of (STree), there is no more than one vertex with this property. By

clause (ii) of (STRUC), $r(v_g) = r(v_j)$. By (2) of the definition (*) of f, since $s(u_l) = r(v_j) = r(v_g)$, $f(u_l) = v_g$. Hence, $E_T(f(u_l), f(u_k))$. By symmetry, the converse holds as well. Hence, clause (ii) of (MLIso) is satisfied.

(iii) Assume in addition that there is a node u_m such that $E_U(u_m, u_k)$ and that $u_l \prec_U u_m$. By the reasoning in (ii), we have $s(u_m) = r(f(u_m))$, as well as $E_T(f(u_m), f(u_k))$.

By clause (iv) of (STRUC), there is a vertex $u_t \in V_U$ and part $a \in A^*$ such that $E_U(u_l, u_t)$, $E_U(u_m, u_t)$, $s(u_t) = a$, and $S(a, s(u_l), S(u_m))$. Since $E_U(u_l, u_k)$, it follows that $u_t = u_k$, and hence that $S(s(u_k), s(u_l), S(u_m))$.

By (STRUCiii), $R(s(u_l), s(u_k))$ and $R(s(u_m), s_{u,k})$. Since T is the structure of A, again by (STRUCiv), there are vertices $v_h, v_i, v_j \in V_T$ such that $r(v_h) = s(u_k), r(v_i) = s(u_l)$, $r(v_j) = s(u_m)$, $E_T(v_i, v_h)$, $E_T(v_j, v_h)$, and $v_i \prec_T v_j$. If $f(u_k) = v_h$, $f(u_l) = v_i$ and $f(u_m) = v_j$, then $f(u_l) \prec_T f(u_m)$, and we are done. So we don't suppose that this is true.

Suppose instead that there are possibly distinct vertices $v_p, v_q, v_r \in V_T$ such that $f(u_k) = v_p$, $f(u_l) = v_q$ and $f(u_m) = v_r$. By the induction hypothesis, $r(f(u_k)) = s(u_k)$, and by clause (2) of (*), $r(f(u_l)) = s(u_l)$ and $r(f(u_m)) = s(u_m)$, and hence $r(v_p) = r(v_h)$, $r(v_q) = r(v_i)$ as well as $r(v_r) = r(v_j)$. By clause (ii) of (STRUC), $L(v_p) = L(v_h)$, $L(v_q) = L(v_i)$, and $L(v_r) = L(v_j)$.

Then, by clause (i) of (STree), there is an isomorphism g between the subtrees T_h and T_p of T determined by v_h and v_p respectively such that for any $v_x \in T_H$, $L(v_x) = L(g(v_x))$. Since, by what is said above we know that $E_T(v_i, v_h)$, $E_T(v_j, v_h)$, and $v_i \prec_T v_j$, we can infer that $E_T(g(v_i), v_p)$, $E_T(g(v_j), v_p)$, and $g(v_i) \prec_T g(v_j)$. It also holds that $L(v_i) = L(g(v_i))$ and $L(v_j) = L(g(v_j))$.

Since we have inferred above that $L(v_q) = L(v_i)$, and $L(v_r) = L(v_j)$, it follows that $L(v_q) = L(g(v_i))$, and $L(v_r) = L(g(v_j))$. Since $E_T(g(v_i), v_p)$ and $E_T(f(u_l), f(u_k))$, which means that $E_T(v_q, v_p)$, by clause (ii) of (STree), $v_i = v_q$. Analogously, $v_j = v_r$. Hence, $v_q \prec_T v_r$. That is, $f(u_l) \prec_T f(u_m)$. By symmetry, the converse holds as well. Hence, clause (iii) of (MLIso) is satisfied.

(iv) That this condition is met follows almost immediately from clause (ii) of (STRUC).

Hence, f is an isomorphism. □

References

Bealer, G. (1993). A solution to frege's puzzle. *Philosophical Perspectives*, *7*, 17–60.

Benacerraf, P. (1965). What numbers could not be. *Philosophical Review*, *74*, 47–73.

Chartrand, G., & Zhang, P. (2012). *A first course in graph theory*. Mineola, New York: Dover Publications.

Church, A. (1951). A formulation of the logic of sense and denotation. In P. Henle, H. M. Kallen, & S. K. Langer (Eds.), *Structure, method, and meaning: Essays in honor of Henry M. Scheffer* (pp. 3–24). New York: Liberal Arts Press.

Cresswell, M. J. (1985). *Structured meanings. the semantics of propositional attitudes*. Cambridge, Mass.: MIT Press.

Cresswell, M. J. (2002). *Why propositions have no structure. Nous*, *36*, 643–662.

Dummett, M. (1973). *Frege: Philosophy of language*. London: Duckworth.

Duzi, M., Jespersen, B., & Materna, P. (2010). *Procedural semantics for hyperintensional logic: Foundations and applications of transparent intensional logic*. Dordrecht: Springer.
Hanks, P. (2011). Structured propositions as types. *Mind*, *120*, 11–52.
Hanks, P. (2015). *Propositional content*. Oxford: Oxford University Press.
Jespersen, B. (2003). Why the tuple theory of structured propositions isn't a theory of structured propositions. *Philosophia*, *31*, 171–183.
Jespersen, B. (2012). Recent work on structured meaning and propositional unity. *Philosophy Compass*, *7*, 620–630.
King, J. C. (2007). *The nature and structure of content*. Oxford: OxfordUniversity Press.
King, J. C. (2009). Questions of unity. In *Proceedings of the Aristotelian Society CIX* (pp. 257–277).
King, J. C. (2011). Structured propositions. In E. N. Zalta (Ed.), *Stanford encyclopedia of philosophy*. http://plato.stanford.edu/archives/fall2011/entries/propositions-structured/%3E.
King, J. C. (2014a). Naturalized propositions. In King, Soames, and Speaks 2014, chap. 4.
King, J. C. (2014b). Structured propositions. In E. N. Zalta (Ed.), *Stanford encyclopedia of philosophy*. http://plato.stanford.edu/archives/fall2011/entries/propositions-structured/.
King, J. C., Soames, S., & Speaks, J. (2014). *New thinking about propositions*. Oxford: Oxford University Press. Electronic resource at SUB.
Lewis, D. (1970). General semantics. Synthèse 22:18–67. *Reprinted in Lewis*, *1983*, 189–232.
Lewis, D. (1983). *Philosophical papers* (Vol. i). Oxford: Oxford University Press.
MacFarlane, J. (2014). *Assessment sensitivity: Relative truth and its applications*. Oxford: Oxford University Press.
Pagin, P., & Pelletier, F. J. (2007). Content, context and composition. In G. Peter & G. Preyer (Eds.), *Context-sensitivity and semantic minimalism. New Essays on Semantics and Pragmatics* (pp. 25–62). Oxford: Oxford University Press.
Pagin, P., & Westerståhl, D. (2010). Pure quotation and general compositionality. *Linguistics and Philosophy*, *33*, 381–415.
Pickel, B. (2015). Are propositions essentially representational? *Pacific Philosophical Quarterly*. doi:10.1111/papq.12123.
Russell, B. (1903). *Principles of mathematics* (2nd ed.). Routledge. Available at the Internet Archive.
Soames, S. (2010). *What is meaning?*. Princeton, NJ: Princeton University Press.
Soames, S. (2014). Cognitive propositions. In J. C. King, S. Soames, & J. Speaks, chap 6.