

What, if anything, does it mean to Import Knowledge

**Alexander Borgida
Dept. of Computer Science
Rutgers University**

Antecedents in Software Development

Analogy with Programming Languages:

- **Why import?**

Independent development. Reuse of code, testing,...

- **How to import?**

- Cut and paste

- Function library calls

- `#include` header file (in C,...)

- create *modules* that *export* types, constants, functions; then have programs *import* modules (in ADA,...)

- explicit import statements of the form (in Python)

```
from YourModule import name1 as name2, name3 as  
name4, ...
```

Semantics and Issues:

module specification, implementation hiding, change
“propagation”

Importing Knowledge - the framework

INPUT:

- Knowledge base KB_{exp} containing specification of the meaning of terms (concepts, attributes, associations, individuals,...) using some KR&R language/logic. (**Think OWL or UML formalized in FOPC**).
- Expressed as a variety of “judgements” α (consistency, implication, subsumption, membership, disjointness, ...)
- logic of reasoning given by entailment $KB_{exp} \models \alpha$
- set of identifiers of interest $S \subset \text{vocab}(KB_{exp})$

GOAL: Developer of KB_{imp} wants to be able to use names from the signature $S = \{\text{Name}_1, \dots\}$, so that the “semantics” of the names are preserved.

ISSUES:

- **Specifying what is the effect of importing**
 - what can you do with the imported names syntactically?
 - **only use in limited context?**
 - what is the result semantically?
 - **"reason as if you included entire KB_{exp} "**
- **(Implementing this specification)**
 - by caching locally only what is needed
 - using distributed reasoning

A real-life example

ON 9.3 medical ontology (in KIF)

<http://www.loa-cnr.it/medicine/anatomy/class3.html>

Anatomy module

Theories included by Anatomy:

Meronymy, Positions, Topo-Morphology

The following constants were used from included theories:

- * 3d-Area-Of defined as a relation in theory Topo-Morphology
- * > defined as a relation in theory Kif-Numbers
- ... (60+ other terms)

The following constants were used from theories not included:

- * Anatomical-Abnormality defined as a class in theory Abnormalities
- * Antigen defined as a class in theory Biologic-Substances
- ... (20+ other terms)

(Aside: a tiny intro to Description Logics)

- Express “noun phrases” (unary predicates, concepts) in a term-like notation without variables that start from
 - atomic concepts (e.g., BOOK, JOURNAL, PERSON, FEMALE)
 - atomic roles (binary relations) such as (author_of, child_of, located_at, taken_by)and build up complex concepts using empirically useful “constructors”

and (PERSON, FEMALE)

PERSON \wedge FEMALE,

not (**or** (BOOK, JOURNAL))

\neg (BOOK \vee JOURNAL)

some (author_of, FEMALE)

\exists author_of.FEMALE

all (author_of, PERSON)

\forall author_of.PERSON,

- **Assertions/judgements**

➤ *Subsumption* BOOK :< \forall author_of.PERSON

➤ *Definition* MOTHER_OF_JOHN \equiv PERSON \wedge FEMALE \wedge
 \exists child_of.{John}

➤ *Instance of* PERSON (John), MOTHER_OF_JOHN (Megan)

➤ *Inferences about the above*

\models B :< and (B, C)

Some categories of approaches

1. **Ontology_j** creates “modules” $M_{j,k}$ *a priori*. These modules are then imported as a unit (so imported concept name N comes with everything in its module)

I. user-specified views/modules

e.g., the ON9.3 medical ontology we saw before

II. automatic modularization

- on more or less syntactic (graph theoretic) grounds
- on logical grounds

2. KB_{imp} specifies list of names to be imported, and system *automatically* chooses a set of axioms from KB_{exo}

III. ontology trimming

used esp. with general/top-level ontologies like Cyc, WordNet

IV(a). “import” statements with logical semantics

IV(b). “import” statements with “local-domain” semantics

II. Modules by Auto Segmentation (“graphs”)

"Web Ontology Segmentation: Analysis, Classification and Use"
Seidenberg & Rector [WWW 06]

