The Underdetermination of Typings

Jan Westerhoff

19th July 2004

[*Erkenntnis* 2003, 58:3, 379–414]

Abstract

This paper argues that there is no possible structural way of drawing a distinction between objects of different types, such as individuals and properties of different adicities and orders. We show first that purely *combinatorial* information (information about how objects combine to form states of affairs) is not sufficient for doing this. We show that for any set of such combinatorial data there is always more than one way of typing them — that is, there are always several ways of classifying the different constituents of states of affairs as individuals and properties. Therefore, contrary to received ontological opinion, no object is essentially of any specific type. In the second part we argue that taking into account *logical* information does not help either, since logic presupposes the very distinction we are trying to draw. Furthermore, this distinction is not even essential for logic, since logic can function perfectly well without it. We conclude that certain distinctions which have been traditionally regarded as ontologically basic (such as that between individuals and properties) cannot be as fundamental as is often supposed.

1 Typings and their transformations

1.1 The problem

Let us look at an uncontroversial example of objects belonging to the different types of individuals, properties, relations and so on. I take it that it is as uncontroversial as anything can be in ontology that we consider a particular like the Tower of London as belonging to the type of *individuals*, something like 'being red' as a *first order (monadic) property* of individuals and something like 'being a colour' as a *second order (monadic) property of first order monadic properties.* We do this because of the way these three

objects can go together to form states of affairs. The individual and the first order property can go together to make up a state of affairs (the one picked out by the sentence 'The Tower is red'), as can the first order property and the second order property ('Red is a colour'). However, the individual and the second order property cannot go together to form a state of affairs. We assume that these facts are encoded in our typing the three objects in the way we do, i.e. that these combinatorial possibilities are expressed by our calling one an individual, another a first order monadic property and so on.

But this is not the only way in which we can type the objects. Suppose we consider 'being red' as an individual (let us call it 'Red' for that purpose) and *both* the Tower and 'being a colour' as first order properties (let us call them 'to tower' and 'to colour'). In this case the objects fit together to produce the very same states of affairs as those above. The state of affairs corresponding to the one denoted by 'The Tower is red' (and which is now denoted by 'Red towers') consists still of an individual and a first order (monadic) property, although their rôles are reversed.[1] There is also an equivalent to the state of affairs denoted by 'Red is a colour' (now 'Red colours'), but now this has exactly the same form as the previous one, rather than being made up of a first order and a second order property. Equally there could not be a state of affairs consisting of 'to tower' and 'to colour' since things forming a state of affairs cannot be of the same order.

There are still further ways of typing the above objects. For example, we could consider both the Tower and 'being a colour' as individuals (and call them 'Tower' and 'Colour') and once again take 'being red' as a first order property. Or we could 'lift' the two last examples of typings by moving everything to the next higher type; individuals would become first order properties, first order properties second order properties. Finally, we could reverse our initial typing which made use of three levels, so that now 'being a colour' is a individual ('Colour'), 'being red' a first order property, and the Tower a second order property. As the reader is invited to check, all these typings allow for the same states of affairs to be put together and are thus equally acceptable.

1.2 Data and conventions

Before we proceed any further, however, we need to be somewhat more explicit about how we can be so sure that the above typings really all are alternatives in that they make the same states of affairs possible.

Note first of all that there are such things as *ontological data*. These are information about what can form states of affairs with what; a particular

[1] From a purely linguistic point of view it might be interesting to note that there are languages which treat their adjectives in the way we treat nouns, and others which treat them like verbs. Translations of the sentence 'The Tower is red' would therefore become something like 'Tower reds' or 'Red towers'. See Baker (2001, 175–176).

example of such a datum is that 'being red' and the Tower go together in the state of affairs denoted by the sentence 'The Tower is red'. To be a bit more concise let us write t, r and c for 'Tower', 'being red' and 'being a colour', independent of the ontological type assigned to them in a particular typing. We shall call the collection B of these the *basis* of our data. Let us express the fact that some of them go together to form a state of affairs by prefixing the multisets containing them with a $+$, else with a $-$.[2] Thus the ontological data D we have about t, r and c is that $+[t,r]$, $+[c,r]$ and $-[t,c]$, $-[t,t]$, $-[r,r]$, $-[c,c]$. These data act as a constraint on the typing we want to construct: they are what the typing is to be a theory of.

There is also another kind of constraint involved, which I will call the *type-form conventions*. The conventions which were implicitly in play in the above examples entailed that types are indexed by an ascending, uninterrupted sequence of ordinal numbers beginning with zero, and that members of these types can go together into states of affairs only if they are all located in two directly neighbouring types. This is just the picture of types found in the Russellian simple theory of types. In fact it is presumably more restrictive than we want our theory of types to be. For example, this theory will have problems accounting for objects which are applicable to other objects at more than two levels ('is a property' is applicable to first order properties, to second order properties and so forth). There are also difficulties with relations which take arguments from different levels ('instantiates' takes one object from level n and one from a directly adjacent level). But of course all we shall have to say below will apply equally to *less* restrictive type-form conventions.

The task of constructing a typing for a particular set of objects means finding a way of remaining faithful to the ontological data within the type-form conventions in play. Of course it can happen that a certain set of ontological data cannot be typed at all given a set of type-form conventions. For example, if your data are $+[a,b]$, $+[b,c]$ and $+[c,a]$, there is clearly no way of typing this in the stratified way just suggested. If we put a at one level, b must go into the level directly above or below. c must then go directly above or below *that* level, so that in every case the resulting typing cannot remain faithful to all the ontological data. But of course if such a thing happens, this just shows that our type-form conventions (the structural constraints on our type theory) are unable to deal with the data

[2] Multisets are just like ordinary sets apart from the fact that they allow for repetitions. Thus while $\{a,b\} = \{a,a,b\}$, $[a,b] \neq [a,a,b]$. They are distinguished from ordered sets in that the order of elements does not matter, i.e. while $< a,a,b > \neq < a,b,a >$, $[a,a,b] = [a,b,a]$. We assume that each multiset of cardinality one is automatically prefixed with a $-$. (Nothing can form a state of affairs on its own. This, however, does not yet exclude the possibility that it can form a state of affairs with itself, i.e. although for all a, $-[a]$, $+[a,a]$ might be the case.) We will use $\phi, \psi, \ldots$ as variables ranging over multisets. See Blizard (1989) for the formal background of multiset theory.

given. Since the data cannot be captured in the framework, we had better choose a different framework.

Of course this was not the case with the examples discussed above. There it was straightforward to check that all of the above typings were in accordance with the ontological data. But we have also seen that there can be more than one way of achieving such accordance. For example, if our data demand that two objects cannot go together in a state of affairs (e.g. t and c above) we have different ways of accounting for this: we can put them 'types apart' or, to achieve the same effect, put them in the same type. The important point is that this can happen even if we have agreed on one system of type-form conventions. It is clear that any such system is purely conventional (hence the name). We could have said that elements from the same type can go together, or that only members of types with the same parity can go together, or any other such convention. But even settling for a framework in which to give an account of our ontological data radically underdetermines the form of the resulting typing. Above we have given five substantially different alternative typings for a set of data on the basis of one set of type-form conventions. In some there are three levels of types, in others only two; in some there are individuals, in others there are not; and in particular each element could turn up at any level, it could be an individual, a first order property or a second order property.

Of course the data restrict the number of possible typings relative to some type-form conventions to some extent. For example, typings where t was an individual, c a first order property and r a second order property, or where t and r are individuals and c a first order property, cannot faithfully represent the data since on the above type-form conventions they would entail that $+[t, c]$ and $-[r, t]$. However, the restriction provided by the data is not restrictive enough to guarantee uniqueness.

1.3 A graph-theoretic perspective

We can represent ontological data and their typings visually by invoking some simple graph theory. Take the members of the basis B to be the vertices and say that whenever we have an entry of the form $+[s, t]$ in D, the vertices s and t are linked by an edge. We then introduce an extra non-graph-theoretic construction to represent a *typing* of a graph: a typing of a graph is a function which assigns a natural number to every vertex in B in such a way that vertices s, t are mapped to numbers i, j such that $|i - j| = 1$ iff s and t are linked by an edge.

Intuitively, what a typing of a graph does is to divide it into slices numbered $0, 1, 2, \ldots$ in such a way that vertices connected by an edge are assigned to directly neighbouring slices. Diagram 1 shows what the intuitive way of typing the graph described by the data considered earlier looks like.

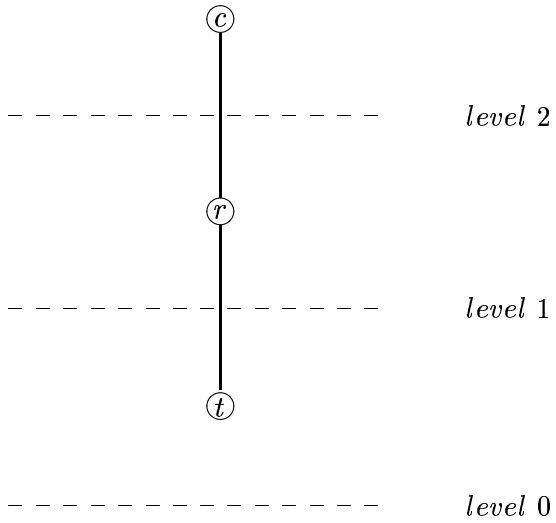

Diagram 1: Typing 1

It is now easy to see that the alternative typings discussed above were achieved by three kinds of transformations of the typings of the graphs. These are:

> LIFTING, which increases the ordinal assigned to each type and thus lifts all types one level up.
>
> MIRRORING, which takes some vertex a (which is not connected to any object at a higher level) at level n and moves all the vertices which are connected to a by a path at lower levels to higher levels: those at $n-1$ go to $n+1$, those at $n-2$ to $n+2$ and so on (this is *upwards* mirroring); or, alternatively, it takes some vertex a (which is not connected to any object at a lower level) at level n and moves all the vertices at higher levels to lower levels: those at $n+1$ go to $n-1$, those at $n+2$ to $n-2$ and so on (*downwards* mirroring). Upwards mirroring can be applied to all typings, downwards mirroring only if there is 'enough space' below: if a is at level n and the highest vertex in the path is at level m, we require that $(m-n) \leq n$.
>
> FOLDING, which 'folds' the graph downwards (upwards) at some vertex located at level n so that some or all of the vertices which are connected by an upward (downward) path to that vertex and which are m levels above (below) it are m levels below (above) it in the transformation.

