Basic Semantic Integration

Christopher Menzel

Texas A&M University
Department of Philosophy
College Station, Texas 77843-4237
cmenzel@tamu.edu

Abstract. The use of highly abstract mathematical frameworks is essential for building the sort of theoretical foundation for semantic integration needed to bring it to the level of a genuine engineering discipline. At the same time, much of the work that has been done by means of these frameworks assumes a certain amount of background knowledge in mathematics that a lot of people working in ontology, even at a fairly high theoretical level, lack. The major purpose of this short paper is provide a (comparatively) simple model of semantic integration that remains within the friendlier confines of first-order languages and their usual classical semantics and logic.

Keywords: ontology, semantic integration, first-order logic, model theory, SCL

1 Introduction

The important work of Joseph Goguen ([6], [7]), Robert Kent ([8]), Marco Schorlemmer and Yiannis Kalfoglou ([11], [12]), and others point the way toward very promising general framework for characterizing of a variety of concepts of ontology integration. Such high-level frameworks are essential for the sort of theoretical foundation for semantic integration needed to bring it to the level of a genuine engineering discipline. At the same time, this work is done in the rarefied theoretical air of category theory and channel theory, and therefore assumes a certain amount of background knowledge that a lot of people working in ontology, even at a fairly high theoretical level, lack. In fact, however, while this work is far more abstract and, concomitantly, far more general and far-reaching in its implications and applicability, I believe some of the most basic insights beneath the idea of semantic integration can be expressed in terms of basic first-order logic and model theory. Moreover, I believe it is important to do so to provide relatively simple, comparatively concrete accounts of integration that can help to fix the basic ideas of the emerging theory for the broader community of ontological engineers. The major purpose of this brief paper, then, is provide a simple model of integration that remains within the friendlier confines of first-order languages and their usual classical semantics and logic. The model might also serve as a sort of "bidirectional" test-bed for the higher-level theoreticians as well — any virtues of the approach that are not reflected in the higher-level theories can be appropriated by them, and any infelicities in the approach can be corrected on general grounds provided by the theories.

Dagstuhl Seminar Proceedings 04391
Semantic Interoperability and Integration
http://drops.dagstuhl.de/opus/volltexte/2005/42

The approach I'll discuss is quite similar in certain respects to the one outlined by Ciocoiu and Nau in their short paper [2]. I have myself in the past been somewhat critical of the approach for being a bit too model theoretic in orientation (see [9]), and that may still well be for some of the applied purposes that Ciocoiu and Nau have in mind. Once again, however, the point here, is theoretical — to fix the ideas needed to provide a proper *conceptual* ground for building the actual infrastructure to support integration, much as the ϵ-δ definition of a derivative provided a conceptual foundation for the mathematics actually used to build bridges and fly spacecraft to distant planets. And on this count, a model theoretic approach serves admirably well.

1.1 Ontologies

As with about everything having to do with semantic integration, there are many different definitions of what an ontology is. Perhaps the best known is one of the earliest, from Tom Gruber: An ontology is a "specification of a conceptualization" [REF]. There is a certain appeal to this proposal — an ontology begins with a certain way of conceptualizing the world, or some prominent piece of it, and this conceptualization is made concrete — specified — in some fashion. The question remains exactly what a specification is, of course, but a natural understanding is that a specification is some sort of concrete representation, e.g., an ER diagram or a set of axioms in a given language. This understanding, in turn, suggests that ontologies can in fact be *identified* with their representations. There is reason to hesitate at this idea, as there is also an intuition that the *same* ontology can be expressed in different languages; and indeed one could take this intuition as a starting point (see, e.g., [9]). Here, however, we will individuate ontologies by their representations and, to capture the intuition noted, develop instead the idea that two distinct ontologies can have the same *content*.

At the same time, we have to acknowledge that, intuitively, ontologies are more than sets of sentences. The primitive terms of an ontology also have *intended* meanings. However, in general, for applied languages, the notion of an intended model is essentially unformalizable and as such, though critically important, it is not a matter for theory but for methodology and practice. A formalization of semantic integration can only provide an answer to the question *what it means* to integrate disparate ontologies. While we can hope to write programs that render aid and comfort to the task, the hard work of determining intended meanings will always ultimately require human intervention.

2 Languages: SCL and Abstract Syntax

Developing an account of semantic integration, even at the more concrete level presented here, still requires some level of generality. It won't be much of a general theory if it restricts its attention just to, say, OWL and RDF. Rather, we need an account of what a language is that provides an *abstract structural* characterization of any possible, or at least any reasonable, Knowledge Representation (KR) / Semantic Web (SW) language without specifying any of the concrete details of the language. This permits a variety of languages that differ in the concrete details to flourish without engendering a

"Tower of Babel" problem — insofar as each language comports in some fashion with the general characterization of a language.

Providing such a characterization has been a large part of the motivation of the (Simplified) Common Logic project (http://cl.tamu.edu), where a very general, very abstract notion of a syntax is defined, one designed to encompass the needs, choices, and preferences of any possible concrete language. (See [13] for a reasonably friendly version, and [14] for a hostile formal version.) I will use a very compressed and hand-waving version of SCL here.