- **INPUT:**
 - large ontology T in OWL
 - concept name N in T
- **DESIRED:**

$Module(N, T) = \text{subset of vocab}(T) + \text{axioms restricted to it, which is “small and focused”}.$
- **METHOD:** (*informally*)
 1. Module = $\{N\} + \text{subclasses of } N$
 2. Module += superclasses of N and role restrictions of N
 3. Module += super-roles of roles in Module
 4. Repeat 2 & 3 for other concepts in Module

Could be described as path traversal pattern in graph G_T with edges representing **IsA**, roles/properties (restrictions on them)

(cont'd)

EXPERIENCE: Basic segment of the Heart concept in Galen
size

	orgnl	sgmnt	difference
number of classes	23139	5794	25%
primitive classes	13168	2771	21%
defined classes	9971	3023	30%
number of props	522	380	71%
filesize in KB	22022	5815	26%

REFINEMENTS:

- add a list of properties to be filtered out: R_
- limit traversal depth, resulting in boundary classes

METHOD: collapse concepts that now have identical definitions

other Auto Segmentation (graphical)

Stuckenschmidt & Klein [ISWC 2004]

"Structure-Based Partitioning of Large Concept Hierarchies"

Automatic Segmentation into Modules based on Logic

"Modularity and Web Ontologies"

Cuenca-Grau, Parsia, Sirin & Kalyanpur [KR 2006]

module(Concept N , Terminology T) = "subset of T needed to capture standard inferences (satisf'n, classific'n, instantiat'n)"

SPECIFICATION:

- $T \models N \rightarrow X$ iff $module(N,T) \models N \rightarrow X$
- $T \models X \rightarrow N$ iff $module(N,T) \models X \rightarrow N$
- there is no $N2 \in module(T,N)$, $N3 \notin module(T,N)$ whenever $T \models N2 \rightarrow N3$ or $T \models N3 \rightarrow N2$

IMPLEMENTATION:

- produce directed graph and partitioning of axioms and concept identifiers of T such that $module(N,T)$ = collection of axioms on node with N on it, which has above properties; nodes connected by roles connecting modules
- module = node
- works efficiently for OWL if T is consistent and "safe" (a reasonable property)

Automatic Segmentation (logical 2)

"A modularity approach for a fragment of ALC"

Herzig & Varzinczak [JELIA 2006]

INPUT: Terminology T with *ALC* subsumption axioms

OUTPUT: Modified theory T_2 (with $\text{vocab}(T) \subset \text{vocab}(T_2)$) such that

- T and T_2 are equivalent concerning judgments involving identifiers in $\text{vocab}(T)$
- T_2 is composed of modules

$$T_2^\emptyset = \{T_2 \text{ axiom involving no role names}\}$$

$$T_2^R = \{T_2 \text{ axioms involving only role } R \text{ and concept names}\}$$

with the property that for concept expressions C and D

$$T \models C \rightarrow D$$

iff

$$T_2^\emptyset + T_2^{\{\text{roles in } C \text{ and } D\}} \models C \rightarrow D$$

III. Modules as View Definitions

"Specifying Ontology Views by Traversal"

Noy & Musen [ISWC 04]

INPUT:

- ontology T in RDF or OWL
- starting focus concepts N (eg "*Heart*")
- property directive set $\{ (P_1, n_1), (P_2, n_2), \dots \}$
eg $\{ (PartOf, 3), (ContainedIn, 1) \}$

RESULT: {N} + restriction/range/value classes of property P_j +
repeat directives on these after decrementing count by 1.

GENERALIZATION: algebra of such directives

Q: Resulting ontology has boundary concepts - status is unclear.

Other view definition work

- **Stuckenschmidt & Klein [IJCAI 2003]**
"Integrity and Change in Modular Ontologies"
conjunctive queries over roles and concepts
- **Magkanaraki, Tannen, Christophides & Plexousakis [ISWC 2003]**
"Viewing the Semantic Web through RVL Lenses"
query language over RDF
- ...