Applying LIFTING to typing 1 once results in the following:

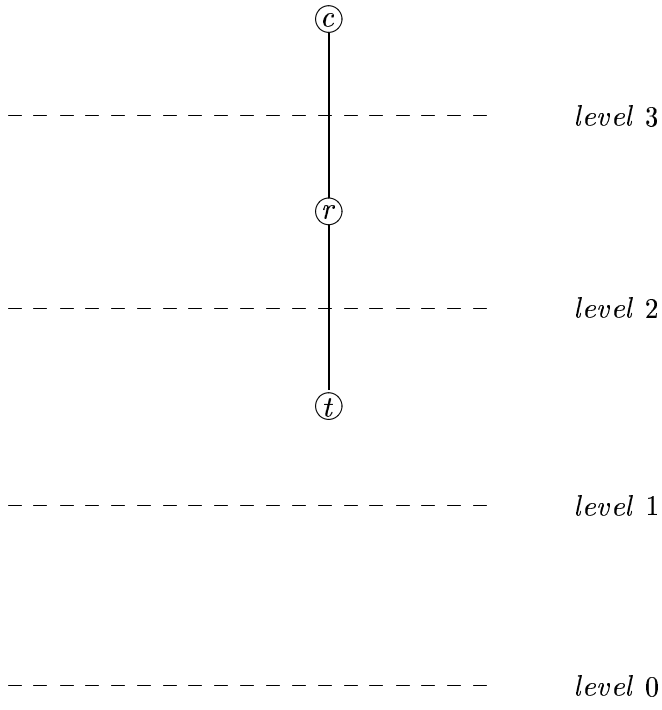

Diagram 2: LIFTING applied to typing 1

LIFTING typing 1 up two levels and then MIRRORING it downwards reverses it:

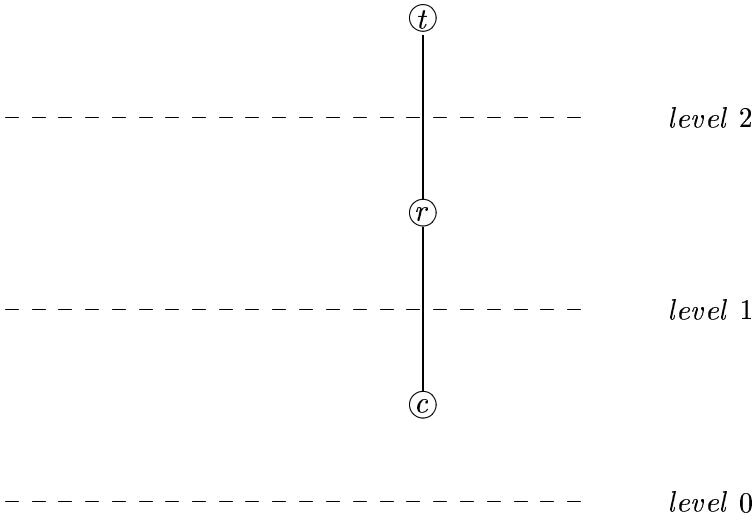

Diagram 3: LIFTING and MIRRORING applied to typing 1

And finally FOLDING downwards at r gives us

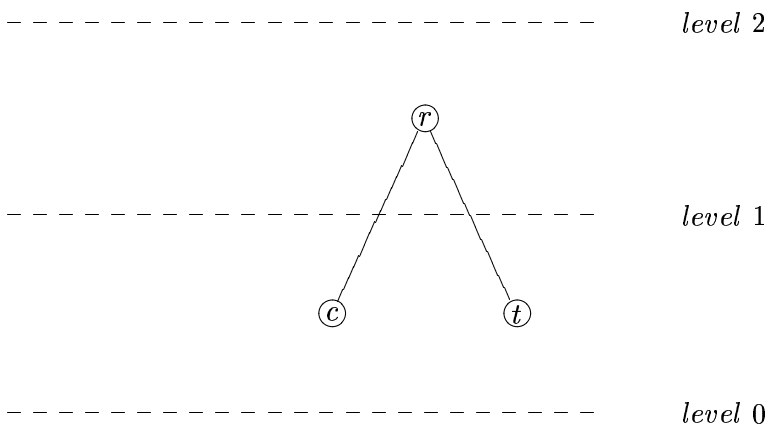

Diagram 4: FOLDING applied to typing 1

We get a general *applicability result* for these transformations of the following form: *If a graph can by typed in a stratified way according to the above conventions at all, all applications of* LIFTING, MIRRORING *and* FOLDING *preserve accordance with the data.*

This is immediately obvious in the case of LIFTING and MIRRORING. To see that it holds for FOLDING as well consider that if we fold downwards (or upwards) at some level n, all vertices at level $n+1$ (or $n-1$) go to $n-1$ (or $n+1$), those at $n+2$ (or $n-2$) go to $n-2$ (or $n+2$) and so forth, while the edges between them are preserved. FOLDING thus keeps each vertex always in the company of its direct level-neighbours.

1.4 Ontological import of transformations

It is debatable to what extent all three kinds of transformations result in genuinely different typings. LIFTING seems to be the least problematic in this respect. Even though we might say that a typing which dispensed with individuals altogether by regarding all (former) individuals as first order properties, all (former) first order properties as second order properties and so on presents a genuine alternative to the typing with individuals, iterating this lifting further does not make much of a difference, in particular since all the levels below the level at which the first items occur are empty. It does not seem sensible to say that the crucial difference between two typings is that in the one the items of the lowest level are at level 26 while in the other they are at level 27 and that this has any ontological import. We would rather be tempted to say, for instance, that the individuals in a typing are *whatever* inhabits the lowest level, in which case LIFTING would not constitute an ontologically relevant transformation at all.

This is essentially what Russell claims in the *Principia*:

> In practice, we never need to know the absolute types of our variables, but only their *relative* types. That is to say, if we prove any proposition on the assumption that one of our variables is an individual, and another is a function of order n, the proof will still hold if, in place of an individual, we take a function of order m, and in place of our function of order n we take a function of order $n+m$, with corresponding changes for any other variables that may be involved.[3]

MIRRORING is more problematic since it allows us (together with LIFTING) to reverse a typing, making individuals of a typing into properties of the highest order, and vice versa, as well as reversing all the levels in between. Gaskin, however, considers reversing to be ontologically harmless:

> Let us try to imagine a world-view which systematically reverses the order in the Fregean hierarchy of his zeroth- and first-level expressions, by systematically reallocating the expressions which he locates in the zeroth level to the second level, and then renumbering the levels accordingly. It is at least clear that that under this transformation the valencies of different types of expressions would not be affected. Proper names such as 'Socrates', now located at level 1, would still be constructed with predicates like 'wise', now located at level 0, and their corresponding referents would still engage with one another in the appropriate way, though now with reversed ontological allegiance: that is, the erstwhile basic particulars would constitute the new first level universals, and the erstwhile first level universals the new basic particulars. [...] Metaphysically speaking, the two purportedly different ways of looking at things would surely just be doublets of one another. In other words, there would be no absolute difference between the rival hierarchies: the only absolute difference would be the *difference of level* (i.e. the different valencies of expression or object) respected by both hierarchies. [...] The one view says that the particular Socrates instantiates (among other universals) the universal wisdom, the other that the particular wisdom instantiates (among other universals) the universal Socrates. But the two views are surely just using different notations to calibrate the same facts.[4]

Note, furthermore, that we can only reverse typings got from finite sets of data and those got from infinite sets which can be typed using only a finite

[3] Whitehead and Russell (1925, I:165), see also 161–162.
[4] Gaskin (1998, 29–30).

number of types. A hierarchy which is infinite in one direction obviously cannot be reversed in the above way. Nor could we reverse a finite but potentially infinite hierarchy which is considered to be always extensible by adding elements at the level above the highest one. If we assume that the hierarchy has a largest element but that we can also extend it into one in which the largest element is greater by one (i.e. by adding higher order properties), then this gives us a substitute for directionality in the finite case. We cannot reverse the hierarchy since the 'real' top of the hierarchy is the part which can always be extended.

FOLDING seems to be the most problematic transformation. After all, it implies that elements of levels with the same parity can suddenly be at the same level, or that in certain cases elements at level n can be moved to level $n+2$ or $n-2$. FOLDING goes even further than the transformations Gaskin is willing to countenance since for him the '*differences* of level in the hierarchy [...] are absolute'.[5] If we accept FOLDING, this will no longer be true. It is unclear not only which objects are located at which level, but also how many levels there are (in the above example applying FOLDING could reduce a typing with three levels to one with only two).

The three transformations together give us the following *flexibility result*: *A typing of the graph according to the above conventions where some object* a *is at level* n *can by a successive applications of the three transformations be made into another typing which is also in accordance with the data but where* a *is located at level* m*, for any* m. (See the appendix for the proof.)

Thus by applying the above transformations to a typing of some ontological data according to the above type-form conventions, any particular object can be put in any type we choose, provided we fudge around with the other assignments in the required way. Clearly that does not mean that this can be done for *all* objects and thus that all typings are adequate. Rather, if we put some object in some type first, faithfulness to the data will imply that we are restricted in where to put the other objects. Nevertheless, for each *particular* object we can put it into whatever type we like.