Of course, as noted, for such a framework for integration to be effective, KR/SW languages must be comport with the general characterization. Thus, another goal of SCL is to serve as a clear standard with which KR/SW languages can demonstrably comport. I will illustrate briefly how this is done below.

2.1 Syntax

A *language* consists of a *signature* and a *grammar*.

Signatures The signature of a language L consists of a set of *syntactic classes*. These must divide into a class of *variables* and two (not necessarily disjoint) classes: *individual constants*, *predicate constants*. and *function symbols*. (That individual and predicate constants can overlap comes from earlier versions of KIF, but also reflects an important syntactic feature of RDF; see [15].) An element of a syntactic class is an *atom* — typically, a string consisting of elements from some set of basic characters (e.g., unicode characters). We will assume a countable number of atoms in each class. (This is an innocuous assumption that will smooth the approach to integration below.)

Grammars Informally, a grammar is a set of rules (typically recursive) that specify how to construct *well-formed* expressions from atoms and other, less complex, well-formed expressions. We can formalize the notion of a grammar in terms of a set of one-one functions with pairwise disjoint ranges. More specifically, every grammar will include a function APP that forms terms from function symbols some terms, and a function PRED that forms atomic sentences from a constant and some terms. (Arity for predicates can be introduced as a separate notion.) CONJ and DISJ form sentences from any finite number of sentences; COND and BICOND form sentences from pairs of sentences; EXQUANT and UNIVQUANT form sentences from a sequence of pairwise distinct variables and a given sentence. Notions of bondage and freedom for variable occurrences can be defined straightforwardly.

Example 1: KIF As examples I will choose use (a simplified version of) KIF and a standard sort of first-order language. Although the latter is not a language for the Semantic Web, that is beside the point here — we are concerned to nail down some notions of integration between ontologies in different languages, and techniques will apply regardless of choice of language. So for purposes here, languages have been chosen that are efficient, easy to work with, and (not least) with which the author is particularly familiar. Future work will explicitly encompass RDF, OWL, and other more explicitly Web-oriented languages.

Syntactic Classes KIF's syntactic classes consist of *constants*, *variables*, and *sent-ops*. Constants are strings of alphanumeric characters, dashes, and underscores. Variables are simply constants prefixed by '?'. The sent-ops are: `not`, `and`, `or`, `implies`, `iff`, `forall`, and `exists`.

KIF's grammar is as follows; I will use corner quotes to indicate *quasi-quotation* that allows to use metalanguage variables ranging over linguistic objects to indicate general classes of expressions;[1]

– Every constant is both an individual constant and a predicate constant.
– Constants and variables are *terms*
– $\ulcorner(\pi\ \tau_1\ \ldots\ \tau_n)\urcorner$ is an (atomic) sentence, for constants π and terms τ_i.
– $\ulcorner(\mathtt{not}\ \varphi)\urcorner$ is a sentence if φ is.
– $\ulcorner(\mathtt{and}\ \varphi_1\ \ldots\ \varphi_n)\urcorner$ and $\ulcorner(\mathtt{or}\ \varphi_1\ \ldots\ \varphi_n)\urcorner$ are sentences if the φ_i are.
– $\ulcorner(\mathtt{implies}\ \varphi\ \psi)\urcorner$ and $\ulcorner(\mathtt{iff}\ \varphi\ \psi)\urcorner$ are sentences if φ and ψ are.
– $\ulcorner(\mathtt{forall}\ (\nu_1\ \ldots\ \nu_n)\ \varphi)\urcorner$ and $\ulcorner(\mathtt{exists}\ (\nu_1\ \ldots\ \nu_n)\ \varphi)\urcorner$ are sentences if φ is, for any variables $\nu_1, \ldots, \nu_n$.

Example 2: A Typical First-order Language SCL is designed to allow for languages with maximal (first-order) expressiveness, a feature that is particularly desirable for languages designed chiefly for purposes of representation rather than automated reasoning. Not all languages, of course, will want to make use of all of those features. Some will also wish to impose more structure than SCL requires, e.g., by the assignment of arities to predicates. SW languages in particular will put restrictions on permissible sentences. Nonetheless, all such languages can be considered conformant "as far as they go" insofar as their sentences constitute a recursive subset of a fully compliant SCL language; such conformance is not hard to show for most any SW language. Here the point will be illustrated simply with a more traditional first-order language L.

Syntactic Classes The syntactic classes of L are *individual constants*, *variables*, *predicate constants*, and *sentence operators*. Individual constants are lower case letters a – t, possibly with numerical subscripts. Variables are lower case letters u – z, possibly with numerical subscripts. Predicate constants are upper case letters A – Z with numerical superscripts and possible with numerical subscripts. (A predicate with numerical superscript n is an *n-place predicate*. Sentence operators are $\neg,\ \wedge,\ \vee,\ \rightarrow,\ \leftrightarrow,\ \forall,\ \exists$.

Grammar The grammar for L is as follows:

– Constants and variables are *terms*
– $\ulcorner\pi\tau_1\ldots\tau_n\urcorner$ is an (atomic) sentence, for n-place predicates π and terms τ_i.
– $\ulcorner\neg\varphi\urcorner$ is a sentence if φ is.
– $\ulcorner(\varphi \wedge \psi)\urcorner$, $\ulcorner(\varphi \vee \psi)\urcorner$, $\ulcorner(\varphi \rightarrow \psi)\urcorner$, and $\ulcorner(\varphi \leftrightarrow \psi)\urcorner$ are sentences if φ and ψ are.
– $\ulcorner\forall\nu\varphi\urcorner$ and $\ulcorner\exists\nu\varphi\urcorner$ are sentences if φ is, for any variable ν.

