MICROCONCEPTUAL-KNOWLEDGE SPREADING IN FUNGRAMKB

Carlos Periñán-Pascual
Universidad Católica San Antonio de Murcia
Campus de los Jerónimos s/n
30107 Guadalupe (Murcia)
jcperinan@pdi.ucam.edu

Francisco Arcas-Túnez
Universidad Católica San Antonio de Murcia
Campus de los Jerónimos s/n
30107 Guadalupe (Murcia)
farcas@pdi.ucam.edu

ABSTRACT
In Artificial Intelligence, the main problem in the successful development of natural language understanding systems lies on the lack of an extensive common-sense knowledge base. Common sense is mainly made up of semantic and procedural knowledge, which FunGramKB stores in the form of meaning postulates and cognitive macrostructures respectively. The objective of this paper is to describe the reasoner running on the ontology with the aim of minimizing redundancy as well as maximizing informativeness in the semantic knowledge repository of FunGramKB.

KEY WORDS
Natural language understanding, common sense, knowledge representation, meaning postulate, and reasoning.

1. Introduction

FunGramKB is a user-friendly environment for the semiautomatic construction of a multipurpose lexico-conceptual knowledge base for a natural language processing (NLP) system within the theoretical model of S.C. Dik's Functional Grammar [1]. Efficient NLP systems require a knowledge base solidly based on a linguistic theory, so that the system can identify regularities in meaning extensions, capture syntactic and semantic generalizations from lexical clusters, and establish criteria to structure and interpret new data. However, FunGramKB is not a literal implementation of Dik's lexical database, but we depart from the functional model in some important aspects with the aim of building a more robust knowledge base. FunGramKB is made up of five independent but interrelated components: lexicon, morphicon, onomasticon, cognicon and ontology. Lexical units are assigned syntactic, pragmatic and collocational information in the lexicon, but their meaning representations are conceived as conceptual properties in the ontology, so that every sense of a lexical unit is linked to a conceptual unit. The morphicon helps our system to handle cases of inflectional morphology. Names of entities, such as cities, products, etc, are stored in the onomasticon. Finally, the cognicon stores procedural knowledge by means of cognitive macrostructures, i.e. script-like schemata in which a sequence of stereotypical actions is structured on the basis of temporal continuity.

In this paper, we describe the reasoner running on the meaning postulates in the ontology with the aim of maximizing informativeness as well as minimizing redundancy.

2. Meaning Postulates in FunGramKB

Velardi *et alii* [2] distinguish two well-defined strategies when describing meaning in computational lexicography: the cognitive content in a lexical unit can be described by means of semantic features or primitives (conceptual meaning), or through associations with other lexical units in the lexicon (relational meaning). Strictly speaking, the latter doesn't give a real definition of the lexical unit, but it describes its usage in the language via 'meaning relations' with other lexical units. It is certainly easier to state associations among lexical units in the way of meaning relations than describing the cognitive content of lexical units formally, but the inference power of conceptual meaning is much stronger. Surface semantics can be sufficient in some NLP systems, but the construction of a robust knowledge base guarantees its use in most NLP tasks, consolidating thus the concept of resource reuse.

When meaning postulates are built for an NLP knowledge base in order to represent an adult speaker's linguistic competence, dictionaries must be our guide, since they are reliable repositories of information which has been judged to be relevant for lexical meaning by several generations of expert speakers [3]. Indeed, Ide and Véronis [4] recommend using several dictionaries as sources of lexical acquisition, because what a particular dictionary lacks is usually supplied by another dictionary. Machine-readable dictionaries used in the acquisition of our knowledge base are *Collins COBUILD English Dictionary* [5], *Oxford Advanced Learner's Dictionary* [6] and *Longman Web Dictionary* [7].