IV(a). Import Terms by Ontology Winnowing

GIVEN

- a general ontology O_G (WordNet,Cyc),
- a local ontology O_L specific to a subdomain
- connections from concepts in O_L and O_G

DESIRED

- subontology O_S of O_G , with which one can extend O_L
($O_S =$ “what should O_L import from O_G ”)

Winnow Ontology (1)

"Extending, Pruning and Trimming General Purpose Ontologies"

Navigli [2nd IEEE SMC 2002]

INPUTS: upper ontology O_G (WordNet)

+ specific domain ontology in the form of trees T_1, T_2, \dots

METHOD

1. map tree roots R_k to synsets in O_G using heuristics (we ignore this)

Winnowing:

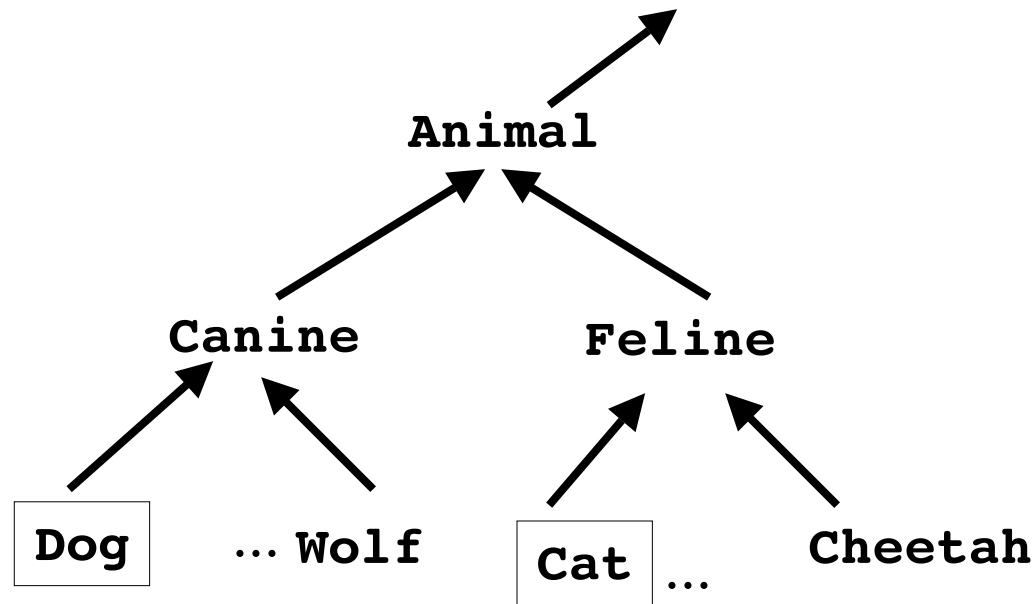
a) top down prune: remove nodes *not* on a path from TOP to some R_k (but possibly keep siblings of R_k)

b) bottom up trim: starting from $A=R_k$, moving upward, repeatedly remove node B if (A is-a-child B) and there is *no sibling of A under B*

EXPERIENCE:

start: 539 roots; WordNet : 66,000 nodes; prune:596 +539; trim: 480 +539

Desirable effect



When importing **{Dog, Cat}** from the above hierarchy, it makes sense to also get **Animal**, but there is *no necessity* to also get **Canine** and **Feline**. Nor all superclasses of **Animal**.

Winnow Ontology (2)

"A Method for Pruning Ontologies in the Development of Conceptual Schemas of Information Systems"

Conesa & Olivé [JoDS 2006], [DEXA 03],...