[1] C and Perl programmers might consider by analogy the difference between single quotes and double quotes.

Obviously this is a recursive sublanguage of a fully compliant SCL language — extracted by the assignment of arities to predicates and the elmination of non-binary conjunctions and disjunctions and the binding of multiple variables. So this language can be considered conformant "as far as it goes".

2.2 Semantics

We will assume a fairly standard model theory, albeit one that has a bit more flexibility to it to accommodate possible overlap between the individual and predicate constants. Specifically, an interpretation for a language consists of two sets of objects — individual and relations; relations are assigned sets of n-tuples as their extensions. Each individual constant κ is assigned an individual $den(\kappa)$ as its denotation and each predicate constant and function symbol π a relation $den(\pi)$ — with the added stipulation that the extension of the relation assigned to a function symbol must be functional.[2] For an interpretation **M** with individuals M, relations R, extension function *ext* and denotation function *den*, then we then have that an atomic sentence PRED$(\pi, \tau_1, ..., \tau_n)$ is *true in* **M** just in case $\langle den(\tau_1), ..., den(\tau_n)\rangle \in ext(den(\pi))$. The remaining clauses are just as one would expect, notably:

- A quantified sentence EXQUANT(ν, φ) [UNIVQUANT(ν, φ)] is true in **M** just in case just in case, for some [every] individual $e \in I$, φ is true in M$[\nu/e]$, where M$[\nu/e]$ is just like **M** except that the denotation function for **M** assigns e to the variable ν.

An interpretation **M** of a language L is a *model* of a set of sentences O of L just in case every member of O is true in **M**. O *entails* a sentence φ just in case φ is true in every model of O.

3 Ontologies Defined

We noted above that we will be taking ontologies to be identified with their representations. In this context, this means that we identify ontologies with their axioms, as expressed in some language. Specifically: An *ontology* O in a language L is a class of sentences of L. The members of O are called the *axioms* of O. We stipulate without any loss of generality that, for any ontology O in a language L, there must always be countably many atoms of L that do not occur in any of the sentences of O. This smooths the definition of semantic integration below. Given simple facts about infinite cardinalities, this too is an innocuous assumption.

4 Semantic Mapping as Meaning Preserving Translation

This basic framework already provides a fairly robust formal notion of semantic mapping on which to build. Intuitively, a semantic mapping is (as far as possible) a *meaning*

[2] That is, if a (e.g., 1-place) function symbol σ is assigned relation r as its denotation, then if $\langle e_1, e_2\rangle \in ext(r)$ and $\langle e_1, e_3\rangle \in ext(r)$, then $e_2 = e_3$.

preserving translation from the language of one ontology into that of another. Here's one way to cash this out using standard concepts from first-order model theory. (I am draw heavily upon [3] and [5] in what follows.) Let $O2$ be an ontology in a language L2. (For the sake of familiarity, I will let L2 be the standard language of first-order logic.) Let φ be a sentence of L2 with a single free variable ν such that $O2$ entails $\ulcorner\exists\nu\varphi\urcorner$. Define a *Φ-map* from a language L1 into O2 to be a function α from the individual constants, function symbols, and predicate constants of L1 into sentences of L2 such that:

– For each individual constant κ of L1:
  - Only the variable ν occurs free in $\alpha(\kappa)$, and
  - $O2$ entails $\ulcorner\exists_1\nu(\varphi \wedge \alpha(\kappa))\urcorner$.

  When $\alpha(\kappa)$ is of the form $\nu = \lambda$ for some constant λ of L2, let $\alpha^\circ(\kappa)$ be λ.
– For each n-place function symbol σ of L1:
  - Exactly the (distinct) variables $\nu,\ \nu_1,\ ...,\ \nu_n$ occur free in $\alpha(\sigma)$,
  - None of the variables $\nu_1, ..., \nu_n$ occurs in φ, and
  - O2 entails $\ulcorner\forall\nu_1...\nu_n((\varphi^\nu_{\nu_1} \wedge ... \wedge \varphi^\nu_{\nu_n}) \rightarrow \exists_1\nu(\varphi \wedge \alpha(\sigma)))\urcorner$.[3]

  When $\alpha(\sigma)$ is of the form $\nu = \beta(\nu_1, ..., \nu_n)$ for some function symbol β of L2, let $\alpha^\circ(\sigma)$ be β.
– For each n-place predicate symbol σ of L1:
  - Exactly the (distinct) variables $\nu_1,\ ...,\ \nu_n$ occur free in $\alpha(\sigma)$.