In FunGramKB, a meaning postulate is a set of one or more logically connected predications, which are cognitive constructs carrying the generic features of the concept[1]. To illustrate, some predications in the meaning postulates of an entity (a), event (b) and quality (c) are presented[2]:

a. BIRD
+(e$_1$: BE (x$_1$: BIRD)$_{\text{Theme}}$ (x$_2$:VERTEBRATE)$_{\text{Referent}}$)
*(e$_2$: HAVE (x$_1$)$_{\text{Theme}}$ (x$_3$: $_{\text{m}}$ FEATHER $_{\&\ 2}$ LEG $_{\&\ 2}$ WING)$_{\text{Referent}}$)
*(e$_3$: FLY (x$_1$)$_{\text{Theme}}$)

b. KISS
+(e$_1$: TOUCH (x$_1$: PERSON)$_{\text{Agent}}$ (x$_2$)$_{\text{Theme}}$ (f$_1$: $_2$ LIP)$_{\text{Instrument}}$ (f$_2$: (e$_2$: LOVE (x$_1$)$_{\text{Agent}}$ (x$_2$)$_{\text{Theme}}$) | (e$_2$: GREET (x$_1$)$_{\text{Agent}}$ (x$_2$)$_{\text{Theme}}$))$_{\text{Reason}}$)

c. HUGE
+(e$_1$: BE (x$_2$)$_{\text{Theme}}$ (x$_1$: HUGE)$_{\text{Attribute}}$)
+(e$_2$: BE (x$_1$)$_{\text{Theme}}$ (x$_3$: SIZE)$_{\text{Referent}}$)
+(e$_3$: BE (x$_2$)$_{\text{Theme}}$ (x$_4$: $_{\text{m}}$ BIG)$_{\text{Attribute}}$)

For example, predications in (a) have the following natural language equivalents:

Birds are always vertebrates.
A typical bird has many feathers, two legs and two wings.
A typical bird flies.

Dik [1] proposes using lexical units from the own language when describing meaning postulates, since meaning definition is an internal issue of the language. However, this strategy contributes to lexical ambiguity in representation due to the polysemic nature of the defining lexical units. In addition, describing the meaning of lexical units in terms of other lexical units leads to some linguistic dependency [9]. Instead, FunGramKB employs conceptual units for the formal description of meaning postulates, resulting in an interlanguage representation of meaning.

3. Monotonic Reasoning and FunGramKB

An NLP application is actually a knowledge-based system, so it must be provided with a knowledge base and a reasoning engine. Researchers in Artificial Intelligence have proved that standard logic is not able to manage common-sense reasoning. Monotonic inheritance is not able to deal with intrinsic properties of middle-level concepts in the ontology, because this type of inheritance doesn't admit exceptions to inherited default values. Therefore, non-monotonicity is a key issue in human reasoning, because it permits the withdrawal of conclusions which are true just for the typical members of a particular class.

Non-monotonic logics is an umbrella term for a family of formalisms based on default reasoning, where the system can override previous beliefs in the light of further information. Although non-monotonic logics has been widely developed in the last 25 years, resulting in formalisms such as circumscription [10], default logic [11] and autoepistemic logic [12], most of these models involve a high computational cost when being implemented in working applications which require knowledge representation. On the contrary, and despite its less-expressive power, the simplicity of defeasible logic [13] makes it one of the most efficient non-monotonic reasoning model for NLP [14, 15]. Defeasible reasoning allows the possibility of working with incomplete information, so a closed-world assumption cannot be applied.

There are usually three types of rules in a defeasible theory [13]: strict rules, defeasible rules and defeaters. Strict rules are law-like rules, which have no exceptions: e.g. whales are mammals, circles are round. On the other hand, defeasible rules can be defeated by contrary evidence: e.g. birds typically fly. Finally, defeaters are used to block some defeasible rules in order to prevent some conclusions. For example, a rule such as "if an animal is heavy, then it may not be able to fly" may override the conclusion drawn from the defeasible rule 'birds typically fly'. The superiority relation, in which a superior rule may override an inferior one, can be expressed by means of rules (e.g. $r_2 > r_1$, in case rules can be labelled) or a superiority operator.

In FunGramKB, each predication taking part in a meaning postulate is preceded by a reasoning operator in order to state if the predication is strict (+) or defeasible (*). Our inference engine handles predications as rules, allowing monotonic reasoning with strict predications, and non-monotonic with defeasible predications. The superiority relation is not explicitly stated in the predications, but the priority principle is applied at the different levels of the Microconceptual-Knowledge Spreading to resolve conflicts between predications. In other words, if two defeasible predications which meet during the spreading of the meaning postulate turn out to be contradictory, our system removes that predication being at the highest level of the spreading process. A more accurate account of this process is described in the following section.