INPUT:

- extended ontology O_X = generic ontology (e.g., Cyc)
- domain specific conceptual schema (e.g., in UML,OWL)
- names of interest (N_of_I) - **names in schema, action descriptions**

OUTPUT specification: refined ontology O_P that is

- a subset of O_X
- $N_of_I \subset \text{vocab}(O_P)$
- O_P includes all constraints in O_X whose vocab is in $\text{superclassesOf}(N_of_I)$
- O_P *is minimal*

METHOD

1. prune unnecessary concepts and constraints

- remove concepts not in $super(N_of_I)$ /* superclasses may be useful because of inherited props, and to get IsA's */
- remove constraints f with $vocab(f) \not\subset super(N_of_I)$

2. prune unnecessary parents

- $NeededConcepts = N_of_I \cup \{vocab(f) \mid vocab(f) \subset super(N_of_I)\}$
- $PossiblyUnneeded = super(N_of_I) - NeededConcept$
- remove concepts that are not descendants of Needed

3. prune unnecessary IsA paths:

- if there are alternate IsA paths, keep only one

EXPERIENCE

	Entity types	Attributes	Assoc'ns	Integr. Constrnts
Cyc + Extension	12+2900+	9	8 + 1300+	6
O_X	2715	255	1397	6
O1	140	15	30	6
O2	106	15	11	6
O_P	75	15	11	6

KBus [1998]

B. J. Peterson, W. A. Andersen, J. Engel, "Knowledge Bus: Generating Application-focused Databases from Large Ontologies", KRDB'98.

INPUTS: general ontology O_G (Cyc) + seed terms S

PURPOSE: Information system (Datalog⁺ database +) about this domain

METHOD

- repeat
 - from concept C (a.k.a. *type*), find n -ary predicates declared with argument of type C
 - from predicate P , gather all generalizations of concepts that are types for arguments of P
- at the end, return all formulas involving the resulting P and C (intuition: to apply $p(x,y) \wedge q(y,z) \rightarrow r(x,z)$ the types of y will be *IsA* related)
- reformulate in Datalog⁺

Ontology Winnowing (others)

- **Swartout, Patil, Knight & Russ [KAW 1996]**
"Toward Distributed Use of Large-Scale Ontologies"
- **Volz, Studer, Maedche & Lauser [J. Universal Computer Science, 2003]**
"Pruning-based Identification of Domain Ontologies"
- **Wouters, Dillon, Rahayu & Chang**
"A Practical Walkthrough of the Ontology Derivation Rules"

Conclusions: some properties of *import*

- ***Effect***: $\text{import}(S, \text{KB}_{\text{exp}}) + \text{KB}_{\text{imp}}$ has “the same logical consequences” as $\text{KB}_{\text{exp}} + \text{KB}_{\text{imp}}$, when reasoning within KB_{imp}
- ***MinPrinciple***: $\text{import}(S, \text{KB}_{\text{exp}})$ should be as small as possible for purposes of understandability (and reasoning efficiency)
- ***Expanded vocabulary S_**** : $\text{import}(S, \text{KB}_{\text{exp}})$ may contain additional symbols that help explain the meaning of identifiers in S
- ***Intelligibility***: Axioms imported need to make sense to knowledge engineers
- ***Syntax vs. Semantics***: purely syntactic/graph theoretic specifications of *import* are less desirable than ones based on logic

Sources for *import*

1. Subset of formulas as expressed in KB_{exp}

- *pro*: formulas likely understandable;
- *con*: subject to vagaries of syntactic expression: when importing names in $S=\{A,B\}$ theories $\{A \rightarrow B \wedge C\}$ vs $\{A \rightarrow B, A \rightarrow C\}$ might behave differently.

2. Subset of logical closure of KB_{exp}

- *pro*: a semantic notion --avoids problem above
- *con*: set is infinite, may not have finite representation/uncomputable, lack of intelligibility, ...

3. Syntactic subset of some *expansion* of KB_{exp}

- expansion would contain *explicitly added* or *easily explained/understood logical consequences* of KB_{exp}
- compromise solution

Varieties of *import*

1. *import1*(S, KB_{exp}) selects axioms based only on S and KB_{exp}
2. *import2*(S, KB_{exp}, KB_{imp}) also takes into consideration where symbols in S are being used.
 - hence could be much smaller set (viz *MinPple*);
 - but the material imported might change as the importer evolves
3. *import3*(S, KB_{exp}, *logic*_{imp}) considers *all* possible importing KBs expressible in *logic*_{imp}
 - Better than *import2*. Gets more complicated if the logics are very different!
4. *import4*(S, KB_{exp}, restrictions on use of imported names)
 - e.g., Navigli: only superclasses;
 - e.g., imported names from geography ontology can only be used as values for property **hasLocation**