The idea here is straightforward. A *Φ*-map is designed to take the non-logical elements of the lexicon of a language L1 of a given ontology $O1$ into sentences of the language L2 of a target ontology $O2$ that, in a certain intuitive sense, preserve their meaning. Thus, first of all, φ is intended to carve out the intended domain of $O1$ from that of $O2$. Thus, intuitively, the *Φ*-map of a constant κ in L1 is a sentence of L2 with a free variable that is true of the same object that κ denotes. (In the simplest case, there is a constant λ in L2 that intuitively denotes the same thing that κ does in L1. In this case the *Φ*-map of κ is simply the sentence $\nu = \lambda$.) Similarly, the *Φ*-map of an n-place function symbol σ will, intuitively, be a sentence in $n + 1$ variables that expresses a (functional) relation that is definable in $O2$ and which is true of $n + 1$ things $a_1, ..., a_n, a_{n+1}$ (all satisfying φ) if and only if the function that σ denotes maps the objects $a_1, ..., a_n$ to a_{n+1}. (Again, if there is a single function symbol β of L2 that intuitively expresses the same function as σ, then the *Φ*-map of σ will simply be $\nu = \beta(\nu_1, ..., \nu_n)$.) And, finally, the *Φ*-map of an n-place predicate constant π should be a sentence in n variables that, in $O2$, expresses the n-place relation relation denoted in L1 by π.

Given any model **M2** of $O2$, then, a *Φ*-map induces an interpretation **M1** of L1 whose domain consists of the things in the domain of **M2** that satisfy φ, and which interprets a constant κ by the thing satisfying $\alpha(\kappa)$, an n-place function symbol σ by the set of $n+1$-tuples satisfying $\alpha(\sigma)$, and n-place predicate symbol π by the set of n-tuples satisfying $\alpha(\pi)$. Intuitively, then, a *Φ*-map α from a language L1 to an ontology $O2$ preserves the meaning of the non-logical vocabulary of L1 as circumscribed by an ontology $O1$ if, given any model of $O2$, the interpretation of L1 induced by α is a

[3] φ^ν_τ means the result of replacing every free occurrence of ν in φ with an occurrence of τ.

model of $O1$. We make this precise as follows; for simplicity we assume that individual constants, function symbols, and predicate symbols are pairwise disjoint classes.

Let α be a Φ-map from L1 to $O2$, let **M2** = $\langle D2, R2, ext, den\rangle$ be a model of $O2$, and let $\mathbf{M2}[\nu/e]$ be the interpretation that is just like **M2** except that it maps ν to e. Define the interpretation $\mathbf{M2}^{-\alpha} = \langle D1, R1, ext, den\rangle$ of L1 *induced* by α as follows:

- $D1 = \{e \in D2 : \varphi$ is true in $\mathbf{M2}[\nu/e]\}$.
- $R1 = \{\{\langle e_1, ..., e_n\rangle \in D1^n : n > 0$ and φ is true in $\mathbf{M2}[\nu_1/e_1, ..., \nu_n/e_n]\}$: φ is a sentence in which $\nu_1, ..., \nu_n$ occur free$\}$.
- $ext(r) = r$, for $r \in R1$.
- $den(\kappa) =$ the unique $e \in D1$ such that $\alpha(e)$ is true in **M2**, for constants κ of L1.
- $den(\sigma) = \{\langle e_1, ..., e_n, e_{n+1}\rangle \in D1^{n+1} : \alpha(\sigma)$ is true in $\mathbf{M2}[\nu_1/e_1, ..., \nu_{n+1}/e_{n+1}]\}$.
- $den(\pi) = \{\langle e_1, ..., e_n\rangle \in D1^n : \alpha(\pi)$ is true in $\mathbf{M2}[\nu_1/e_1, ..., \nu_n/e_n]\}$.

We can now define a Φ-map of L1 into $O2$ to be *meaning preserving for* an ontology $O1$ in L1 iff, for any model **M2** of $O2$, the interpretation $\mathbf{M2}^{-\alpha}$ of L1 induced by α is a model of $O1$.

A Φ-map α yields a natural, fully-fledged *translation* function α^* from the sentences of L1 into those of L2 that enables us to define meaning preservation relative to the translation of one ontology $O1$ into another $O2$. For simplicity's sake, we will assume that there is a constant of L2 corresponding to each constant of L1, and a function symbol of L2 corresponding to each function symbol of L1, so that we can use the shorter notation that is allowed above under these conditions:

- $\alpha^*(\nu) = \nu$, for variables ν of L1. (We assume for simplicity that L1 and L2 share the same variables.)
- $\alpha^*(\kappa) = \alpha^\circ(\kappa)$, for constants κ of L1.
- If τ is a function term $\ulcorner\sigma(\omega_1, ...\omega_k)\urcorner$, $\alpha^*(\tau) = \ulcorner\alpha^\circ(\sigma)(\alpha^*(\omega_1), ..., \alpha^*(\omega_k))\urcorner$
- If φ is an atomic sentence $\mathrm{PRED}(\pi, \tau_1, ...\tau_n)$, $\alpha^*(\varphi) = \ulcorner\alpha(\pi)(\alpha^*(\tau_1), ..., \alpha^*(\tau_n))\urcorner$
- As expected for the boolean cases.
- If φ is $\mathrm{EXQUANT}(\nu, \psi)$, then $\alpha^*(\varphi) = \ulcorner\exists\nu(\varphi \wedge \alpha^*(\psi))\urcorner$.
- If φ is $\mathrm{UNIVQUANT}(\nu, \psi)$, then $\alpha^*(\varphi) = \ulcorner\forall\nu(\varphi \rightarrow \alpha^*(\psi))\urcorner$.