1.5 Expansions: adding more of the same

Of course it is important to find out whether the above results are only produced in situations with three elements or whether they can be generalized. Let us look at one with four elements c, s, t and r which are supposed to be the type-neutral equivalents of a cup of tea, 'being 63°C', 'being a temperature between 30°C and 70°C', and 'being a temperature-range'. We would intuitively regard these four objects as an individual, a first order monadic property, a second order monadic property and a third order monadic property respectively. Our D will contain $+[cs]$, $+[ts]$, $+[rt]$, while all other

[5] Gaskin (1998, 27), see also 30–31.

pairwise combinations of members of the basis will be prefixed by $-$. The intuitive picture is thus the following:

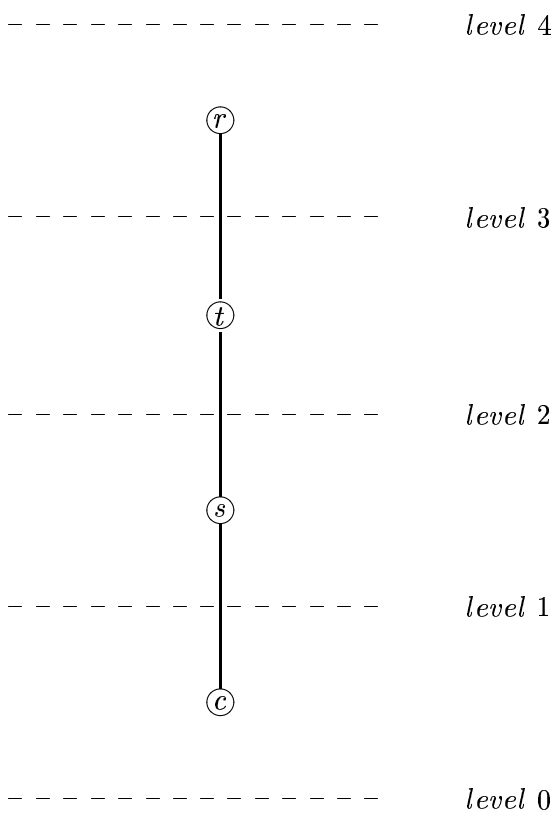

Diagram 5: Typing 2

If we consider a typing of D in accordance with the above type-form conventions it is clear that LIFTING will preserve accordance with the data. We can also reverse the typing by regarding temperature-ranges as individuals, 'being a temperature between 30°C and 70°C' as a first order property of such ranges, 'being 63°C' as a second order property of this, and a cup of tea as a third order property of temperatures. This would then just be the reverse of the 'intuitive' typing 2 suggested above and would look like this:

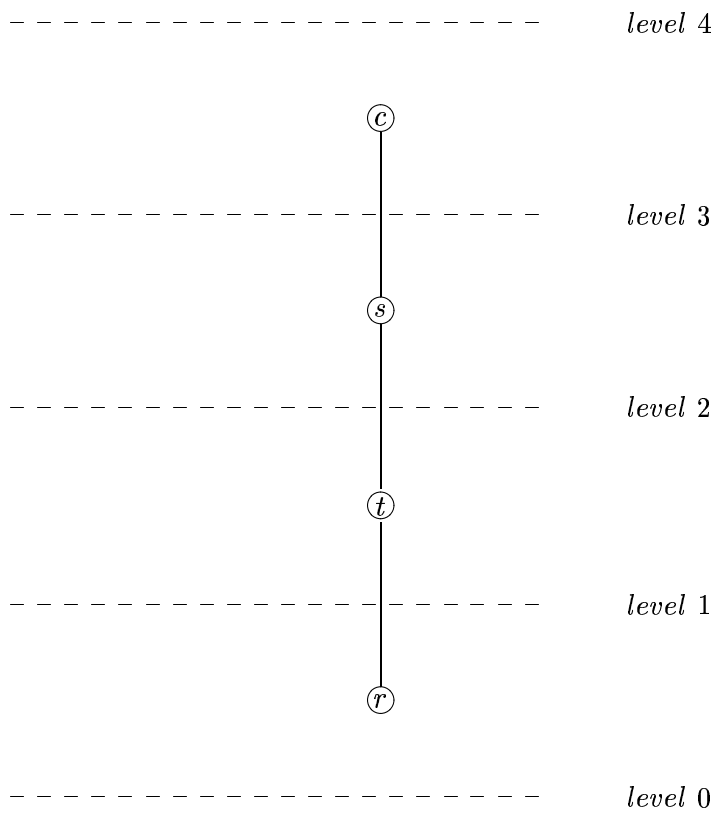

Diagram 6: LIFTING and MIRRORING of typing 2

Similarly we could apply FOLDING to let the temperature of 63°C come out as a first order property, a cup of tea and 'being a temperature between 30°C and 70°C' as second order properties of this, and 'being a temperature-range' as a third order property of the latter.

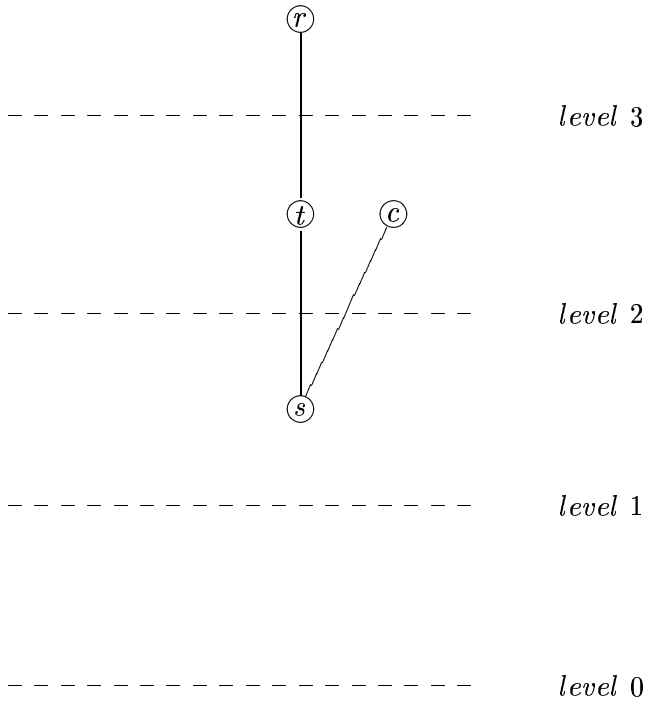

Diagram 7: A FOLDING of typing 2

It thus turns out that the typing remains just as underdetermined even if we consider worlds with more than three elements. Nevertheless, we note that in the above case we encounter a phenomenon we did not get in the case with three elements: now there can be elements at level n such that they cannot go together with both elements at level $n+1$ or with elements at level $n-1$, *although there are elements at both levels.* In the above case after FOLDING a cup of tea could go together with the object directly below it (the temperature of 63°C) but not with the only thing on the level above it ('being a temperature-range'). While this might seem odd, it is not ruled out by the type-form conventions given above, for these only specified that no two elements on the same level could go together and that only elements at two directly adjacent levels could go together. This does not imply that elements must be able to go together with all other elements at adjacent levels. In fact we would not want our type-form conventions to demand this. We suppose that these give necessary but not sufficient conditions for 'fitting': two objects 'fit' *only if* they are obeyed, but not *whenever* they are obeyed. If we want our theory of types to be able to cope with individuals of different kinds, for example, we would not want to say that e.g. 'is divisible by 3' fits together with 'the moon', although the first is a monadic first order property and the second an individual.

Given that the above result generalizes in this way it is then apparent that, for any set of data which consists only of pairs of elements (as is the case for the above data) and which can be typed given the above type-form conventions, all three transformations preserve accordance with the data. This would then imply that in a world where all elements are monadic, nothing is essentially of any specific type.[6]

1.6 Expansions: the polyadic case

The next thing to consider is what happens in the polyadic case, where the lists marked by + or − can contain more than two elements. Consider the fact denoted by 'Peter is married to Clare' and abbreviate its constituents by p, c and m. Our ontological data D will be as follows: $-[pc]$, $-[pm]$, $-[cm]$, $+[mpc]$, $-[mpp]$, $-[mcc]$, $-[mmp]$, $-[mmc]$, $-[mmm]$, $-[ppp]$, and $-[ccc]$. Intuitively we will type p and c as individuals and m as a first order dyadic property. Now suppose we reverse this typing and let m be an individual ('MarriedTo') and p and c first order dyadic properties ('peters', 'clares'). Then the fact that Peter is married to Clare ('MarriedTo peters clares') consists of one individual and two first order dyadic properties. Note that the property 'peters' differs from 'married to' in that, although both are dyadic and first order, 'married to' requires two individuals to form a state of affairs, while 'peters' requires one individual and one first order dyadic property (in our case this is 'clares'). Again this might appear odd, but it is not in conflict with the type-form conventions. They do not imply that no two elements of the same type can form part of a state of affairs (otherwise even the 'intuitive' typing of the above D would not be correct since 'Peter' and 'Clare' are both of the same type). They only specify that no collection *purely from one type* could go together.

Clearly this reverse of the original typing is in accordance with the data. It is also possible to extend our graph-theoretic account to deal with elements of D with more than two members. We continue to treat elements with two members in the usual way. For each $+[s_1, \ldots, s_n]$, with $n > 2$ we connect the vertices in such a way that each s_n is connected with exactly two others s_j, s_k. We call the resulting structure a *cluster*. To distinguish clusters from the graphs describe above we will draw the edges in clusters as lines of a different kind. The above example would thus look like this:

[6] See Denyer (1998, 623).