4. Spreading Meaning Postulates

Lexical meaning is like an iceberg - only a small amount is visible from the surface, so a word is associated to much more semantic information which is really shown in its meaning postulate [16]. In FunGramKB, all this underlying cognitive information is revealed through a

1 Periñán-Pascual and Arcas-Túnez [8] describe the formal grammar of well-formed predications for meaning postulates in FunGramKB.
2 For the sake of clarity, the names of conceptual units have been oversimplified.

process called MicroKnowing (Microconceptual-Knowledge Spreading), which takes place in the ontological component of our system. This multi-level process is performed by means of two types of reasoning mechanisms: inheritance and inference. Our inheritance mechanism strictly involves the transfer of one or several predications from a superordinate concept to a subordinate one in the ontology. On the other hand, our inference mechanism is based on the structures shared between predications linked to conceptual units which do not take part in the same subsumption relation within the ontology.

The MicroKnowing process can be formally stated as follows:

$$\Omega_i \Rightarrow \Phi_0 \cup \Phi_1 \cup \ldots \Phi_n \Rightarrow \bigcup_{i=0}^{n} \Phi_i$$

$$\Phi_i \Rightarrow \varphi_1 \cup \varphi_2 \cup \ldots \varphi_k \Rightarrow \bigcup_{j=1}^{k} \varphi_j$$

The MicroKnowing of a particular concept, which is called 'nuclear concept', is originated in the meaning postulate linked to that conceptual unit. Given that φ represents one particular inherited or inferred predication at a particular spreading level, Φ_i is defined as the set of predications being inherited or inferred at spreading level *i* providing that $i \geq 1$, since Φ_0 represents the nuclear meaning postulate. On the other hand, Ω_i represents the extended meaning postulate which has just been spread at level *i* providing that $i \geq 1$, since Ω_0 is equivalent to Φ_0. Assuming that *n* is the total number of levels which are necessary to spread the meaning postulate completely, where $n \leq (2d - 1)$ being *d* the length of the path from the nuclear concept to the father metaconcept, Ω_n represents the final output of a totally-extended meaning postulate.

To illustrate the MicroKnowing, let's take the concept TURKEY, whose meaning postulate is the following one:

+(e$_1$: BE (x$_1$: TURKEY)$_{\text{Theme}}$ (x$_2$: POULTRY)$_{\text{Referent}}$)
*(e$_2$: BE (x$_1$)$_{\text{Theme}}$ (x$_3$: BIG)$_{\text{Attribute}}$)
*(e$_3$: $_n$ HAVE (x$_1$)$_{\text{Theme}}$ (x$_4$: FEATHER)$_{\text{Referent}}$) (f$_1$: HEAD $_\&$ NECK)$_{\text{Location}}$)

At the first spreading level, the inference mechanism is triggered, where the nuclear concept (i.e. TURKEY) infers a predication from the meaning postulate of another concept providing that the nuclear concept acts as the selection preference in any constituent of the meaning postulate of the target concept. For example, the following predication is inferred from the meaning postulate of WATTLE:

*(e$_2$: HAVE (x$_3$: TURKEY)$_{\text{Theme}}$ (x$_1$: WATTLE)$_{\text{Referent}}$)

Now the inferred predication is incorporated into the nuclear meaning postulate, resulting in a first version of the extended meaning postulate at the first level. Therefore:

$$\Omega_1 => \Omega_0 \cup \varphi_1$$

FunGramKB automatically readjusts indices for variables *e* (predication), *x* (argument) and *f* (satellite) in inferred or inherited predications.