The axioms of an ontology infuse its basic lexicon with meaning by putting constraints on how the atoms in the lexicon can be jointly interpreted. A translation function α^* (relative to some Φ-map α) of those axioms into the language of another ontology will be meaning preserving, relative to the given ontologies, just in case those constraints are respected, i.e., just in case the axioms of the source ontology — upon translation under α^* — are all entailed by the target ontology. This can happen only if the constraints on the lexicon of L1 expressed by the axioms of $O1$ are respected — upon translation — by $O2$. More formally then:

Definition 1. Let $O1$ and $O2$ be ontologies in languages L1 and L2, respectively, and let α^* be the translation function from L1 into L2 generated by a given Φ-map. Then α^* is *meaning preserving with respect to O1 and O2* if, for any axiom φ of $O1$, $O2$ entails φ.

It should be obvious that, if a Φ-map from L1 to $O2$ is meaning preserving for $O1$, then its corresponding translation function α^* will be as well. Given this, we can formulate a simple, initial notion of semantic mapping:

Definition 2. A *semantic mapping* from one ontology $O1$ in a language L1 into an ontology $O2$ in a language L2 is a translation function α^*, relative to a given Φ-map α, from L1 to language L2 that is meaning preserving with respect to $O1$ and $O2$.

4.1 A Vivid Formal Example

A well known mapping for number theory into set theory provides a particularly vivid example of semantic mapping so defined. Examples of this kind, because of their formality, can often be misleading, as they abstract away from exactly all of the messy real world problems of ontology integration. However, bear in mind once again that at this point we are only trying to fix ideas — we need a clear notion of the concepts we are *striving for*, ideally, even if, in practice, we can only approximate it. For this purpose, mathematical examples like this one that filter out real world "noise" can be helpful and effective.

The usual language of arithmetic L_{PA} is a first-order containing '$+$', '$\cdot$', a function symbol 's' for the successor function, and the numeral '0'. The usual axioms of Peano Arithmetic (PA) — the most familiar number theory — are the following. First, basic axioms for '0' and 's':

- $\forall x(s(x) \neq 0)$ (0 is not the successor of any number.)
- $\forall x \forall y(s(x) = s(y) \rightarrow x = y)$ (Successor is 1-to-1.)

Next, the basic recursion axioms for '$+$' and '$\cdot$':

- $\forall x(x + 0 = x)$
- $\forall x \forall y(x + s(y)) = s(x + y))$
- $\forall x(x \cdot 0 = 0)$
- $\forall x \forall y(x \cdot s(y) = (x \cdot y) \cdot x)$

Finally, the induction schema. Let φ^{ν}_{σ} be the sentence that results from replacing all free occurrences of the variable ν in φ with occurrences of σ:

- $(\varphi^x_0 \wedge \forall x(\varphi \rightarrow \varphi^x_{s(x)})) \rightarrow \forall x \varphi$, for any sentence φ in which 'x' occurs free.

The usual language L_{ZF} of Zermelo-Fraenkel set theory is a first-order language whose lexicon contains only the one binary predicate '$\in$'. For simplicity we will also assume that the language contains the one individual constant '$\emptyset$' and the binary function symbols '$\cup$' ("union") and '$\times$' ("cartesian product"), axiomatized by their usual definitions. We will also make use of the usual bracket notation ... for defining finite sets, which can also be defined in familiar ways. Now, first, define the "successor" $sc(A)$ of a set A to be $A \cup \{A\}$. Beginning with the empty set $\emptyset$ and iterating the successor operation yields the set of so-called finite von Neumann ordinals: $\{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}, \{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}, ...\}$, the standard representation of the natural numbers in modern set theory. We can therefore take our sentence Φ to be the sentence expressing property of being in *every* set that contains $\emptyset$ and is closed under sc (Φ then will be true of exactly the finite von Neumann ordinals); let's give Φ the more concrete name '*Num*' in this context:

$$Num =_{df} \ulcorner \forall y((\emptyset \in y \wedge \forall z(z \in y \rightarrow sc(z) \in y)) \rightarrow x \in y \urcorner .$$

Now, taking "equal in size to" as usual to mean "can be put into one-to-one correspondence with", for finite sets A and B:

- $sum(A, B) =$ the Num that is equal in size to $(A \times \{\emptyset\}) \cup (B\{sc(\emptyset)\})$.[4]
- $prod(A, B) =$ the Num equal in size to $A \times B$.

(For our purposes we don't care what sum and $prod$ do with infinite sets.) Given these definitions, we can define a Num-map α from $\mathrm{L_{PA}}$ into ZF. We note first that ZF entails '$\exists x Num$', as required. Next, we let α('0') ='$x = \emptyset$' and we note that ZF obviously entails '$Num(\emptyset)$' and so, by logic, ZF entails '$\exists_1 x(Num \wedge x = \emptyset)$' (as is required of a Φ-map for constants in source language).[5]

Next, we let α('$+$') = '$sum(x, y) = z$' and α('$\cdot$') = '$prod(x, y) = z$'. We note again that ZF entails '$\forall xy(Num \wedge Num[x/y] \rightarrow \exists z(Num[x/z] \wedge sum(x, y) = z)$'; similarly for '*prod*'. It is a well-known fact that, given a model of ZF, the interpretation that this Num-map induces on $\mathrm{L_{PA}}$ is a model of PA. Notably, when **M** is the "intended" standard model of ZF, the induced interpretation of $\mathrm{L_{PA}}$ has the set of von Neumann ordinals as its domain, sc as the interpretation of 's', and sum and $prod$, restricted to the von Neumann ordinals, as the interpretations of '$+$' and '$\cdot$', respectively.