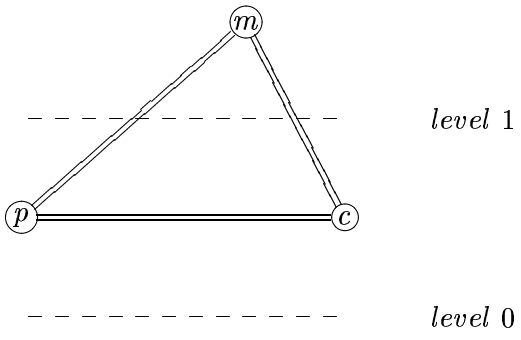

Diagram 8: Typing 3

Given that our type-form conventions stipulate that only objects at directly adjacent types can go together, every cluster will be wholly located within two distinct levels. We then note that clusters behave relative to LIFTING and MIRRORING exactly as the usual edges do. For example, we can reverse typing 3 by applying LIFTING and MIRRORING to get the following typing:

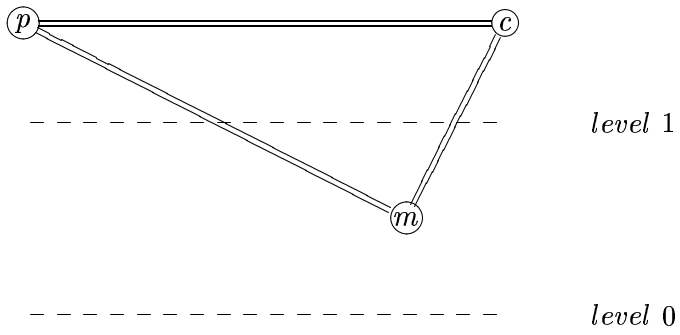

Diagram 9: LIFTING and MIRRORING of typing 3

Clusters can be FOLDED in a similar way to ordinary typings. To see how this works consider a slightly more complex example. Consider a world with five elements c, g, s, t and f, which are supposed to be the type-neutral equivalents of 'being the converse of', 'being greater than', 'being smaller than', the number three and the number five. Our D will contain $+[c, g, s]$, $+[g, t, f]$, $+[s, t, f]$, while all other pairwise combinations of members of the basis will be prefixed by $-$. Intuitively of course the numbers three and five are taken to be individuals, 'being smaller than' and 'being greater than' dyadic properties of these, and 'being the converse of' a dyadic property of

properties. The typing representing this is thus the following:[7]

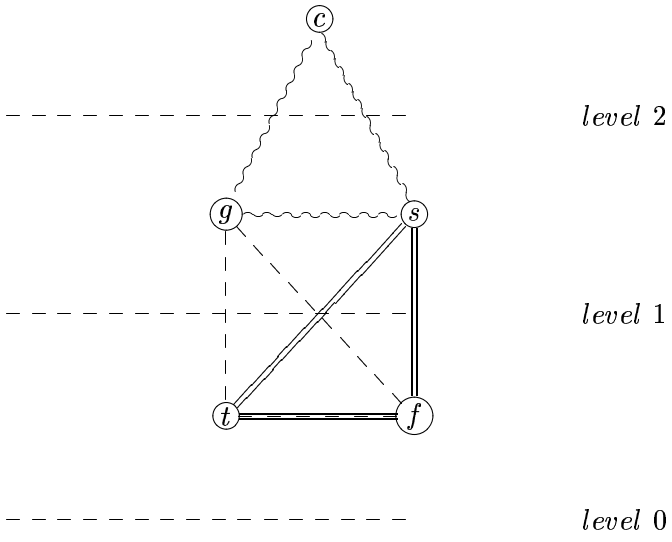

Diagram 10: Typing 4

FOLDING this downwards along the vertices g and s gives us

[7]We use different styles of edges to tell apart the three different clusters involved.

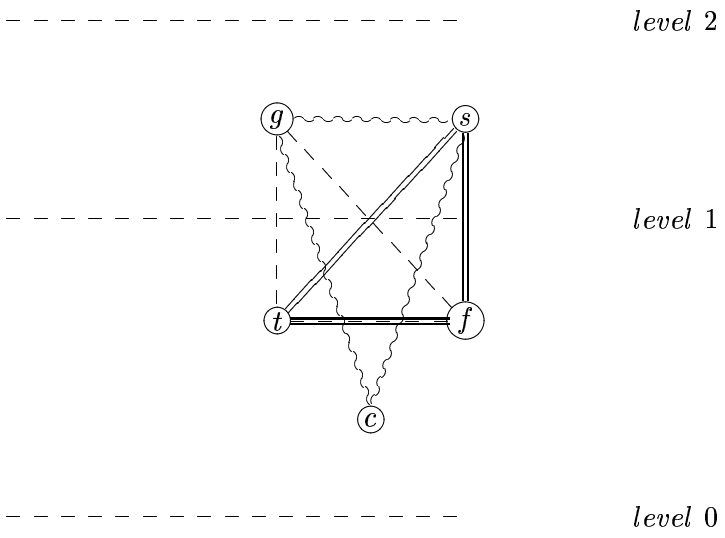

Diagram 11: FOLDING typing 4

It is then easy to check that the above application and flexibility results generalize to the polyadic case. Neither adding more elements nor the introduction of polyadic elements makes the underdetermination of typings by D disappear. We can always apply the above transformations to some typing to end up with something which is also in accordance with the data, but which takes the elements to belong to quite different types from the original typing.

1.7 Tightening the type-form conventions?

Let us now look at two ways of ruling out some of the transformations on typings so as to avoid the flexibility result. Both rely on tightening the type-form conventions to ensure that in the case of certain transformations the transformed typing is no longer in harmony with the conventions. We will look at two ways of enforcing an asymmetry in the relation between the different levels within a typing.

1.8 Rigidity

Suppose we enlarge the above example involving Peter, Claire and 'married to' by adding the monadic property 'sleeps'. We thus enlarge our data by $+[s, c]$ and $+[s, p]$ (while all other combinations containing s will be prefixed by $-$). The intuitive typing of the result will look like the one given

in diagram 12. If we then reverse typing 4 we end up with two individuals ('MarriedTo' and 'Sleeps') and two first order properties, 'peters' and 'clares', as in diagram 13.

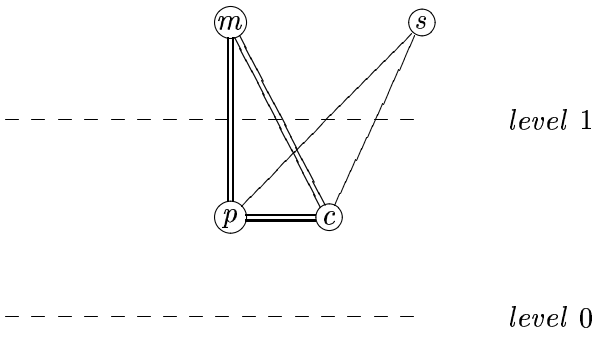

Diagram 12: Typing 5

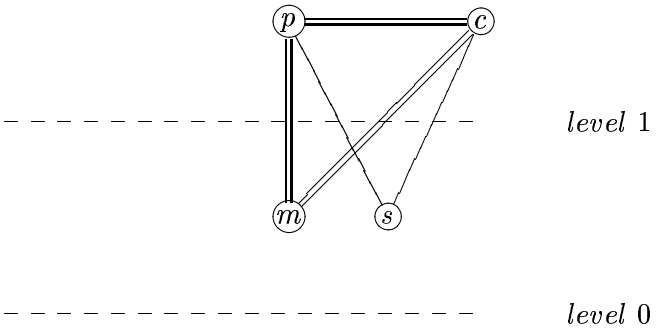

Diagram 13: Reverse of typing 5

It might now be argued that applying the transformation in this case shows that we have destroyed a property which we intuitively expect of typings. We usually think that objects can go together with a *fixed number* of other objects at the level directly below (a predicate either takes one name, or two, or three, etc.). while the cardinality of sets of objects they can go together with which involve objects from the level directly above is *flexible* (a name can go with one monadic predicate, or with one dyadic predicate and another name, or with one triadic predicate and two other names, etc.).[8]

But in the above case the first order properties can go together with a flexible number of individuals: 'peters' can take just 'Sleeps' or 'MarriedTo' together with 'clares'. The first order properties would thus have to be regarded as multigrade properties which can have a variable number of

[8]This point has been made by Carruthers (1983, 53–55) and Gaskin (1998, 27–28).

arguments.

The type-form conventions given above do not enforce this asymmetry of 'upwards flexibility' and 'downwards rigidity'. Of course we can enlarge them by adding the following conditions.

Suppose that a typing assigns s_i to level n such there is some $\phi = [s_i, s_j, \ldots, s_n]$ such that $+\phi$.

> If all the members of ϕ apart from s_i are assigned to level $n-1$, then there is no ψ containing s_i which is of a different cardinality from ϕ such that $+\psi$.
>
> If all the members of ϕ apart from s_i are assigned to level $n+1$, then there are ψ containing s_i which are of a different cardinality from ϕ such that $+\psi$.

As is now easy to check, if we enlarge our type-form conventions in this way then the transformations of LIFTING and MIRRORING together (which allow us to reverse) and FOLDING will generate typings of the graph which are no longer in harmony with the new type-form conventions in cases containing at least one dyadic and one monadic property (LIFTING on its own, however, still remains possible). Thus the most problematic sources of underdetermination are eliminated.

But of course now the question is whether we want to adopt these strengthened type-form conventions which are obviously more complicated than the ones we employed in the beginning. We already considered *these* to be too restrictive and might therefore be wary of accepting anything which confines us even more. The greatest problem with the above strengthened conventions is undoubtedly that they rule out multigrade predicates and properties. And indeed there are some who think that properties essentially have fixed adicities:

> It is naturally an essential part of understanding — that is, 'grasping the sense of' — a relational expression for instance, that you should know that it requires completing by *two* proper names. Similarly it would be natural to think that it must belong to the essence of the relation itself — conceived of as the *reference* of the relational expression — that it should hold between two objects, rather than holding of one or between three.[9]

However, while it is perfectly possible to give a precise semantics for multigrade predicates,[10] it is far from obvious that we can get by with

[9] Carruthers (1983, 54).

[10] See e.g. Taylor and Hazen (1992).

reducing all supposedly multigrade predicates by changing them to fixed-adicity predicates holding of some sort of 'plurality'.[11] If we cannot, then of course we would not want our type-form conventions to forbid the existence of something we obviously need.