However, our inference model ignores those predications in which the *genus* of the target concept is the nuclear concept. FunGramKB's reasoner recognises these cases because the nuclear concept appears as a selection preference of the Referent argument in the first predication of the meaning postulate. This is the case of GOBBLER:

*(e$_1$: BE (x$_1$: GOBBLER)$_{\text{Theme}}$ (x$_2$: TURKEY)$_{\text{Referent}}$)

On the other hand, the inheritance mechanism is triggered at the second spreading level. Now the extended meaning postulate inherits those predications in the meaning postulate linked to the immediate superordinate of the nuclear concept. In our case, the father of TURKEY is POULTRY, whose meaning postulate is as follows:

+(e$_1$: BE (x$_1$: POULTRY)$_{\text{Theme}}$ (x$_2$: BIRD)$_{\text{Referent}}$)
*(e$_2$: BE (x$_1$)$_{\text{Theme}}$ (x$_3$: DOMESTIC)$_{\text{Attribute}}$)
*(e$_3$: KEEP (x$_4$: PERSON)$_{\text{Theme}}$ (x$_1$)$_{\text{Referent}}$ (f$_1$: FARM)$_{\text{Location}}$)
*(e$_4$: OBTAIN (x$_5$: PERSON)$_{\text{Theme}}$ (x$_6$: EGG, MEAT)$_{\text{Theme}}$ (f$_2$: x$_1$)$_{\text{Origin}}$)

From this point on, both inference (I) and inheritance (H) mechanisms are cyclically applied to any genus in any meaning postulate within the conceptual path between the nuclear concept and its metaconcept. In figure 1, circles represent conceptual units, whose meaning postulates take an active part in the reasoning task:

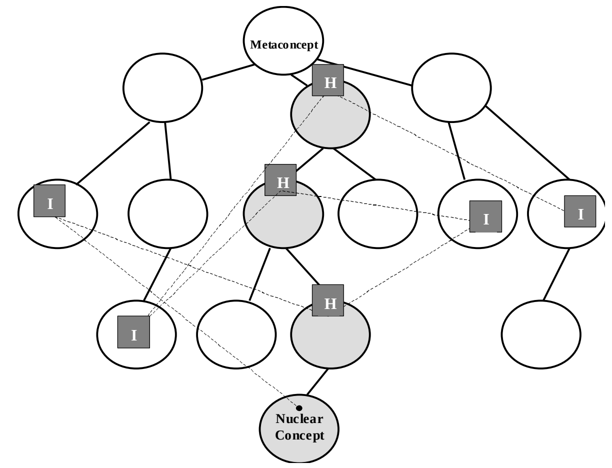

Figure 1. Inference and Inheritance in FunGramKB's Ontology.

The following table presents some of the output in the MicroKnowing for the nuclear concept TURKEY, where levels 1 and 4 happen with inference, and levels 2, 3 and 5 with inheritance:

Level	Meaning Postulate	Activator
Φ_0	+ (e_1: BE (x_1: TURKEY)$_{Theme}$ (x_2: POULTRY)$_{Referent}$)	TURKEY
	*(e_2: BE (x_1)$_{Theme}$ (x_3: BIG)$_{Attribute}$)	TURKEY
	*(e_3: $_n$ HAVE (x_1)$_{Theme}$ (x_4: FEATHER)$_{Referent}$) (f_1: HEAD $_{\&}$ NECK)$_{Location}$)	TURKEY
Φ_1	+(e_4: HAVE (x_1)$_{Theme}$ (x_5: WATTLE)$_{Referent}$)	WATTLE
	+(e_5: GOBBLE (x_1)$_{Theme}$)	GOBBLE
	+(e_6: OBTAIN (x_6: PERSON)$_{Agent}$ (x_7: TURKEY_01)$_{Theme}$ (f_2: x_1)$_{Origin}$)	TURKEY_01
Φ_2	+(e_7: BE (x_1)$_{Theme}$ (x_8: BIRD)$_{Referent}$)	POULTRY
	*(e_8: BE (x_1)$_{Theme}$ (x_9: DOMESTIC)$_{Attribute}$)	POULTRY
	*(e_9: KEEP (x_{10}: PERSON)$_{Theme}$ (x_1)$_{Referent}$ (f_3: FARM)$_{Location}$)	POULTRY
Φ_3	+(e_{10}: BE (x_1)$_{Theme}$ (x_{11}: VERTEBRATE)$_{Referent}$)	BIRD
	*(e_{11}: HAVE (x_1)$_{Theme}$ (x_{12}: FEATHER)$_{Referent}$)	BIRD
	*(e_{12}: HAVE (x_1)$_{Theme}$ (x_{13}: LEG)$_{Referent}$)	BIRD
	*(e_{13}: HAVE (x_1)$_{Theme}$ (x_{14}: WING)$_{Referent}$)	BIRD
	*(e_{14}: LAY (x_1)$_{Agent}$ (x_{15}: EGG)$_{Theme}$)	BIRD
	*(e_{15}: FLY (x_1)$_{Theme}$ (f_4: WING)$_{Instrument}$)	BIRD $\cup$ WING
Φ_4	*(e_{16}: CHIRP (x_1)$_{Theme}$)	CHIRP
	*(e_{17}: HAVE (x_1)$_{Theme}$ (x_{16}: CLAW)$_{Referent}$ (f_5: TOE)$_{Location}$)	CLAW
	*(e_{18}: PECK (x_1)$_{Agent}$ (x_{17}: THING)$_{Theme}$)	PECK
	*(e_{19}: HAVE (x_1)$_{Theme}$ (x_{18}: BEAK)$_{Referent}$)	BEAK
	*(e_{20}: PREEN (x_1)$_{Theme}$)	PREEN
Φ_5	+(e_{21}: BE (x_1)$_{Theme}$ (x_{19}: ANIMAL)$_{Referent}$)	VERTEBRATE
	+(e_{22}: CONTAIN (x_1)$_{Theme}$ (x_{20}: SKELETON)$_{Referent}$)	VERTEBRATE
	+(e_{23}: CONTAIN (x_1)$_{Theme}$ (x_{21}: BACKBONE)$_{Referent}$)	VERTEBRATE
	*(e_{24}: HAVE (x_1)$_{Theme}$ (x_{22}: TAIL)$_{Referent}$ (f_6: BACK)$_{Location}$)	VERTEBRATE

A key issue in defeasible reasoning is how to process the conflict that arises with competing rules. In FunGramKB, conflicts are presented between contradictory predications. When our system needs to check if predication φ is compatible with the predications within meaning postulate Ω, our reasoning engine uses both strict and defeasible predications. If φ is a strict predication, then it is automatically incorporated into Ω. On the contrary, if φ is a defeasible predication, then compatibility with predications in Ω must be validated; in other words, predication φ is compatible with predications in Ω if and only if Ω does not contain a predication ¬φ. In order to implement the resolution method of incompatible predications, it is necessary to apply the 'stepwise conceptual decomposition' (SCD) of all predications involved in the MicroKnowing.

The SCD is founded on the 'stepwise lexical decomposition' principle proposed by Dik [1]. This principle establishes a way of interrelating lexical entries where the *definiens* in a meaning postulate can be converted into the *definiendum* of another meaning postulate. The fact that the functional model is provided with a lexical decomposition process enables the construction of meaning postulates in a simple fashion, as well as minimizing information redundancy. In FunGramKB, the SCD is similar to Dik's principle, apart from the fact that the main building blocks of our meaning representations are not lexical units but concepts. Therefore, the SCD is defined as the process in which conceptual units in a predication are substituted by their respective meaning postulates until a meaning representation composed of basic concepts is reached. When the SCD is applied in the MicroKnowing, the nuclear concept is the only one which does not undergo this SCD process; otherwise, a large number of predications would be repeated. The SCD of a predication φ which is incorporated into an extended meaning postulate Ω is essential in order to check consistency of φ with Ω, that is, if there is no ¬φ in Ω. The compatibility problem can be efficiently tackled just in case two predications are compared with the same conceptual depth, and more particularly at the root level. For example, the extended meaning postulate of TURKEY has predications [a] and [b] among others and we want to know if predication [c] can be inferred from AVIARY so that conclusion [d] can be reached:

[a] +(e_7: BE (x_1: TURKEY)$_{Theme}$ (x_8: BIRD)$_{Referent}$)
[b] *(e_2: BE (x_1: TURKEY)$_{Theme}$ (x_3: BIG)$_{Attribute}$)
[c] *((e_{18}: KEEP (x_{19}: PERSON)$_{Theme}$ (x_1: BIRD)$_{Referent}$ (f_4: AVIARY)$_{Location}$) (e_{19}: BE (x_1)$_{Theme}$ (x_{20}: SMALL)$_{Attribute}$))

---

[d] *(e_{18}: KEEP (x_{19}: PERSON)$_{Theme}$ (x_1: TURKEY)$_{Referent}$ (f_4: AVIARY)$_{Location}$)

Predications [a], [b] and [c] are compatible at this level of decomposition, but after the SCD the system recognises that (b) is in conflict with (c) because of predication [e] in the meaning postulate of SMALL, so predication [c] is rejected:

SMALL
+(e_1: BE $(x_2)_{Theme}$ $(x_1$: SMALL$)_{Attribute}$)
+(e_2: BE $(x_1)_{Theme}$ $(x_3$: SIZE$)_{Referent}$)
[e] +(e_3: $_n$ BE $(x_2)_{Theme}$ $(x_4$: BIG$)_{Attribute}$)

The most relevant advantage of this reasoner for an NLP knowledge base is to avoid unnecessary duplication of information, as well as the possibility of updating the knowledge base in a consistent way. When the language engineer modifies an existing meaning postulate or builds a new one, just before being stored, FunGramKB automatically performs the MicroKnowing for that meaning postulate in order to check the compatibility of the newly-incorporated predications with other predications involved in the reasoning process. The language engineer is informed about all those inferred or inherited predications which are not true for the nuclear concept. In addition, FunGramKB displays the whole MicroKnowing process step by step, enabling us to verify inference and inheritance conditions in a transparent way.

5. Computational Implementation of Meaning Postulates

Language engineers enter meaning postulates through a dedicated editor in FunGramKB's ontology interface. The construction of our knowledge base is semiautomatic in the sense that human intervention is required but the engineer's intuition is guided and reviewed through a series of user-friendly tools for lexico-conceptual acquisition. For example, when creating or modifiying meaning postulates, a syntactic-semantic checker is activated, so that consistent well-formed meaning postulates can be stored. When a meaning postulate is stored, a parser outputs an XML-formatted feature-value structure used as the input for the reasoning engine, so that inheritance and inference mechanisms can be applied. Both the syntactic-semantic validator of meaning postulates and the XML parser were written in C#. XML was chosen as the language for knowledge representation in FunGramKB because it helps the system with the establishment of premises for structured data transfer, the autonomous separation of knowledge from its representation formalism, and the atomization of the various components the meaning postulate is made up of, speeding up in this way the inference process and the direct access to particular conceptual units in the meaning postulate. To illustrate, figure 2 presents the XML representation of some predications in the meaning postulate of concept SILVER:

SILVER
+(e_1: +BE_00 $(x_1$: +SILVER_00$)_{Theme}$ $(x_2$: +METAL_00$)_{Referent}$)
*(e_2: +BE_00 $(x_1)_{Theme}$ $(x_3$: +EXPENSIVE_00$)_{Attribute}$)
+(e_3: +TRAVEL_00 $(x_4$: +ELECTRICITY_00$)_{Theme}$ $(f_1$: $x_1)_{Means}$)

```
<S>
    <e N="1" OPr="+">
        <LBCv>+BE_00</LBCv>
        <x N="1" SFx="Theme">
            <LBCea>+SILVER_00</ LBCea >
        </x>
        <x N="2" SFx="Referent">
            <LBCea>+METAL_00</LBCea>
        </x>
    </e>
    <e N="2" OPr="*">
        <LBCv>+BE_00</LBCv>
        <x N="1" SFx="Theme"/>
        <x N="3" SFx="Attribute">
            <LBCea>+EXPENSIVE_00</LBCea>
        </x>
    </e>
    <e N="3" OPr="+">
        <LBCv>+TRAVEL_00</LBCv>
        <x N="4" SFx="Theme">
            <LBCea>+ELECTRICITY_00</LBCea>
        </x>
        <f N="1" SFf="Means">
            <x>1</x>
        </f>
    </e>
</S>
```

Figure 2. An XML Meaning Postulate.