This Num-map α yields an obvious concomitant, meaning preserving semantic mapping α^* from the language $\mathrm{L_{PA}}$ into $O2$. (we will assume that the two languages have identical variables).

- $\alpha^*(\tau) = \tau$, if τ is a variable of L1.
- α^*('0') = '$\emptyset$'
- $\alpha^*(\ulcorner s(\tau) \urcorner) = \ulcorner sc(\alpha^*(\tau)) \urcorner$
- $\alpha^*(\ulcorner \tau = \sigma \urcorner) = \ulcorner \alpha^*(\tau) = \alpha^*(\sigma) \urcorner$
- $\alpha^*(\ulcorner \tau + \sigma \urcorner) = \ulcorner sum(\alpha^*(\tau), \alpha^*(\sigma)) \urcorner$
- $\alpha^*(\ulcorner \tau \cdot \sigma \urcorner) = \ulcorner prod(\alpha^*(\tau), \alpha^*(\sigma)) \urcorner$
- $\alpha^*(\ulcorner \neg \varphi \urcorner) = \ulcorner \neg \alpha(\varphi) \urcorner$
- $\alpha^*(\ulcorner (\varphi \wedge \psi) \urcorner) = \ulcorner (\alpha^*(\varphi) \wedge \alpha^*(\psi) \urcorner$; similarly for the other binary connectives.
- $\alpha^*(\ulcorner \exists \nu \varphi \urcorner) = \ulcorner \exists \nu (Num^x_\nu \wedge \alpha^*(\varphi)) \urcorner$
- $\alpha^*(\ulcorner \forall \nu \varphi \urcorner) = \ulcorner \forall \nu (Num^x_\nu \rightarrow \alpha^*(\varphi)) \urcorner$

It follows almost trivially that if α is a meaning preserving Num-map, then α^* is a semantic mapping from PA to ZF; anything that the number theoretic ontology PA can say about numbers is something ZF says about them in their set theoretic guise. ZF, however, says a lot more besides; in particular, for definiteness, ZF chooses to identify

[4] We can't define addition in terms of $\cup$ alone, of course, because every member of *Num* is a subset of every larger member. Hence, for any two Nums A and B, $A \cup B$ will just be the larger of the two. The trick here is simply that we "paint" the members of A and B, respectively, "different colors" by pairing the members of A with $\emptyset$ and the members of B with its singleton $\{\emptyset\}$. The union of these "painted" sets will then be the right size to represent addition.

[5] '$\exists_1 x \varphi$' says that there is exactly one thing satisfying φ, and can be defined in the usual way as '$\exists x \forall y(\varphi \leftrightarrow x = y)$'.

the numbers with a particular set that exemplifies the structure described by PA, and, more significantly, it generalizes the notion of number into the transfinite.[6] [7]

It is worth re-emphasizing that what we are after here is a definition of what semantic mapping *is*. Such a definition does not of itself yield any immediate insight into *how* to map one ontology into another; it does not *generate* the translation from the source ontology to the target. That requires insight into the intended meanings of the axioms of both ontologies. The definition only tells us what it is for such a translation to be semantically *correct*.

5 Semantic Integration: Bridge Axioms and Merging

In actuality, of course, it will rarely be the case that a one ontology can be mapped entirely into another the way that PA can be mapped into ZF. Much more likely is that neither ontology will contain all of the content of the other. Rather, when one $O1$ is mapped into the other $O2$, $O1$ will bring new information that is not implicit in $O2$. It is not enough for genuine intergration, however, simply to take the union of the two ontologies. For in general, the information in $O1$, while not strictly contained in $O2$ (under an appropriate translation function) will have many logical connections to the information in $O2$ that are explicit in *neither* ontology. Fully-fledged semantic integration, then, will require identifying these logical connections and making them explicit. Axioms introduced to make these connections are called *bridge axioms*, formulated in the language of $O2$. (Recall that we have required there always to be countably more n-place predicates in the language of an ontology than actually occur in the axioms of the ontology.) True integration between two ontologies, then, will involve a semantic mapping from $O1$ in $O2$ *plus* a set of bridge axioms. The result of such a "merge", then, will be a new ontology incorporating the information from both and their salient logical connections. We might begin, then, with the following definition:

Definition 3. A *merge* of ontology $O1$ into $O2$ is a triple $\langle \Phi, \alpha, B \rangle$, where Φ is a sentence of L2, B is a set of bridge axioms in the language L2 of $O2$, and α is a meaning-

[6] [1] is still about as good an introduction to transfinite arithmetic as there is. For its modern development in ZF, see, e.g., [4].