1.9 Cardinality

The different levels in the set-theoretic hierarchy are related to one another by cardinality constraints. If there are n objects at some level m, there are exactly 2^n objects at level $m+1$. One might want to transfer this cardinality constraint to ontology and incorporate it into the type-form conventions.[12] The idea would be to demand that a typing is adequate only if the cardinalities of objects at two neighbouring levels m and $m+1$ are related to one another as n and 2^n.

The motivation for such a demand is obvious in the case of such positions as set-theoretic nominalism, which conceives of all properties as sets of individuals. It is clear that according to such an account there cannot be *more* properties of individuals than sets of individuals. But equally once we accept the set-builder we accept all the sets which can be recursively constructed from the individuals. So if properties are really nothing more than sets of individuals, in a world with n individuals there will be 2^n first order properties, 2^{2^n} second order properties and so on, all the way up the set-theoretic hierarchy.

For an intensionalist who claims that for every extension there is a corresponding property, but who does not want to identify properties with extensions (since he fears there might not be enough extensions to account for all the properties), 2^n is obviously only the lower limit: there might conceivably be even more first order properties.

The opposite position would be held by someone who wants to differentiate properties by extensions but who insists that all properties must be instantiated. In this case 2^n would be the upper limit: there could not be more first order properties, but there might be fewer, since the properties corresponding to some extensions might not be instantiated.

Once we incorporate the cardinality constraint into our type-form conventions it is evident that none of the above three transformations is allowed any more. Even the unproblematic LIFTING is ruled out. It is easy to see why this is so. Suppose we have some typing which satisfies the cardinality constraint which we LIFT by two levels. The first two levels will now be empty so that the first objects will not be encountered until level 2. But given that level 0 has cardinality 0, level 1 should have cardinality $2^0 = 1$, which it does not, since it is empty as well. Therefore the typing transformed

[11] Oliver and Smiley (2001).

[12] This consideration was suggested to me by Peter Simons.

by LIFTING no longer satisfies the type-form conventions incorporating the cardinality constraint.

It is similarly obvious that both MIRRORING and FOLDING are ruled out as well. Both transformations affect the number of objects at a given level: MIRRORING by permuting the cardinalities at different levels, FOLDING by increasing the cardinality of one level at the expense of another. Each of these necessarily disturbs the rigid structure imposed by the cardinality constraint.

The asymmetry introduced via the cardinality constraint into the type-form conventions allows us to determine in all cases in which we can construct a typing at all whether some level in the typing is or is not above another one. For example, if our ontological data consist of information about eleven objects, eight of which are grouped together in one level, while the remaining three are in another one, it is clear that the second level must be the more fundamental one, with the first level directly above it.

But such determinacy comes at a price. The price in our case is the impossibility of typing vast ranges of ontological data in a way that obeys the type-form conventions incorporating the cardinality constraint. Data such as those represented in typings 1, 2 and 3 above could no longer be typed with the strengthened type-form conventions. Given that these data are in no obvious way deviant, we should be able to type them with the resources at hand. If our conventions do not allow us to do this, so much the worse for the conventions.

But in fact this is not the only problem with the cardinality constraint. Much more problematic is the fact that the picture of individuals and properties contained in it is highly implausible from an epistemological point of view. If we consider properties as regularities by which we structure information about the individuals we encounter, there should not be more properties than individuals, but considerably fewer. And if the existence of *predicates* of different levels is anything to go by, the picture we get is not that of the bucket or cone familiar from set theory, but rather that of a funnel turned upside-down. It is easy to think of examples of first order predicates, second order predicates are a bit harder to come by, while the number of third order predicates is really very small. Predicates of even higher order are rarely encountered in natural languages. All of this is of course as it should be. There is a great epistemological demand for systematizing similarities between individuals, which is done by the first order predicates. There is less need for regularities between regularities, while we hardly ever need to refer to regularities between regularities of regularities. If there is anything like a cardinality constraint at work in the way we actually type objects in natural language, it seems to be the opposite of the constraint proposed above.

It thus turns out that the cardinality constraint only really makes sense in the context of set-theoretic nominalism. But unfortunately this account

is riddled with so many intrinsic and epistemological problems that I find it hard to consider it as a satisfactory way out. It therefore turns out that neither the appeal to rigidity nor that to cardinality presents a satisfactory way of tightening the type-form conventions. The problem of underdetermination remains.

We therefore see that if we say that an object belongs to a certain type we have certainly not mentioned any essential property of the object. It is even doubtful whether we have said anything important about the object at all, since by making amendments elsewhere, everything could be in any type. Whether certain constituents of states of affairs can go together will constitute ontologically relevant information, but not whether a particular object is assigned to this or that type in a particular typing. It is thus evident that the statement that something is an individual, a property, or a relation cannot claim to express anything of fundamental ontological importance.

2 Adding Logic

We have argued that there is no procedure for drawing a distinction between objects belonging to different types relying solely on the combinatorial information about which objects go together to form states of affairs. But perhaps combinatorial information is not the only information which we can appeal to here.

The ontological data we have considered above provide us with purely combinatorial information, i.e. with information about what can go together with what. As such the data need not be about parts of states of affairs at all: they could describe any collections of objects where it made sense to speak of some objects 'fitting into' some other objects. Our data could for example describe which atoms can go with which other atoms to make up a molecule, or which tiles of some given set can be used to tile a plane, or which nuts fit into which bolts. It is therefore not surprising that we could systematize these data in lots of different ways and that they do not automatically fall into the slots of the simple theory of types in a unique way. One kind of object is not inherently above another, nor is there an inherent 'up' in the hierarchy of levels: these are just results of our theoretical systematization of the data.

We might think that the hierarchy only emerges in a determined (or at least less underdetermined) way once we regard the objects whose composition is encoded in the combinatorial data (i.e. the states of affairs) as objects we can apply a logic to, i.e. as *propositional*. The idea is thus that the simple theory of types is embedded in our conception of inference. It only shows up once we consider not only what goes together with what but also what *follows* from what. This would then distinguish the objects we are considering from atoms, tiles, nuts and bolts: it does not make sense to

say that some molecule or some collection of tiles entails another molecule or another tile, while it *does* make sense to say this about propositional objects. If we take this additional feature into account, much of the above underdetermination might disappear.

Let us therefore explore the idea that although distinctions between objects belonging to different types, and in particular the distinction between individuals and properties, cannot be drawn in a purely combinatorial way, some or all of them can be regained by taking logical relations into account.

2.1 Quantificational reasoning

Quantificational reasoning seems to be what we want to look out for when considering entailment relations. For a simple example consider the possibility of reversing a typing discussed above. Although there are different ways in which to interpret quantifiers, one that has been rather influential is to regard them as higher level properties. In particular, quantifiers over objects at level n are regarded as objects at level $n+2$. (For example, we can regard the first order existential quantifier as a property of a property of individuals, namely as the property of being non-empty.) Now if we have objects at a given level n, there does not seem to be anything to keep us from quantifying over these, thus introducing level $n+2$ objects into our hierarchy. Therefore the hierarchy will always be finite (have some highest level) but potentially infinite: we can make this level as high as we want it to be. Now if we have this picture, it seems to be no longer possible to reverse a typing since we can now identify a top end of the hierarchy: this is the end where new levels can be added to the hierarchy.

Hugh Mellor has argued (in his exegesis of Ramsey) that we should identify the place of individuals in the hierarchy via quantification: they are exactly that which our first order quantifiers range over.[13] If we adopt this position then it is clear why the transformation of LIFTING is ontologically unproblematic.

Our aim is of course to take into account the *inferences* we can draw from particular typings so that the transformations will no longer constitute genuine alternatives, for we are looking for a case in which it can happen that we can draw some inferences from one particular typing which we can no longer draw from another. Here is how we can develop the above idea to do just this.

Consider a language containing quantifiers of different orders $\exists^0, \exists^1, \ldots$. A formula like $(\exists^0 x)(p, x)$ will be interpreted as 'there is some element x at level 0, p is at level 1, and $+[p, x]$'. In general we take $(\exists^n x)(p, x)$ to mean 'there is some element x at level n, p is at level $n+1$, and $+[p, x]$'. Now clearly typing 1 depicted in diagram 1 allows us to infer $(\exists^0 x)(r, x)$, while

[13] Craig (1998, VIII: 48), Mellor (1990, xx).

this is not true in the typing represented in diagram 2. Similarly $(\exists^1 x)(c, x)$ holds in diagram 1 but not in 2, 3 or 4.

So all transformations can make a difference regarding the inferences we can draw. It is, however, possible to let LIFTING come out as a transformation which does not affect inferences by saying that the quantifiers over individuals (i.e. our $\exists^0$) range over *whatever* are the objects at the lowest level, regardless of whether this is level 0 or level 17. To do this we adjust the interpretation of the quantifiers in the following way. Let m be the lowest level at which there are any elements. Then $(\exists^0 x)(p, x)$ will be interpreted as 'there is some element x at level m, p is at level $m+1$, and $+[p, x]$'. This generalizes in the obvious way, $(\exists^n x)(p, x)$ means 'there is some element x at level $m+n$, p is at level $m+n+1$, and $+[p, x]$'. It is then obvious that $(\exists^1 x)(c, x)$ is true in diagrams 1 and 2, but not in 3 or 4.

In this way we can account for the fact that LIFTING on its own, as opposed to LIFTING with MIRRORING, and FOLDING, is supposed to be an ontologically harmless transformation.

However, none of the above approaches attempting to remove the underdetermination by recourse to logic is really satisfactory. Consider first the idea that reversing a typing conflicted with the introduction of higher order quantifiers. This is the case only if we assume that the quantifier is always two levels higher than the objects it ranges over. But in fact this is an additional assumption not implied by the way in which quantifiers (understood as higher order properties) enter into states of affairs. Clearly all that is demanded is that the quantifier is located at a level directly adjacent to that of the properties of the objects it quantifies over. But this can also be the case if it is on the very same level as the objects it quantifies over. Thus we could apply FOLDING and say e.g. that quantifiers over individuals are also a particular kind of individual (since they can form states of affairs with properties of individuals). But then it is clear that the demand for adding quantifiers of ever higher orders does not force the hierarchy to be open-ended in one direction. If our current highest level is m and we want to quantify over objects at level $m-1$ we can introduce a level $m+2$, but we could equally place the quantifier at level $m-1$.