6. Conclusion

FunGramKB is a workbench for the semiautomatic construction of a multipurpose knowledge base for NLP systems, mainly those requiring natural language understanding. In this paper we have presented the multi-level process of spreading those knowledge representations linked to ontological concepts. In particular, we have shown that the cyclical application of the inheritance and inference mechanisms on our meaning postulates allow FunGramKB to minimize redundancy as well as keeping our knowledge base as informative as possible.

Since lexical meaning is not enough to build human common-sense, FunGramKB is provided with general procedural knowledge through the cognicon. We are currently working on the MacroKnowing (Macroconceptual-Knowing Spreading), i.e. the process of integrating meaning postulates from the ontology with the cognitive macrostructures in the cognicon. This combination of semantic and procedural knowledge, so distinctive of human reasoning, is hardly found in NLP systems to date.

7. Acknowledgements

FunGramKB is part of the research carried out in the project "Implementación de una base de conocimiento para el razonamiento del sentido común" funded by Universidad Católica San Antonio de Murcia, code number PMAFI-PI-11/1C/05.

References:

[1] S.C. Dik, *The Theory of Functional Grammar* (Berlin-New York: Mouton de Gruyter, 1997 (1989)).

[2] P. Velardi, M.T. Pazienza, & M. Fasolo, How to encode semantic knowledge: a method for meaning representation and computer-aided acquisition, *Computational Linguistics 17*(2), 1991, 153-170.

[3] D. Marconi, *Lexical Competence* (Cambridge, Mass.: MIT Press, 1997).

[4] N. Ide & J. Véronis, Machine readable dictionaries: what have we learned, where do we go?, *Proc. Int. Workshop on the Future of Lexical Research*, Beijing, 1994, 137-146.

[5] J. Sinclair, ed., *Collins COBUILD English Dictionary* (London: Collins, 1995 (1987)).

[6] A.S. Hornby, *Oxford Advanced Learner's Dictionary of Current English* (Oxford: Oxford University Press, 2003). [http://www.oup.com/elt/global/products/oald]

[7] Longman Group, *Longman Web Dictionary* (Harlow: Longman, 2001). [http://www.longmanwebdict.com]

[8] C. Periñán-Pascual & F. Arcas-Túnez, Meaning postulates in a lexico-conceptual knowledge base, *Proc. 15th International Workshop on Database and Expert Systems Applications*, Zaragoza, 2004, 38-42.

[9] P. Vossen, The end of the chain: where does decomposition of lexical knowledge lead us eventually?, E. Engberg-Pedersen, L. Falster Jakobsen, & L. Schack Rasmussen, eds., *Function and Expression in Functional Grammar* (Berlin-New York: Mouton de Gruyter, 1994, 11-39).

[10] J. McCarthy, Circumscription – a form of non-monotonic reasoning, *Artificial Intelligence 13*, 1980, 27-39.

[11] R. Reiter, A logic for default reasoning, *Artificial Intelligence 13*, 1980, 81-132.

[12] R.C. Moore, Semantical considerations on nonmonotonic logic, *Proc. 8th International Joint Conference on Artificial Intelligence*, Karlsruhe, 1983, 272-279.

[13] D. Nute, Defeasible Prolog, *AAAI Fall Symposium on Automated Deduction and Nonstandard Logics*, Raleigh, NC., 1993, 105-112.

[14] G. Antoniou, D. Billington, G. Governatori, & M. J. Maher, A flexible framework for defeasible logics, *Proc. 8th Workshop on Non-monotonic Reasoning*, Breckenridge, Co., 2000, 405-410.

[15] M. J. Maher, A. Rock, G. Antoniou, D. Billington, & T. Miller, Efficient defeasible reasoning systems, *International Journal on Artificial Intelligence Tools 10*(4), 2001, 483-501.

[16] W. Peters & A. Kilgarriff, Discovering semantic regularity in lexical resources, *International Journal of Lexicography 13*(4), 2000, 287-312.