Again, this may allow fewer axioms to be imported

Example *import*₃

Cuenca-Grau, Horrocks, Kazakov, Sattler [WWW'07,AIJ'09]

“ Choose *subset* F of formulas in KB_{exp} so that for every KB_{imp} and ψ only sharing S with $\text{vocab}(KB_{\text{exp}})$

$$KB_{\text{imp}} \cup F \models \psi \quad \text{iff} \quad KB_{\text{imp}} \cup KB_{\text{exp}} \models \psi$$

and minimize F .”

- Interesting connection to logical concept of “conservative extension”
- Though formula sets like F are hard to compute, interesting syntactic check (“locality”) allows approximation
- Minimality is not a problem because one can keep eliminating one formula at a time till the above property fails to hold
- (Expanded vocab. S_* is determined *indirectly*, by subset condition above, plus the need to get all conclusions)

Example *import*₃

B. [DL'07]

“ Choose *subset* F of formulas in

obviousConsequences(KB_{exp}) so that for

every KB_{imp} and ψ only sharing S with $\text{vocab}(\text{KB}_{\text{exp}})$

$$\text{KB}_{\text{imp}} \cup F \models \psi \quad \text{iff} \quad \text{KB}_{\text{imp}} \cup \text{KB}_{\text{exp}} \models \psi$$

which *minimizes vocab(S), and then F*”

- *obviousConsequences*(K) include
 - if $A \rightarrow B \wedge C$ in K then add $\{A \rightarrow B, A \rightarrow C\}$
 - if $A \rightarrow B$, and $B \rightarrow C$ are in K , then add $A \rightarrow C$
- although it makes sense to minimize the set of additional concepts one has to learn in order to import S , sadly the problem of computing this is NP-complete even for ordinary concept hierarchies that are not trees

IV(b). Multi-logics with “importing”

Characteristic:

“Local semantics”: denotational semantics does not assume the same domain of interpretation for all KB’s

Examples:

- **DDL (Distributed DL) [B. & Serafini ODBASE’02]**
- **[Stuckenschmidt&Klein ISCW 04] uses DDL**
- **E-connections [Cuenca Grau, Parsia, Sirin ISWC’04]**
- **P-DL [Bao, Caragea, Honavar. CTS’06, ISWC’06,...]**

Distributed DL - a version of *import4*

B. & Serafini [J of Data Semantics 2004]

Serafini, B. & Tamlin [IJCAI 2005]

Specification: $\langle T1, T2, \{ T2 \text{ imports } T1\$A \} \rangle +$ very restricted use of these imported names in T2! Only in axioms of the form

$H \rightarrow T1\$A$ /*A onto H*/ $T1\$B \rightarrow G$ /*B into G*/

(and actually, \rightarrow is not real subsumption: it is mediated by *domain relation* r_{12} connecting Domain_1 to Domain_2

$1:\text{teamBrasil} \rightarrow \{2:\text{Pele}, 2:\text{Julinho}, \dots\}$

Results:

- *specification* of DDL entailment
 $\langle T1, T2, \text{imports} \rangle \models_{\text{ddl}} 2: E \rightarrow F$
- *implementation* as distributed tableaux theorem prover
- *fixed point characterization* using $H \rightarrow G1 \vee \dots \vee Gn$ derived from **imports** statements and T1

Conclusions

- **“Importing” is different from “Modularization”**
- **Have presented some desirable criteria for general importing**
 - “information hiding” is not as significant aspect as in software development, but minimization is
 - material to be imported affected by *where* and *how* it is to be used
- **Many syntactic specifications of *import*(S,KB) possible but they are language-specific and hard to defend in general**
- **“Semantic/logical” specifications of *import* are more robust**
- **Minimizing set of new concept names S_* adds an additional degree of difficulty to minimizing the set of formulas imported**
- **Distributed KBs have special *kinds* of axioms relating their terminologies; result is quite different notion of *import*.**