Similarly the proposal inspired by Mellor's exegesis of Ramsey does not seem to be able to provide us with a non-circular logical characterization of individuals. It presupposes that we have a prior way of checking whether some quantifier is first or higher order. But this cannot just be based on taking them to be located at particular levels in the hierarchy for, as we have seen, their position will be affected by transformations as well.

2.2 Distinguishing objects through their logics

A more sophisticated argument for a purely structural distinction between objects at different levels has been developed by Nicholas Denyer.[14] He argues that a structural difference between objects at different levels is implied by the different metalogical properties of logics quantifying over them. Denyer considers first order logic, which quantifies over objects at level 0, pure second order logic, which quantifies over objects at level 1,[15] and standard second order logic, which quantifies over both. First order logic is compact, complete and the Löwenheim-Skolem theorem holds for it. Standard second order logic has none of these properties. Pure second order logic has a property which neither of the other two has: logical truth is decidable in it.

If this works it is surely a most elegant way of removing the underdetermination of typings by appealing to logic. We will then e.g. be able to say that the objects which *really* belong to the zeroth level are those such that the logic quantifying over them has a particular set of properties, while other levels are distinguished by the specific metalogical properties of the logics quantifying over *them*.

But unfortunately several caveats have to be added here. The metalogical properties of a logic depend crucially on the semantics relative to which we determine these properties. The metalogical distinctions between first and second order logic appealed to certainly hold if we give second order logic a standard semantics. But if we take a Henkin or a many-sorted semantics (or first order semantics, as the latter is sometimes called), the differences disappear: second order logic is now compact, complete and the Löwenheim-Skolem theorem holds for it.[16]

Henkin semantics and many-sorted semantics differ from the standard semantics in important ways. In Henkin semantics the analogues of properties are regarded as subsets of the powerset of the domain (and thus as set-theoretical constructions from the individuals). But as compared to standard semantics not all the members of this powerset are included in the semantics (unless we have a full Henkin model, which is then in turn equivalent to a standard model). Many-sorted semantics, on the other hand, takes the denotation of predicates to be objects in some distinct and disjoint sets in the model (and therefore *not* as constructed from the individuals). Because of this, many-sorted semantics needs a primitive predication function which connects the individuals with the properties (standard and Henkin semantics could just use set-membership for this purpose). Henkin and many-sorted semantics are very closely related — for any model of the one sort there is an

[14] Denyer (1998).

[15] For the formal details of pure second order logic see Denyer (1992).

[16] Shapiro (1991, 88–95), see pp. 70–74 for a survey of these different kinds of semantics.

equivalent model of the other sort.[17] However, in the context of our inquiry there are important differences which are brought out by distinguishing the two.

The standard semantics for second order logic seems be attractive because it is a natural continuation of that of first order logic, whereas Henkin or many-sorted semantics deviate from this standard. But such a unity can also be achieved by using a many-sorted semantics for first *and* for second order logic. In such a uniform semantics the metalogical differences just mentioned will disappear.

Let us first consider a simple variant of first order logic which we will call **L1** and which has the peculiarity that it contains the primitive predication predicate ε.[18] Thus instead of formulae like Fa we have $\varepsilon(F, a)$, instead of Gab we have $\varepsilon(G, a, b)$, and instead of $(\exists x)(Fx)$ we have $(\exists x)(\varepsilon(F, x))$. Instead of the non-well-formed formulae aF, aGb, Gb and the like we have $\varepsilon(a, F)$, $\varepsilon(a, G, b)$ and $\varepsilon(G, b)$, which are well-formed but false.

To see how this works consider that the semantics for this first order language is very similar to the standard semantics for standard first order logic. It consists of a domain $D = \{\mathbf{a}, \mathbf{b}, \ldots\}$ of objects and an interpretation function I which assigns an $\mathbf{x} \in D$ to each name of our language, a subset of D to each monadic predicate letter, a set of ordered pairs from D to each dyadic predicate letter and a set of n-tuples from D to each n-adic predicate letter. We shall denote the referents of the names and predicate letters given in this way by underlining them.

A sentence of the form $\varepsilon(F, a)$ is then true in a model of the above kind if $\underline{a} \in \underline{F}$, $\varepsilon(G, a, b)$ is true if $\langle \underline{a}, \underline{b} \rangle \in \underline{G}$. $(\exists x)(\varepsilon(F, x))$ is true if there is at least one $\mathbf{x} \in D$ such that $\mathbf{x} \in \underline{F}$, etc. It is then clear that $\varepsilon(a, F)$, $\varepsilon(a, G, b)$ and $\varepsilon(G, b)$ are all false since they denote the non-obtaining facts that $\underline{F} \in \underline{a}$, $\langle \underline{G}, \underline{b} \rangle \in \underline{a}$ and $\underline{b} \in \underline{G}$.

We thus interpret the predication predicate ε simply as set-membership. In this way the semantics for **L1** does not need any more resources than standard first order logic. Its point is just to make the predication expressed implicitly in standard first order language by concatenation explicit by assigning a special syntactic element to it.[19]

We now introduce a first order language **L2** which is syntactically just like **L1** but which does not have a standard semantics like **L1** but a many-sorted semantics. A model of **L2** will consist of a domain D which is the union of infinitely many sets $D_0, D_1, \ldots$, an asymmetric dyadic relation P between n-tuples from D_0 and one member from D_n (where $n \geq 1$) and the interpretation function I which assigns a member of D_0 to each name, a member of D_1 to each monadic predicate and so on. A sentence $\varepsilon(F, a)$

[17] Shapiro (1991, 88).

[18] Compare the Δ-predicate introduced in Bealer (1982, 82).

[19] Shapiro (1991, 77).

will then be true if $P(\underline{F}, \langle \underline{a} \rangle)$, $\varepsilon(G, a, b)$ will be true if $P(\underline{G}, \langle \underline{a}, \underline{b} \rangle)$, and $(\exists x)(\varepsilon(F, x))$ will be true if there is at least one member of D_0 such that the singleton containing that member stands in the relation P to $\underline{F}$.

This semantics introduces the special relation P as the interpretation of the ε predicate, rather than using $\in$ to fulfil this rôle. This is usually only done in giving many-sorted semantics for *second* order languages[20] but there is no good reason why it should not be done for first order languages as well. It allows us to give a uniform semantics for first order, pure second order and standard second order languages, simply by postulating that in the first case our quantifiers range only over members of D_0, in the second case over members of any set *other* than D_0, and in the third case over members of any D_n whatsoever.

Given that for the third case it can be shown that the system is compact, complete and the Löwenheim-Skolem theorem holds for it, these results also hold for the first and the second case, where we quantify not over members from *all* D_n, but only over some. But in this case the argument for the difference between individuals and predicates from the different metatheoretical properties of the logics quantifying over them hinges on our adopting the standard semantics in all three cases, rather than the many-sorted semantics. We will thus need an argument why the standard semantics is inherently preferable to many-sorted semantics.

The obvious thing to say here seems to be that Henkin models do not allow our quantifiers to range over really *all* properties. In assigning a denotation to e.g. a dyadic predicate-letter in a Henkin model, we might not have the entire set of pairs of individuals in the domain to choose from, but only a limited subset of it. This limitation is essential for the metatheoretical properties of the logic. If it is lifted, we get a full Henkin model (or equivalently a full many-sorted model) which will allow us to prove the same metalogical results as the standard model.[21] We might now argue that if we formalize our talk about 'all properties' of a certain sort, we should choose a semantics in which this is faithfully represented, rather than one which understands our 'all' as an 'all of some particular kind'.

2.3 Blind logic

Let us suppose we accept this criticism. In that case the metalogical distinctions are reinstated. But even so they do not give us the required distinction between individuals and properties. The fundamental problem is that, as we have seen, metalogical distinctions only make sense once we have a semantics. But all usual semantics, including the ones discussed above, already *presuppose* the distinction between individuals and properties we want to draw, since they assign structurally different objects to names and

[20] Shapiro (1991, 75).
[21] Shapiro (1991, 89).

to predicate letters (objects to the former and sets of objects to the latter in the standard case, objects from different sets in the many-sorted case). Therefore the metatheoretical differences appealed to are actually the result of the distinction between individuals and properties drawn at the level of semantics, rather than a guide to such a distinction.

In all of the above systems the properties of the predication function (regardless of whether we took it to be set-theoretical membership or to be primitive relation) provided us with an obvious way of identifying the individuals in the semantics. There were always some items in the semantics which could not occupy the first position in the predication function — the ones which could not have any members nor could be predicated of something. These, of course, were the individuals. This identification relies on the fact that the predication function is asymmetric: if a can be predicated of b (b is a member of a) b cannot be predicated of a (a cannot also be a member of b) and b must therefore be an individual. But the fact that there is such a predication function present is not a fact of logical necessity. In fact we can devise a logic, a *blind logic* as one might call it, which can do everything which the above logics can, but whose semantics does not contain a predication function but only a weaker symmetric substitute, called *application*. In this logic individuals and properties cannot any longer be distinguished in the above way.

Let us look at a toy example of such a blind logic. The alphabet of our language will consist of the seven names t, j, p, f, l, n, o, multigrade application predicate A, the propositional connectives $\neg$ and $\wedge$, a way of expressing quantification, and the necessary punctuation conventions. The names will be divided into three groups, $\tau_1 = \{t, j, p\}$, $\tau_2 = \{f, l\}$ and $\tau_3 = \{n, o\}$. An atomic sentence is well-formed if it has one of the two following forms:

$$A(\tau_1, \tau_2)$$

$$A(\tau_1, \tau_1, \tau_2)$$

Writing down the application predicate, followed by a name from the first and one from the second group will therefore result in a wff, as will writing down the application predicate followed by two (not necessarily distinct) names from the first and one from the second group. We further stipulate that if $A(\tau_i, \ldots, \tau_k)$ is well-formed, so is every formula obtained by permuting the τs.