[7] It might be argued that our formal example is perhaps a bit misleading in that it involves not simply a semantic mapping but what philosophers sometimes call an ontological "reduction" (see [10] — talk of numbers is "reduced" to talk of a certain class of sets. But this is actually a bit inaccurate, as what ZF provides is not so much a different ontology than PA but simply a higher degree of specification. For Peano Arithmetic is not really an ontology of a certain well-defined set — *the natural numbers*. It actually makes no claim about, and doesn't concern itself with, what the numbers are *really*. Rather, it is about a certain type of *structure*, one that can be exhibited by infinitely many different sets. ZF simply provides one particularly convenient set to play this role. In real world contexts however, typically, more than structure is at issue; rather, there is some definite ontology in question that distinct ontologies *share* in common, at least in part. In these cases, what a meaning preserving translation with respect to ontologies $O1$ and $O2$ will show is, not that a certain class of entities can be "seen as" another class, the way numbers can be seen as sets in ZF, but rather that the target ontology $O2$ talks about *the very same things as* $O1$, and perhaps more besides.

preserving Φ-map from L1 into $\alpha^*[O1] \cup O2 \cup B$.[8]

This needs refinement, however, as the notion of a bridge axiom is undefined; and, indeed, if we allow any sentence of L2 to count as a bridge axiom, then the above definition allows for "trivial" merges in which the bridge axioms are the result of simply translating the axioms of $O1$ into L2 in such a way that every atomic sentence of L1 is mapped to sentence of L2 that involves no constants or predicates that occur in 02. The result would be a "merge" of $O1$ and $O2$ in which the information expressed in each ontology was completely isolated from the information in the other; though merged into one, the two ontologies would in effect remain entirely unintegrated.

Obviously, what's missing here is the idea of *bridging* that a bridge axiom should embody: A bridge axiom should *connect* the objects and concepts of $O1$ logically to those of $O2$. This can happen in two ways. First, and perhaps most typically, a bridge axiom will involve at least one (translation of a) constant of $O1$ and at least one non-logical constant occurring in the axioms of $O2$. This motivates the following definition:

Definition 4. Let $\langle \Phi, \alpha, B \rangle$ be a merge of ontologies $O1$ and $O2$. The bridge axiom $\beta \in B$ is a *connecting axiom* if it contains at least one constant (individual or predicate) or function symbol of L2 and is such that, for some constant or function symbol χ of L1, $\alpha(\chi)$ contains at least one constant or function symbol of L2 not occurring in any axiom of $O2$.

This gets us closer, but the definition still allows for the possibility that particular connecting axioms could be trivial in a certain sense — notably, we could construct tautologies that satisfy the definition of a connecting axiom; or the axiom could add nothing to the work already done by the translation scheme T. While ensuring that all of the bridge axioms in a merge are doing some heavy lifting is perhaps more a pragmatic, even aesthetic, issue than a theoretical one, it might still be useful to have a rigorous notion of nontrivial to serve as an ideal for bridge axioms to meet. We do this by formalizing the insight that a nontrivial connecting axiom should impose a (consistent) constraint on the interpretation of the "union" of the two ontologies:

Definition 5. Let $\langle \Phi, \alpha, B \rangle$ be a merge of ontologies $O1$ and $O2$. A connecting axiom $\beta \in B$ is *nontrivial* if $\alpha^*[O1] \cup O2$ is consistent with, but does not entail, β, i.e., if β is true in in some models of $\alpha^*[O1] \cup O2$ and false in others.

As intimated above, not all conceivable bridge axioms are connecting axioms. A second possibility is that an axiom of $O1$, when translated, might nontrivially extend $O2$, but add no new vocabulary. For example, suppose in a given automobile manufacturing ontology $O2$ there are only cars with two doors. Suppose now this ontology is merged with a different automobile ontology $O1$ in which there are in fact cars with four doors. The concept "sedan" is therefore definable using concepts available in $O2$, but the axiom "There are sedans" is not provable from $O2$. Call this sort of axiom an *augmentation axiom*:

[8] Where, in general, for a function $f : A \longrightarrow B$, and where $C \subseteq A,\ f[C] = \{f(a) : a \in C\}$. So $\alpha^*[O1]$ is simply the image of the ontology $O1$ in $O2$ under the semantic mapping from L1 into L2 generated by α.

Definition 6. Let $\langle \Phi, \alpha, B \rangle$ be a merge of ontologies $O1$ and $O2$. An bridge axiom $\beta \in B$ is an *augmentation axiom* if every constant and function symbol of β occurs in some axiom of $O2$ but $O2$ does not entail β.

Given these definitions, we can define a merge $\langle \Phi, \alpha, B \rangle$ to be nontrivial just in case B contains at least one augmentation axiom or one nontrivial bridge axiom.[9]

6 Semantic Mapping and Practical Integration

A final word about practical semantic integration that reflects the ideas worked out here. Given two ontologies $O1$ and $O2$ to be merged, one can at the outset, typically, make no assumptions whatever about the logical relations between the constants in those ontologies, even — or perhaps better, especially — when the constants are similar or identical. Rather the logical connections between the constants of the two ontologies is something that must, in general, be stipulated later in the integration process, either through the addition of bridge axioms or through the subsequent development of a translation scheme that, to some extent at least, identifies the concepts and objects in one ontology with those of another. Typically, though, it will be useful to *defer* the question of logical connections and simply form an initial "union" of the two ontologies in which the information in each is sequestered from the information in the other. The easiest way to accomplish this is simply by means of a sort of quasi-merge that is in fact trivial in the sense above. In such a merge, the translation function that maps the atomic sentences of one ontology $O1$ to sentences of L2 that share no constants in common with any of the axioms of $O2$. One can then incrementally identify logical connections explicitly by means of bridge axioms, or by refining the translation function in such a way that some sentences of $O1$ are translated entiredly into sentences of L2 that are theorems of $O2$. One then moves incrementally toward a lean and robust ontology by the addition of genuine, nontrivial bridge axioms.