Molecular sentences sentences will be formed from atomic sentences in the familiar way: if some sentences ϕ, ψ are wffs, so will be $\neg\phi$ and $\phi \wedge \psi$. Furthermore, if ϕ is well-formed, replacing one name belonging to τ_i in it by an x and prefixing the result with an $\exists x_{\tau_i}$ is also well-formed.

A model of this language consists of three sets $D_1 = \{T, J, P\}$, $D_2 = \{F, L\}$ and $D_3 = \{N, O\}$, a set S of multisets of elements of D_1–D_3 and an interpretation function I which assigns an object from some D_i to every

name and an element from S to every atomic sentence. A sentence of the form $A(u, v)$ will be true if $[I(u), I(v)]$ is in S. $\neg\phi$ is true if ϕ is not true, $\phi \wedge \psi$ if both ψ and ϕ are. $\exists x_{t_i} A(x, u)$ is true if there is some x in the D_i such that $[I(x), I(u)]$ is in S.

There is also an intuitive interpretation of this language. In this interpretation T, J and P stand for three Cambridge Colleges, Trinity, St John's and Peterhouse, F stands for the property 'being first in the league-table', L for 'is liked by tourists', O for 'is as old as' and N for 'is next to'. So $A(j, l)$ is true if there is an actual state of affairs containing just St John's and 'being liked by tourists' (i.e. if St John's is liked by tourists); $A(t, l) \wedge A(t, j, n)$ is true if both Trinity is first in the league-table and Trinity is next to St John's; and $\exists x_{\tau_i} A(x, p)$ if either Peterhouse is liked by tourists or it is first in the league-table.

Now it should be clear that everything a standard first order theory of the three Colleges, their two properties and three relations can express can be expressed using this blind logic. However, as opposed to standard systems with standard or many-sorted semantics we cannot draw a distinction between the different items in the model according to the position they occupy in expressions of the form $A(u, v)$, since e.g. $A(u, v)$ and $A(v, u)$ will be logically equivalent.

Intuitively we would claim that T, J and P in the model are individuals, F and L first order monadic and N and O first order dyadic properties. This is surely a sensible assumption to make. But there is nothing in the semantics which forces us to go down that road. As we have seen in the previous sections, we could equally regard T, J and P as first order monadic properties and F, L, N and O as different kinds of individuals. Or T and J could be taken to be individuals, P as a second order property and F, L, N and O as first order properties which can take arguments both from level 0 and from level 2.

2.4 Ordered application

The reader may have realized already that the expressive equivalence between our theory of the three Colleges formulated in blind logic and in standard first order logic holds only because we took care to include just symmetric relations. Assume now that we had included some non-symmetric predicates (such as 'admires') or asymmetric predicates (such as 'has more students than'). In this case expressive equivalence would not have been preserved, since clearly we can express that Trinity has more students than St John's in the standard first order language as $M(t, s)$, while the blind logic equivalent $A(m, t, s)$ only tells us that m can go together with t and s in a state of affairs, but not in which direction it does.

In order to get full expressive equivalence with standard first order logic we therefore have to add to our logic some device for expressing this 'direc-

tion' or 'ordering' of elements. We do this by introducing a second application predicate A'. The syntax of this will be similar to that of A, with the difference that it will apply to different sorts of names. For example if we add the two names a, m to the above language (standing for the predicates 'admires' and 'has more students than') which belong to a type τ_4, we would amend the above rules for forming atomic sentences by adding

$$A'(\tau_1, \tau_1, \tau_4)$$

Since A' is supposed to express 'ordered' predication it is obviously not the case any more that if $A'(\tau_i, \ldots, \tau_k)$ is well-formed, so is every permutation of the τs. We had multisets to serve as the denotations of 'unordered' application and will consequently have lists to serve as the denotations of 'ordered' application. Thus we add a set L of lists of elements from D_1–D_3 to the model and stipulate that a sentence of the form $A'(u, v)$ is true if (u, v) is in L. We can then e.g. express sentences like 'St John's admires Trinity but not the other way round' as $A'(j, t, a) \wedge \neg A'(t, j, a)$, something which obviously could not be done in a language containing A alone. It is then clear that a language containing the two application predicates has the same expressive power as standard first order predicate calculus.

Note furthermore that the expressive power does not decrease if we just restrict ourselves to A'. Everything we can express by using unordered application can be expressed by ordered application. Such a system will be identical to a many-sorted first order system with a single special multigrade predicate representing ordered application.

2.5 Partially ordered states of affairs

We saw above that on the basis of a blind logic containing just the application predicate A we are not able to distinguish the elements of the domain which are individuals from those which are properties. We then realized, however, that this logic fell short of the expressive power of standard first order logic. We had to add the ordered application predicate A' in order to get a language in which we could say the same things as in standard first order logic.

Having such an ordered application predicate makes for a simple syntax, but given that we are presently concerned with the extent to which formal syntactic features are merely convenient conventions, rather than necessary in some 'more fundamental' sense, let us pause for a moment to see whether we really need the full power of ordered application for achieving what we set out to do.

It turns out that we do not. In order to represent the difference between 'Albert loves Becca' and 'Becca loves Albert' we do not need a framework which fixes the order of *all three* constituents of that state of affairs. All we

need to know is whether — in a manner of speaking — Albert or Becca comes first in that state of affairs. The exact position of 'loves' is then irrelevant; we could position it anywhere in the state of affairs. Therefore it seems to be much more sensible to express the state of affairs that Albert loves Becca not as $A'(l, a, b)$ but rather as $A(l, A'(a, b))$. This notation emphasizes that the state of affairs is not completely flexible, as are states of affairs involving symmetric properties such as 'Albert marries Becca', where the order of constituents is irrelevant, but partially flexible: the order of some, but not of all its constituents is important. $A(m, a, b)$ is identical with $A(m, b, a)$, but $A(l, A'(a, b))$ is different from $A(l, A'(b, a))$.

We should be immediately warned, however, against trying to exploit this syntactical convention for ontological purposes, e.g. by claiming that the constituents whose order is important are the individuals, while the remaining one is the property. That would be reading too much into the syntax. *That* we need some device for expressing the order of (some) constituents of states of affairs is syntax-independent. This is something made necessary by our desire to develop a language with a certain expressive power. But *how* this is achieved is determined by convention, even if we settle for the above syntactic framework. After all, the difference between 'Albert loves Becca' and 'Becca loves Albert' could equally be expressed by writing $A(a, A'(l, b))$ for the one and $A(a, A'(b, l))$ for the other. If we adopt *this* convention, saying that the individuals are the elements linked by ordered application will deliver intuitively false results.

Apart from providing us just with the minimal resources we need for achieving expressive equivalence with standard first order logic, the above notation has the additional advantage of bringing out a distinction between states of affairs which is usually overlooked by focusing just on the difference between symmetric and non-symmetric predicates.

We have seen above that there are states of affairs where the order of constituents is completely irrelevant. These unordered states of affairs are usually considered to contain symmetric properties such as 'is married to' or 'is as tall as'. Nevertheless, states of affairs with just two constituents can also be subsumed under this heading, given that it is irrelevant in which order the individual is linked with the monadic property.

Other states of affairs, such as those containing non-symmetric or asymmetric properties depend on the order of all their constituents but one, as we have seen in the case of 'loves' above. These are the ordered states of affairs.

There is, however, a third class of states of affairs which we will call partially ordered. In these the order of only a small number of constituents is important. An example of a partially ordered state of affairs is 'that a is between b and c'. Clearly this is not unordered, but equally it is not ordered, since it is not the case that the order of all but one constituent is important. This state of affairs has four constituents, 'is between', a, b, and c. But we

do not have to know the order of three of them to identify uniquely which state of affairs is meant; to know the order of two is quite sufficient. After all, 'that a is between b and c' is the very same state of affairs as 'that a is between c and b'.

The same distinction between three kinds of states of affairs can be drawn if we consider their *combinatorial potential*. The combinatorial potential of a state of affairs is the number of different states of affairs which can be put together from its constituents. Unordered states of affairs obviously have a combinatorial potential of 1: there is only one state of affairs which can be constructed by putting together the constituents of 'Albert is married to Becca', namely that very state of affairs. Ordered states of affairs, on the other hand, have a combinatorial potential of $(n-1)!$, if they have n constituents. This is due to the fact that all the permutations of all the constituents but one of such a state of affairs result in different states of affairs.[22] The combinatorial potential of partially ordered states of affairs will be between that of unordered and that of ordered states of affairs. The state of affairs 'a is between b and c' has a combinatorial potential of 3; an unordered state of affairs with four constituents would have one of 1, an ordered state of affairs one of 6.

2.6 Ordered states of affairs and natural language

The problem of distinguishing a state of affairs like 'Albert loves Becca' from its converse is obviously not just relevant for the formal language we have been discussing, but is equally acute for natural languages. Different languages have different ways of tackling this problem. In English the fact that 'Albert' comes before 'Becca' in 'Albert loves Becca' indicates that Albert is the lover and Becca the beloved, rather than the other way round. The two states of affairs are thus distinguished by *word order*.

The second linguistic mechanism by which this can be achieved is *case marking*. To indicate in Latin that Domitilla sees Polybius we put Domitilla's name in the nominative case and Polybius' in the accusative: 'Domitilla Polybium videt'. In the case of triadic relations like 'x gives y to z' we also employ the dative, as in 'Domitilla Polybio librum dat'.