7 Conclusions

In this brief paper, I've drawn upon basic, familiar notions of first-order logic to make some initial steps toward a rigorous theory of semantic integration. Drawing on SCL, we introduced an abstract, structural notion of a language. Such a treatment of languages is necessary for a general account of integration between languages that differ considerably in their concrete features. One must in these cases describe integration in terms of more general, abstract strutural features of the languages in question. Using this framework, a general notion of semantic mapping was defined as meaning preservation, where this is spelled out model theoretically in terms of the notion of a Φ-map: a mapping from the basic lexicon of a given ontology into (hopefully) equivalent concepts of another ontology. Φ-maps, in turn, yield translation functions that, under proper conditions, can be considered semantic mappings between ontologies. This simple notion

[9] As a final touch, one could also require that one's bridge axioms are *nonredundant*, i.e., that for no $\varphi \in B$ is it the case that $\alpha^*[O1] \cup O2 \cup (B - \{\varphi\})$ entails φ.

of semantic mapping is rather limited, as it only applies to cases where one ontology subsumes another in a certain well-defined sense. In the penultimate section, therefore, the notion of a semantic mapping was broadened to that of a *merge* that gives us a notion of a semantic mapping for two ontologies that only overlap in meaning. We closed with a final reflection on the relation between these formal notions and the methodology of real world integration. It is hoped that the notions introduced here make some progress — in approach, at least, if not in actual content — toward a rigorous, well-defined engineering discipline of ontology integration.

References

1. Cantor, G., *Contributions to the Founding of the Theory of Transfinite Numbers*, New York, Dover Publications 1955. Translation of "Beiträge zur Begrundung der transfiniten Mengenlehre," *Mathematische Annalen* **46** (1895) 481-512 and **49** (1897) 207-246.
2. Ciocoiu, Mihai and Nau, Dana S., "Ontology-based semantics" in A. G. Cohn, F. Giunchiglia, B. Selman (eds.), *Principles of Knowledge Representation and Reasoning, Proceedings of the Seventh International Conference*, Morgan Kaufmann, 539-546, URL=http://citeseer.ist.psu.edu/ciocoiu00ontologybased.html.
3. Ebbinghouse, H.-D., Flum, J., and Thomas, W., *Mathematical Logic*, 2nd edition, New York, Springer, 1994.
4. Enderton, H., *Elements of Set Theory*, New York, Academic Press, 1977.
5. Enderton, H., *A Mathematical Introduction to Logic*, New York, Academic Press, 1972.
6. Goguen, J., "A Categorical Manifesto," *Mathematical Structures in Computer Science*, **1** (1991) 49-67.
7. Goguen, J. and Burstall, R., "Institutions: Abstract Model Theory for Specification and Programming." *Journal of the Association for Computing Machinery*, **39**(1) (1992) 95-146.
8. Kent, Robert, "The IFF Foundation for Ontological Knowledge Organization," in Knowledge Organization and Classification in International Information Retrieval (Haworth, 2003).
9. Menzel, Christopher, "Ontology Theory," in J. Euzenat, A. Gomez-Perez, N. Guarino, and H. Stuckenschmidt (eds.), *Ontologies and Semantic Interoperability, CEUR Workshop Proceedings*, **64** (2002), URL=http://CEUR-WS.org/Vol-64/menzel.pdf.
10. Quine, W. V. O., "Ontological Relativity." *Journal of Philosophy* **65** (1968) 185-212. Reprinted in his *Ontological Relativity and Other Essays*, New York, Columbia University Press, 1977.
11. Schorlemmer, M. and Kalfoglou, Y., "On Semantic Interoperability and the Flow of Information," in *ISWC '03 Semantic Integration Workshop*, Sanibel Island, Florida, USA, October 2003. URL=http://citeseer.ist.psu.edu/schorlemmer03semantic.html
12. Schorlemmer, M. and Kalfoglou Y., "Using Information-flow Theory to Enable Semantic Interoperability," in *6e Congres Catala en Intelligencia Artificial*, Palma de Mallorca, Spain, Oct. 2003. Also available as Informatics Report EDI-INF-RR-0161, The University of Edinburgh. URL=http://citeseer.ist.psu.edu/schorlemmer03using.html.
13. Common Logic Working Group, "SCL: Simple Common Logic," URL=http://www.ihmc.us/users/phayes/SCLJune2004.html.
14. Common Logic Working Group, "Abstract Syntax and Semantics for SCL," URL=http://cl.tamu.edu/docs/scl/scl-latest.html.
15. World Wide Web Consortium (P. Hayes, ed.), "RDF Semantics," URL=http://www.w3.org/TR/rdf-mt.