Each of the two methods is each on its own able to fulfil the task of distinguishing different 'directions' of relations. Once a case marking system is in place we do not need to appeal to word order for the same purpose, or vice versa. In Latin the word order is quite flexible, though not completely arbitrary. There are, however, examples of languages employing case markers such as the native American polysynthetic language Mohawk

[22] $(n-1)!$ is the greatest combinatorial potential a state of affairs can have. For any state of affairs there will thus be always more permutations of its elements than different states of affairs which can be constructed from it.

where sometimes every permutation of the words of a sentence is a sentence as well.[23]

Whether a language employs word order or case marking seems to be a quite fundamental difference which is decided by how very general parameters are set.[24] The formal languages of logic have always gone for employing word order rather than case markers. One reason for this presumably is that it keeps the size of the primitive vocabulary down, since case markers would constitute additional primitives. The above syntactic device of partially ordered application can be taken to contain some elements of such case marking within a system relying fundamentally on word order, since the order (like the case marking) applies only to certain parts of the sentence, not to the entirety of its constituents.

2.7 Distinctions which can be drawn

We have now seen that neither ordered nor partially ordered application gives us a way of distinguishing individuals and properties. Let us therefore look at the distinction between the objects in the domain which *can* be drawn in terms of the two application predicates A and A'. The criteria of well-formedness tell us which objects can go together to form states of affairs. For example, from the fact that $A(\tau_1, \tau_1, \tau_2)$ is well-formed we know that one object of the kind denoted by names of type τ_2 and two objects of the kind denoted by names of type τ_1 can be assembled into a state of affairs. This is essentially the same information which is formulated in the graph-theoretic model developed in 1.3–1.6. In this model this fact would be expressed by the fact that each collection of vertices consisting of one member of τ_2 and two members of τ_1 would be part of a cluster of three elements.

This information is not sufficient, however, for achieving a distinction between individuals and properties. We cannot argue that, in a state of affairs described by a formula of the form $A(\tau_1, \tau_1, \tau_2)$, τ_2 is a dyadic property since it takes two other objects to form a state of affairs, because this is also true of the τ_1 (this is essentially the point made in 1.6). We are in the strange situation that we can structurally establish whether some state of affairs contains an n-adic constituent but not which constituent this is. On the basis of all the combinatorial and logical information which we can bring in to decide this problem, there is no way of settling the issue. There seems to be no fact of the matter and therefore no distinction between the ontological categories of individuals and properties.

[23] Baker (2001, 88–89).

[24] For the notion of a hierarchy of parameters and the rôle of the polysynthesis parameter see Baker (2001, chapter 6).

2.8 Conclusion

Let us briefly summarize the results we have reached. In the first section we argued that there is no procedure for distinguishing between objects of different types on the basis of purely combinatorial information about what goes together with what in states of affairs. In the second section we explored whether one of the type-distinctions between the constituents of states of affairs, namely that between individuals and properties, could be salvaged by wheeling in information about logical relations to supplement that about combinatorial relations. Our conclusion was negative since the distinctions logic allowed us to draw were only those which it had already previously made at the level of semantics. Logic is therefore unable to provide us with information about either the origin or the justification of the individual-property distinction.

Both the combinatorial and the logical information can be regarded as *structural* information if we follow Carnap in equating structural information with *relation-descriptions* and non-structural information with *property-descriptions*.[25] Relation-descriptions tell us about certain relations objects stand in, while property-descriptions (unsurprisingly) tell us about their properties. Combinatorial information is structural because it is about the way the different parts of states of affairs are related in 'fitting together' to form complexes. The description of logical relations is structural because it is about entailment relations between states of affairs. As such the above can be taken as a (partial) argument for the impossibility of a structural foundation of the distinction between individuals and properties of different orders and adicities. I consider the philosophical significance of the above results to consist in showing that this distinction between individuals and properties cannot be as fundamental as is usually assumed.

Of course one might argue that there are other ways of drawing the distinction, based on relations which can equally be regarded as structural. Unfortunately I presently do not see any plausible candidates for such supplementary structural relations. Ramsey examined a number of possible procedures in his paper *Universals*,[26] which he calls the psychological, the physical and the logical. The last one corresponds closely to our conception of a 'structural' account. As we have seen above, this does not lead to success. The other two accounts do not look much more promising and are (rightly) given short shrift by Ramsey .[27] His conclusion is therefore corrob-

[25] Carnap (1928, 11-12). Carnap claims that relation-descriptions and thus structural descriptions are the *only* descriptions science deals with.

[26] Ramsey (1990).

[27] The psychological account, based on the distinction of different mental acts does not play any rôle in the present ontological discussion. This is not so for the physical account, which argues that properties, but not individuals can be at more than one place at any one time. For a criticism of this and a conclusion which is in some respects quite similar to our own see MacBride (1998).

orated: it seems high time to dismiss talk about individuals and properties from the set of fundamental ontological distinctions. If we want to confine ourselves to distinctions which *can* be drawn structurally, we have to restrict ourselves to weaker notions codifying which objects fit together to form states of affairs. How far they can be incorporated into existing ontological frameworks and what a purely structurally grounded ontological theory would look like remain fascinating and fundamental questions for further research.*

Appendix

T 1. *A typing of a graph according to the type-form conventions where some vertex a is at level n can by a successive applications of the three transformations be made into another typing which is also in accordance with the data but where a is located at level m, for any m.*

It is clear that we can move a graph arbitrarily far upwards by LIFTING. Showing that this also holds for the downwards direction is a bit more difficult. We will prove the downwards case, the full T1 will then be entailed.

T 2. *Let a (the* candidate*) be a vertex in a graph located at level n (the* candidate-level*). It is possible to transform that graph by repeated application of the three transformations* LIFTING, MIRRORING *and* FOLDING *so that in the result a is at any level n' (the* target-level*), provided $n' < n$.*

We first observe that since the set of data we consider is always finite, each graph in a typing will contain some *maximal* element (which is not connected to any vertex at a higher level) and some *minimal* element (which is not connected to any vertex at a lower level). If t is the level of the maximum, b that of the minimum, the length l of the graph will just be $(t - b)$.

We now need to establish two simple lemmas.

L 1. *If a is a vertex in a graph in some typing the graph can be transformed in such a way that in the result a is minimal.*

Proof. If a is not maximal, we FOLD the graph upwards at a. Now everything which used to be below a is above it, so that a is minimal. If a is maximal, we simply mirror the entire graph upwards, so that the maximal element becomes the minimal element.

□

*Thanks are due to Georg von Hippel, Michael Potter, Hugh Mellor, Nicholas Denyer, Uwe Scheffler, and Peter Simons, as well as to audiences at Cambridge, Lund, Leeds, and Bonn for helpful comments on earlier versions of this paper.

We will therefore always assume in the following that the candidate is minimal.

L 2. *If the candidate is at level n,* MIRRORING *downwards x times results in a graph whose minimal element is located at level $n-(xl-x)$.*

It is now possible to describe a procedure for moving the candidate to any lower target-level which will serve as a proof of T2.

Proof. We distinguish three cases:

1. The candidate is at level n and the target-level is $n-(xl-x)$, where x is an even number ≥ 2.
2. The candidate is at level n and the target-level is lower than $n-(xl-x)$.
3. The candidate is at level n and the target-level is higher than $n-(xl-x)$.

In the first case by L2 we know that we can just apply MIRRORING downwards x times, which moves the entire graph downwards, so that the candidate is now at the target-level.

In the second case we apply MIRRORING downwards until applying it two times more would either be impossible (because there would not be 'enough space' below) or would move the candidate below the target-level. We have then reduced the second to the third case.

In the third case we know that the target-level is $(n-(2l-3))$ or higher. Now if we want to move the candidate to $(n-(2l-3))$, we first LIFT the entire graph by one level and then MIRROR downwards two times. The graph will then have been moved downwards, with the candidate now being at the target-level. If we want to move the candidate to $(n-(2l-4))$ we first LIFT the entire graph by two levels and then MIRROR downwards two times. Generally, if we want to move the candidate to $(n-(2l-y))$, where $(2l-y)>0$ and $y \geq n$ we first LIFT the graph $y-2$ levels and then MIRROR downwards two times.

□

References

Mark C. Baker. *The Atoms of Language. The Mind's Hidden Rules of Grammar.* Oxford University Press, Oxford, 2001.

George Bealer. *Quality and Concept.* Clarendon, Oxford, 1982.

Wayne D. Blizard. Multiset theory. *Notre Dame Journal of Formal Logic*, 30(1):33–66, 1989.

Rudolf Carnap. *Der logische Aufbau der Welt.* Weltkreis-Verlag, Berlin, 1928.

Peter Carruthers. On object and concept. *Theoria*, 49(2):49–86, 1983.

Edward Craig, editor. *Routledge Encyclopedia of Philosophy.* Routledge, London, 1998.

Nicholas Denyer. Pure second-order logic. *Journal of Symbolic Logic*, 33(2): 220–224, 1992.

Nicholas Denyer. Names, verbs and quantification. *Philosophy*, 73:619–623, 1998.

Richard Gaskin. The unity of the declarative sentence. *Philosophy*, 73: 21–45, 1998.

Fraser MacBride. Where are particulars and unviversals? *Dialectica*, 52(3): 203–227, 1998.

D. H. Mellor, editor. *F.P. Ramsey. Philosophical Papers.* Cambridge University Press, Cambridge, 1990.

Alex Oliver and Timothy Smiley. Strategies for a logic of plurals. *Philosophical Quarterly*, 51(204):289–306, 2001.

Frank P. Ramsey. Universals. In D. H. Mellor, editor, *Philosophical Papers*, pages 8–33. Cambridge University Press, Cambridge, 1990.

Stuart Shapiro. *Foundations without Foundationalism. The Case for Second-Order Logic.* Oxford University Press, Oxford, 1991.

Barry Taylor and A.P. Hazen. Flexibly structured predication. *Logique & Analyse*, 139–140:375–393, 1992.

Alfred North Whitehead and Bertrand Russell. *Principia Mathematica.* Cambridge University Press, Cambridge, 1925.